[MS-AUTHWS]:

Authentication Web Service Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional

development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1		Initial Availability
6/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
7/13/2009	1.02	Major	Revised and edited the technical content
8/28/2009	1.03	Editorial	Revised and edited the technical content
11/6/2009	1.04	Editorial	Revised and edited the technical content
2/19/2010	2.0	Minor	Updated the technical content
3/31/2010	2.01	Editorial	Revised and edited the technical content
4/30/2010	2.02	Editorial	Revised and edited the technical content
6/7/2010	2.03	Editorial	Revised and edited the technical content
6/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.04	No Change	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	2.04	No Change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	No Change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	No Change	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.04	No Change	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	2.04	No Change	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	3.0	Major	Significantly changed the technical content.
4/11/2012	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	3.1	Minor	Clarified the meaning of the technical content.
2/11/2013	3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	3.1	No Change	No changes to the meaning, language, or formatting of the

Date	Revision History	Revision Class	Comments
			technical content.
2/10/2014	3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	3.2	Minor	Clarified the meaning of the technical content.
7/31/2014	3.2	No Change	No changes to the meaning, language, or formatting of the technical content.
8/24/2015	4.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

This document specifies the Authentication Web Service Protocol. This protocol enables a protocol client to determine which type of **authentication** is used by a **website**. In addition, if authentication requests for that site are redirected to an HTML form, then this protocol enables a protocol client and a protocol server to authenticate a user.

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.

authentication mode: One of several modes in which an authentication exchange may be performed.

cookie: A small data file that is stored on a user's computer and carries state information between participating protocol servers and protocol clients.

forms authentication: An **authentication** method in which protocol clients redirect unauthenticated requests to an **HTML** form by using **HTTP**. If the protocol client authenticates the request, the system issues a **cookie** that stores the credentials or a key for reacquiring the identity. In subsequent requests, the cookie is submitted in request headers and the requests are authenticated and authorized by an ASP.NET event handler that uses the validation method that is specified by the protocol client.

Hypertext Markup Language (HTML): An application of the Standard Generalized Markup Language (SGML) that uses tags to mark elements in a document, as described in [HTML].

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 Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [SSL3] and [RFC5246].

Internet Information Services (IIS): The services provided in Windows implementation that support web server functionality. **IIS** consists of a collection of standard Internet protocol servers such as HTTP and FTP in addition to common infrastructures that are used by other Microsoft Internet protocol servers such as SMTP, NNTP, and so on. **IIS** has been part of the Windows operating system in some versions and a separate install package in others. **IIS** version 5.0 shipped as part of Windows 2000 operating system, **IIS** version 5.1 as part of Windows XP operating system, **IIS** version 6.0 as part of Windows Server 2003 operating system, and **IIS** version 7.0 as part of Windows Vista operating system and Windows Server 2008 operating system.

replay attack: An attempt to circumvent an **authentication** protocol by copying authentication messages from a legitimate protocol client and resending them to the protocol server during an authentication process.

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data-a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client **authentication** using X.509 certificates (2). For more information, see [X509]. The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [SSL3].

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 action: The HTTP request header field used to indicate the intent of the **SOAP** request, using a URI value. See [SOAP1.1] section 6.1.1 for more information.

SOAP body: A container for the payload data being delivered by a **SOAP message** to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.

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.

ticket: A record generated by the key distribution center (KDC) that helps a client authenticate to a service. It contains the client's identity, a unique cryptographic key for use with this ticket (the session key), a time stamp, and other information, all sealed using the service's secret key. It only serves to authenticate a client when presented along with a valid authenticator.

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 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].

web application: A container in a configuration database that stores administrative settings and entry-point **URLs** for site collections.

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.

website: A group of related pages and data within a SharePoint site collection. The structure and content of a site is based on a site definition. Also referred to as SharePoint site and site.

Windows Live ID: A web-based service that enables participating sites to authenticate a user with a single set of credentials.

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.

