[MS-OFBA]:
Office Forms Based Authentication 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.

Revision Summary

Date	Revision History	Revision Class	Comments
05/16/2008	0.1		Initial Availability
10/06/2008	0.2	Editorial	Revised and edited the technical content
01/16/2009	1.0	Major	Revised and edited the technical content
07/13/2009	1.01	Major	Changes made for template compliance
08/28/2009	1.02	Editorial	Revised and edited the technical content
11/06/2009	1.03	Editorial	Revised and edited the technical content
02/19/2010	2.0	Editorial	Revised and edited the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Editorial	Revised and edited the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Minor	Clarified the meaning of the technical content.
07/23/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	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.
03/18/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	2.5	Minor	Clarified the meaning of the technical content.
04/11/2012	2.5	No change	No changes to the meaning, language, or formatting of the technical content.
07/16/2012	2.6	Minor	Clarified the meaning of the technical content.
09/12/2012	2.6	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2012	2.6	No change	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
02/11/2013	2.6	No change	No changes to the meaning, language, or formatting of the technical content.
07/30/2013	2.6	No change	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	2.7	Minor	Clarified the meaning of the technical content.

Table of Contents

1 Introduction

The Office Forms Based Authentication Protocol provides protocol clients and servers with HTTP forms-based authentication when other authentication mechanisms (as described in [RFC4559] and [RFC2617]) are not available.

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

1.1 Glossary

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

challenge

The following terms are specific to this document:

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

1.2 References

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

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-AUTHSO] Microsoft Corporation, "Windows Authentication Services System Overview".

[MS-FPSE] Microsoft Corporation, "FrontPage Server Extensions Remote Protocol".

[MS-WSSHP] Microsoft Corporation, "HTTP Windows SharePoint Services Headers Protocol".

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, http://www.ietf.org/rfc/rfc1945.txt

[RFC2109] Kristol, D., and Montulli, L., "HTTP State Management Mechanism", RFC 2109, February 1997, http://www.ietf.org/rfc/rfc2109.txt

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

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

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

[RFC3986] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, http://www.ietf.org/rfc/rfc3986.txt

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.txt

1.2.2 Informative References

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

[MS-OCPROTO] Microsoft Corporation, "Office Client Protocols Overview".

[MS-WEBDAVE] Microsoft Corporation, "Web Distributed Authoring and Versioning Error Extensions Protocol".

[MS-WEBSS] Microsoft Corporation, "Webs Web Service Protocol".

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

1.3 Overview

The protocol client connects to a protocol server that is gated by forms based authentication by sending messages via HTTP. The following sequence diagram illustrates one way, entailing three steps, of establishing an identity using forms based authentication between a protocol client and a protocol server.