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 namespace prefix: An abbreviated form of an **XML namespace**, as described in [XML].

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

1.2 References

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

1.2.1 Normative References

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

[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

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

[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

[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

[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

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

1.3 Protocol Overview (Synopsis)

This protocol enables a protocol client to determine which **authentication mode** is used by a **web application**. If the web application uses **forms authentication**, this protocol also enables a protocol client and a protocol server to authenticate a user.

A typical scenario for implementing this protocol is one in which forms authentication is used to programmatically log a user onto an application and authenticate subsequent requests by that user.

1.4 Relationship to Other Protocols

This protocol uses the **SOAP** message protocol for formatting request and response messages, as described in [SOAP1.1], [SOAP1.2/1] and [SOAP1.2/2]. It transmits those messages by using **HTTP**, as described in [RFC2616], or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**, as described in [RFC2818].

The following diagram shows the underlying messaging and transport stack used by the protocol:

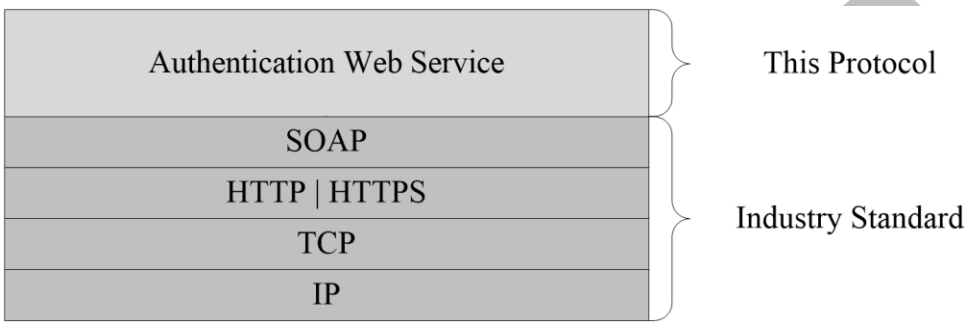

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a website that is identified by a **URL** that is known by protocol clients. The protocol server endpoint is formed by appending "/_vti_bin/Authentication.asmx" to the URL of the site: for example, http://www.example.com/Repository/_vti_bin/Authentication.asmx.

1.6 Applicability Statement

This protocol applies to the following scenarios:

- Retrieving the authentication mode that a specified web application uses.
- By using the logon name and password for a user, logging a user onto a web application that is using forms authentication.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol servers MUST support SOAP over HTTP. Protocol servers SHOULD additionally support SOAP over HTTPS to help secure communication with protocol clients.

Protocol messages MUST be formatted as specified in [SOAP1.1] section 4 or [SOAP1.2/1] section 5. Protocol server faults MUST be returned by using either HTTP status codes, as specified in [RFC2616] section 10, or **SOAP faults**, as specified in [SOAP1.1] section 4.4 or [SOAP1.2/1] section 5.4.

2.2 Common Message Syntax

This section contains common definitions used by this protocol. The syntax of the definitions uses the XML Schema, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and Web Services Description Language, as specified in [WSDL].

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
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] [SOAP1.2/2]
tns	http://schemas.microsoft.com/sharepoint/soap/	
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
(none)	http://schemas.microsoft.com/sharepoint/soap/	

2.2.2 Messages

None.

2.2.3 Elements

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

2.2.4 Complex Types

This specification does not define any common XML Schema complex type definitions.

2.2.5 Simple Types

This specification does not define any common XML Schema simple type definitions.

2.2.6 Attributes

This specification does not define any common XML Schema attribute definitions.

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.

This protocol allows protocol servers to perform implementation-specific authorization checks and to notify protocol clients of authorization faults by using either HTTP status codes or SOAP faults. Except where specified otherwise, protocol clients SHOULD interpret HTTP status codes as specified in [RFC2616] section 10. This protocol allows protocol servers to notify protocol clients of application-level faults by using SOAP faults. Except where specified otherwise, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

3.1 Server Details

All of the operations that are defined by this protocol consist of a basic request/response pair, and the protocol server treats each request as an independent transaction that is unrelated to any previous request.