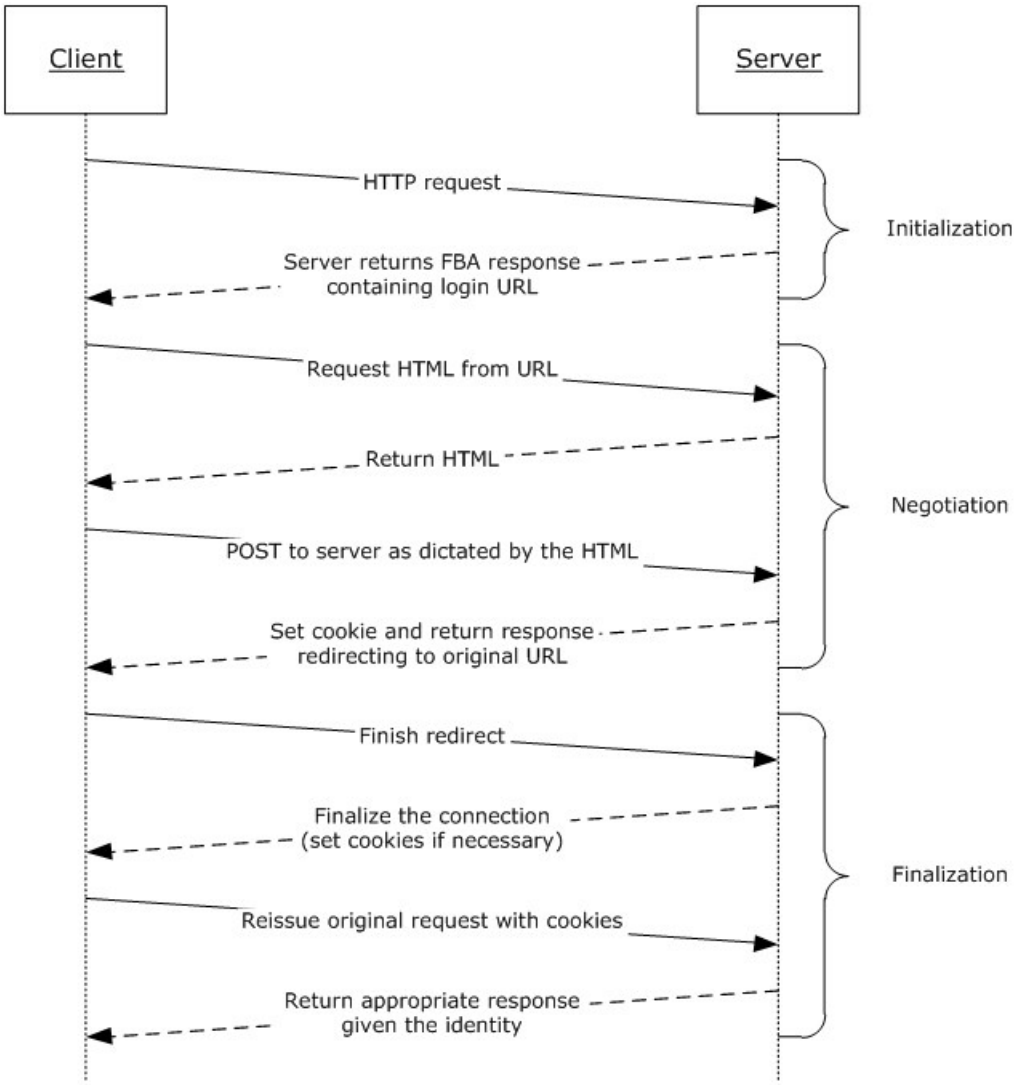

Figure 1: Sequence diagram

The three steps for establishing an identity using forms based authentication between a protocol client and a protocol server are as follows:

1. **Initialization:** The protocol client sends an initial request for any transaction between that client and the protocol server. The server responds that its authentication method is forms based authentication, as specified in section 2.2.2, including the location to which the client should navigate to authenticate. If the server response does not include this location, it is assumed to be the location to which the original request was issued. This response optionally includes the location to which the protocol server will redirect the user upon successfully authenticating that user.

2. **Negotiation:** Having determined that the protocol server is capable of establishing an identity by using forms based authentication, the protocol client renders the HTML returned from the request to the remote location provided by the server in step 1. Note that the duration of this step is neither deterministic nor specified by this protocol. The reason is that the client will continue to

follow as many redirects and refreshes as necessary to successfully establish the identity, until the server redirects either to the original URI or, if specified, the return URI provided by the server in step 1.

3. **Finalization:** After the protocol server redirects the protocol client to the return URI, the protocol client assumes that the identity has been successfully established and reissues the original request from step 1. Note that the process for actually establishing the user's identity is not specified by this protocol.

1.4 Relationship to Other Protocols

This protocol depends on HTTP, as described in [RFC2616]. To transfer the authentication state between the protocol client and protocol server, this protocol also depends on HTTP state management, as described in [RFC2109].

1.5 Prerequisites/Preconditions

Forms based authentication over HTTP assumes the following:

- The HTTP server is configured such that the user's identity is established using forms based authentication. The user's identity is transferred between the protocol client and protocol server by using HTTP state management, as described in [RFC2109].
- The protocol client is configured to store and transmit cookies, as described in [RFC2109].

1.6 Applicability Statement

Forms based authentication is used in environments where Windows Integrated Authentication methods (basic, digest, SPNEGO-based Kerberos, and NTLM HTTP Authentication), as described in [MS-AUTHSO] section 2, are not available. Additionally, the protocol client and protocol server must both support forms based authentication.

1.7 Versioning and Capability Negotiation

Versioning and capability negotation are handled by HTTP, as described in [RFC2617]. (For more information, see [RFC2616].) This protocol provides no additional versioning or capability negotiation.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Forms based authentication over HTTP messages are carried in the HTTP message headers ([RFC2616] section 4.2) and message body ([RFC2616] section 4.3).

2.2 Message Syntax

The use of forms based authentication over HTTP is indicated by the X-FORMS_BASED_AUTH_REQUIRED HTTP response header. The value of this header is a URI that points to an HTTP-based server. For more details about HTTP headers, see [RFC2616]. For more details about URIs, see [RFC3986].

2.2.1 Protocol Discovery Requests

The protocol client establishes an identity with a protocol server based on a specific **challenge** issued by that client to the server, which identifies the protocol client as a nonbrowser client application.

To be recognized as a nonbrowser client that supports this protocol, the protocol client MUST specify either a header ([RFC2616] section 4.2) or a user agent string ([RFC1945] section 10.3) in an HTTP OPTIONS request ([RFC2616] section 9.2). If the protocol client's request is not authenticated, the protocol server SHOULD<1> respond based on the criteria that appears in priority order in the following table. However, the protocol server MAY ignore the header and use only the user agent string, as specified later in this section.

Client request	Server response
The header contains a field name of "X-FORMS_BASED_AUTH_ACCEPTED" and a field value of "f".	If the protocol server supports any type of Windows Authentication, as specified in [MS-AUTHSO] section 2, the protocol server MUST NOT respond with a Forms Based Authentication Required response header (section 2.2.2) and MUST respond with a Windows Authentication challenge. If the protocol server does not support any type of Windows Authentication, it MUST respond with a Forms Based Authentication Required response header (section 2.2.2).
The header does not contain a field name of "X-FORMS_BASED_AUTH_ACCEPTED", and the user agent string contains "MS Search" followed by "Robot".	If the protocol server supports any type of Windows Authentication, as specified in [MS-AUTHSO] section 2, the protocol server MUST NOT respond with a Forms Based Authentication Required response header (section 2.2.2) and MUST respond with a Windows Authentication challenge. If the protocol server does not support any type of Windows Authentication, it MUST respond with a Forms Based Authentication Required response header (section 2.2.2).
The header contains a field name of "X-FORMS_BASED_AUTH_ACCEPTED" and a field value of "t".	The protocol server MUST respond with a Forms Based Authentication Required response header, as specified in section 2.2.2.

If the HTTP request sent by the protocol client is not authenticated, but the protocol server requires the request to be authenticated; and if the HTTP request sent by the protocol client does not include the X-FORMS_BASED_AUTH_ACCEPTED HTTP header<2>; and if the user agent string conforms to the following rules in Augmented Backus-Naur Form (ABNF), as described in [RFC5234], the protocol server MUST respond with the Forms Based Authentication Required response header, as specified in section 2.2.2.

```
"Microsoft Data Access Internet Publishing Provider"
"Microsoft-WebDAV-MiniRedir"
"Non-browser"
"MSOffice 12"
"Mozilla/4.0 (compatible; MS FrontPage "N
N = 1 - 14
```

If the request is a FrontPage Server Extensions Remote Protocol ([MS-FPSE]) request and the client has negotiated a protocol version that is greater than or equal to 12.0.0.6403 ([MS-FPSE] section 1.7.1), the protocol server MUST respond with the Forms Based Authentication Required response header, as specified in section 2.2.2.

If the request is a FrontPage Server Extensions Remote Protocol ([MS-FPSE]) request and the client has negotiated a protocol version that is less than 12.0.0.6403 ([MS-FPSE] section 1.7.1), the protocol server MUST respond with a "200 OK" HTTP status code ([RFC2616] section 10.2.1).

2.2.2 Forms Based Authentication Required Response Header

If the protocol server receives a request for an access-protected object and the request requires a Forms Based Authentication Required response as specified in section 2.2.1, the server MUST respond with a "403 Forbidden" HTTP status code ([RFC2616] section 10.4.4). Servers compliant with this protocol SHOULD<3> also return an HTTP header with a field name of X-FORMS_BASED_AUTH_REQUIRED, as specified in [MS-WSSHP] section 2.2.12. If the server returns an X-FORMS_BASED_AUTH_REQUIRED header, the value of the header MUST be a URI, as specified in [RFC3986], that specifies the protocol server login page. The protocol client MUST navigate to the login page to establish the user's identity with the protocol server.