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of **WSDL operations** that are defined by this protocol.

Operation	Description
Login	Logs a user onto an application by using the user's logon name and password.
Mode	Retrieves the authentication mode that a web application uses.

3.1.4.1 Login

The **Login** operation logs a user onto a web application by using the user's logon name and password. For the operation to succeed, the protocol server MUST use forms authentication and the logon name and password that is provided by the protocol client MUST be valid. If the operation succeeds, a **ticket** for the specified user is created and it is attached to a **cookie** collection that is associated with the outgoing response. A redirect to the HTML login form is not performed.

```
<wsdl:operation name="Login">
    <wsdl:input message="tns:LoginSoapIn" />
    <wsdl:output message="tns:LoginSoapOut" />
</wsdl:operation>
```

The protocol client sends a **LoginSoapIn** request WSDL message and the protocol server responds with a **LoginSoapOut** response WSDL message, as specified in section 3.1.4.1.1.2.

3.1.4.1.1 Messages

The following WSDL message definitions are specific to this operation.

3.1.4.1.1.1 LoginSoapIn

The **LoginSoapIn** message is the request WSDL message that is used by a protocol client when logging on a user.

The **SOAP action** value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/Login
```

The **SOAP body** contains a **login** element, as specified in section 3.1.4.1.2.1.

3.1.4.1.1.2 LoginSoapOut

The **LoginSoapOut** message is the response WSDL message that is used by a protocol server when logging on a user in response to a **LoginSoapIn** request message.

The SOAP body contains a **LoginResponse** element, as specified in section 3.1.4.1.2.2.

3.1.4.1.2 Elements

The following XML Schema element definitions are specific to this operation.

3.1.4.1.2.1 Login

The **Login** element defines the input parameters for the **Login** WSDL operation.

```
<s:element name="Login">
  <s:complexType>
    <s:sequence>
      <s:element name="username" type="s:string" minOccurs="0"/>
      <s:element name="password" type="s:string" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

username: A string that specifies the logon name of the user.

password: A string that specifies the password for the user.

3.1.4.1.2.2 LoginResponse

The **LoginResponse** element defines the output of the **Login** WSDL operation.

```
<s:element name="LoginResponse">
```

```
    <s:complexType>
      <s:sequence>
        <s:element name="LoginResult" type="tns:LoginResult"/>
      </s:sequence>
    </s:complexType>
  </s:element>
```

LoginResult: A **LoginResult** complex type, as specified in section 3.1.4.1.3.1.

3.1.4.1.3 Complex Types

The following XML Schema complex type definitions are specific to this operation.

3.1.4.1.3.1 LoginResult

The **LoginResult** complex type contains an error code and, if a **Login** WSDL operation succeeded, the name of an authentication cookie.

```
<s:complexType name="LoginResult">
  <s:sequence>
    <s:element name="CookieName" type="s:string" minOccurs="0"/>
    <s:element name="ErrorCode" type="tns:LoginErrorCode"/>
    <s:element name="TimeoutSeconds" type="s:int" minOccurs="0" maxOccurs="1"/>
  </s:sequence>
</s:complexType>
```

CookieName: A string that specifies the name of the cookie that is used to store the forms authentication ticket. The default value is "FedAuth".<1> This element MUST NOT be present if the **Login** WSDL operation failed.

ErrorCode: An error code, as specified in section 3.1.4.1.4.1.

TimeoutSeconds: An integer that specifies the number of seconds before the cookie, which is specified in the **CookieName** element, expires.<2>

3.1.4.1.4 Simple Types

The following XML Schema simple type definitions are specific to this operation.

3.1.4.1.4.1 LoginErrorCode

The **LoginErrorCode** simple type indicates the result of a **Login** WSDL operation.

```
<s:simpleType name="LoginErrorCode">
  <s:restriction base="s:string">
    <s:enumeration value="NoError"/>
    <s:enumeration value="NotInFormsAuthenticationMode"/>
    <s:enumeration value="PasswordNotMatch"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for the **LoginErrorCode** simple type:

Value	Description
NoError	The **Login** operation succeeded.

Value	Description
NotInFormsAuthenticationMode	The **Login** operation failed because the authentication mode is not set to forms authentication.
PasswordNotMatch	The **Login** operation failed because the logon name is not found by the server, or the password does not match what is stored on the server.

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

None.

3.1.4.1.7 Attribute Groups

None.

3.1.4.2 Mode

The **Mode** operation retrieves the authentication mode that a web application uses.

```
<wsdl:operation name="Mode">
    <wsdl:input message="tns:ModeSoapIn" />
    <wsdl:output message="tns:ModeSoapOut" />
</wsdl:operation>
```

The protocol client sends a **ModeSoapIn** request WSDL message and the protocol server responds with a **ModeSoapOut** response WSDL message.

3.1.4.2.1 Messages

The following WSDL message definitions are specific to this operation.

3.1.4.2.1.1 ModeSoapIn

The **ModeSoapIn** message is the request WSDL message that a protocol client uses to retrieve the authentication mode.

The SOAP action value of the message is defined as:

```
http://schemas.microsoft.com/sharepoint/soap/Mode
```

The SOAP body contains a **Mode** element, as specified in section 3.1.4.2.2.1.

3.1.4.2.1.2 ModeSoapOut

The **ModeSoapOut** message is the response WSDL message that a protocol server sends after retrieving the authentication mode.

The SOAP body contains a **ModeResponse** element, as specified in section 3.1.4.2.2.2.

3.1.4.2.2 Elements

The following XML Schema element definitions are specific to this operation.

3.1.4.2.2.1 Mode

The **Mode** element specifies the **Mode** WSDL operation.

```
<s:element name="Mode">
  <s:complexType/>
</s:element>
```

3.1.4.2.2.2 ModeResponse

The **ModeResponse** element specifies the output of the **Mode** WSDL operation.

```
<s:element name="ModeResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="ModeResult" type="tns:AuthenticationMode"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

ModeResult: An **AuthenticationMode** simple type, as specified in section 3.1.4.2.4.1.

3.1.4.2.3 Complex Types

None.

3.1.4.2.4 Simple Types

The following XML Schema simple type definitions are specific to this operation.

3.1.4.2.4.1 AuthenticationMode

The **AuthenticationMode** simple type specifies the authentication mode for the **Mode** WSDL operation.

```
<s:simpleType name="AuthenticationMode">
  <s:restriction base="s:string">
    <s:enumeration value="None"/>
    <s:enumeration value="Windows"/>
    <s:enumeration value="Passport"/>
    <s:enumeration value="Forms"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for the **AuthenticationMode** simple type.

Value	Description
None<3>	No authentication is used or a custom authentication scheme is used.

Value	Description
Windows	Authentication is handled by **Internet Information Services (IIS)**. The application context uses a security token that is received from IIS.
Passport<4>	**Windows Live ID** is used for authentication.
Forms	A protocol client submits credentials by using an **HTML** form. If the protocol server authenticates the protocol client, it issues a cookie to the protocol client and the protocol client presents that cookie in subsequent requests.

3.1.4.2.5 Attributes

None.

3.1.4.2.6 Groups

None.

3.1.4.2.7 Attribute Groups

None.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

4.1 Retrieving the Authentication Mode

In this example, a protocol client sends the following **SOAP message** to retrieve the authentication mode:

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

The protocol server uses forms authentication and, therefore, responds with the following SOAP message:

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

4.2 Logging On a User

In this example, a protocol client sends the following SOAP message to log on a user whose name is Anat Kerry:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <Login xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <username>Anat Kerry</username>
      <password>password</password>
    </Login>
  </soap:Body>
</soap:Envelope>
```

The protocol server uses forms authentication and authenticates Anat Kerry. Therefore, the protocol server responds with the following SOAP message:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <LoginResponse xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <LoginResult>
        <CookieName>.ASPXAUTH</CookieName>
```

```
          <ErrorCode>NoError</ErrorCode>
          <TimeoutSeconds>180</TimeoutSeconds>
        </LoginResult>
      </LoginResponse>
    </soap:Body>
  </soap:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

The **Login** WSDL operation requires that a user's logon name and password be sent as plain text in the body of the request WSDL message. Therefore, the message is inherently not secure. In addition, forms authentication is subject to **replay attacks** for the lifetime of the cookie. To help increase the security of the message, use of **Secure Sockets Layer (SSL)** and **Transport Layer Security (TLS)** is recommended.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full **WSDL** is provided below:

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://schemas.microsoft.com/sharepoint/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/">
      <s:element name="Login">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" name="username" type="s:string" />
            <s:element minOccurs="0" name="password" type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="LoginResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="LoginResult" type="tns:LoginResult" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="LoginResult">
        <s:sequence>
          <s:element minOccurs="0" name="CookieName" type="s:string" />
          <s:element name="ErrorCode" type="tns:LoginErrorCode" />
          <s:element minOccurs="0" maxOccurs="1" name="TimeoutSeconds" type="s:int" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="LoginErrorCode">
        <s:restriction base="s:string">
          <s:enumeration value="NoError" />
          <s:enumeration value="NotInFormsAuthenticationMode" />
          <s:enumeration value="PasswordNotMatch" />
        </s:restriction>
      </s:simpleType>
      <s:element name="Mode">
        <s:complexType />
      </s:element>
      <s:element name="ModeResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="ModeResult" type="tns:AuthenticationMode" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:simpleType name="AuthenticationMode">
        <s:restriction base="s:string">
          <s:enumeration value="None" />
          <s:enumeration value="Windows" />
          <s:enumeration value="Passport" />
          <s:enumeration value="Forms" />
        </s:restriction>
      </s:simpleType>
    </s:schema>
  </wsdl:types>
```

```
  <wsdl:message name="LoginSoapIn">
    <wsdl:part name="parameters" element="tns:Login" />
  </wsdl:message>
  <wsdl:message name="LoginSoapOut">
    <wsdl:part name="parameters" element="tns:LoginResponse" />
  </wsdl:message>
  <wsdl:message name="ModeSoapIn">
    <wsdl:part name="parameters" element="tns:Mode" />
  </wsdl:message>
  <wsdl:message name="ModeSoapOut">
    <wsdl:part name="parameters" element="tns:ModeResponse" />
  </wsdl:message>
  <wsdl:portType name="AuthenticationSoap">
    <wsdl:operation name="Login">
      <wsdl:input message="tns:LoginSoapIn" />
      <wsdl:output message="tns:LoginSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="Mode">
      <wsdl:input message="tns:ModeSoapIn" />
      <wsdl:output message="tns:ModeSoapOut" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="AuthenticationSoap" type="tns:AuthenticationSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="Login">
      <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/Login"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Mode">
      <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/Mode"
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="AuthenticationSoap12" type="tns:AuthenticationSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="Login">
      <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/Login"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Mode">
      <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/Mode"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
```

```
</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 SharePoint Foundation 2010
- Windows SharePoint Services 3.0
- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016 Preview

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

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

<1> Section 3.1.4.1.3.1: Windows SharePoint Services 3.0 returns the default value of ".ASPXAUTH".

<2> Section 3.1.4.1.3.1: Windows SharePoint Services 3.0 does not return this element.

<3> Section 3.1.4.2.4.1: Microsoft SharePoint Foundation 2010 Service Pack 1 returns "Forms" when the value for **AuthenticationMode** is "None".

<4> Section 3.1.4.2.4.1: Use of Windows Live ID for authentication is not supported by Windows Server 2008 operating system with Service Pack 2 (SP2) and Windows Server 2012 operating system.

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