The protocol server SHOULD<4> return an HTTP header with a field name of X-FORMS_BASED_AUTH_RETURN_URL header, as specified in [MS-WSSHP] section 2.2.13. The value of this header contains a URI, as specified in [RFC3986], that specifies the protocol server return page, which the protocol client will use to determine whether the authentication succeeded. If the URI is not present, the protocol client assumes that the URI is the same as that of the login page specified by the X-FORMS_BASED_AUTH_REQUIRED header. If the URI of the return page is a path, the path MUST contain a backward slash (/) at the end.

The server MAY return an HTTP header with a field name of X-FORMS_BASED_AUTH_DIALOG_SIZE. The value of this header MUST be formatted as a string that conforms to the following ABNF ([RFC5234]) rules:

```
size   = width "x" height
width  = 1*10(DIGIT)
height = 1*10(DIGIT)
```

The **width** element specifies the preferred width, in pixels, of the login dialog box.

The **height** element specifies the preferred height, in pixels, of the login dialog box.

If the size of the dialog box is not specified, the value "660x495" is used by the protocol client.

Both the login page and the return page MUST point to an HTTP-based server.

2.2.3 HTML Request

After the protocol client has determined that the user's identity will be established using forms based authentication, the client MUST issue an HTTP GET ([RFC2616] section 9.3) to the login page. The user agent string ([RFC1945] section 10.3) of this GET request MUST contain the following:

```
Mozilla/4.0
```

3 Protocol Details

3.1 Common Details

This protocol is used to establish a user's identity with a remote protocol server that uses an HTML form to establish that user's identity. For this reason, a model that uses existing HTTP and HTML semantics within the protocol client is useful.

3.1.1 Abstract Data Model

The protocol client relies on the remote protocol server to set the user's identity as one or more HTTP cookies. After the user's identity is established, the client then sends each cookie with each subsequent HTTP request, as specified in [RFC2109].

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

The Protocol Discovery request MUST be sent by the protocol client (for details, see section 2.2.1). The X-FORMS_BASED_AUTH_REQUIRED header and the X-FORMS_BASED_AUTH_RETURN_URL header SHOULD<5> be returned by the protocol server (for details, see section 2.2.2). Clients and servers MUST be compliant with HTTP/1.1 ([RFC2616]), HTTP Authentication ([RFC2617]), and the HTTP State Management Mechanism ([RFC2109]).

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Client Details

3.2.1 Abstract Data Model

The abstract data model follows that set forth in section 3.1.1.

3.2.2 Timers

The **ProtocolDiscoveryTimeout** timer determines how much time will elapse prior to reissuing the Protocol Discovery request. The value of this timer is not specified by this protocol, and it is up to the protocol client to choose an optimal value.

3.2.3 Initialization

This protocol is initialized when the protocol client receives the X-FORMS_BASED_AUTH_REQUIRED header from the protocol server.

3.2.4 Higher-Layer Triggered Events

The protocol client can cause additional requests to be sent to the protocol server, depending on the content of the HTML sent to the client by the server. The number of HTTP requests sent by the protocol client to the protocol server to establish the user's identity is nondeterministic and not specified by this protocol.

3.2.5 Message Processing Events and Sequencing Rules

No additional message processing events and sequencing rules exist beyond those detailed in section 3.1.5.

3.2.6 Timer Events

The **ProtocolDiscoveryTimeout** event causes the protocol client to send the Protocol Discovery request, as specified in section 2.2.1, prior to sending any additional HTTP request.

3.2.7 Other Local Events

When the remote protocol server issues a redirect to the return page, the protocol client completes that request and then allows any other pending HTTP requests, which MUST contain cookies as specified in [RFC2109], to continue.

3.3 Server Details

3.3.1 Abstract Data Model

The abstract data model follows that specified in section 3.1.1.

3.3.2 Timers

None.

3.3.3 Initialization

This protocol is initialized when the protocol server receives the Protocol Discovery request from the protocol client.

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

No additional message processing events and sequencing rules exist beyond those specified in section 3.1.5.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

4 Protocol Examples

This scenario shows the message exchanges that occur when a protocol client requests an access-protected document at the URI https://www.contoso.com/dir/document.docx from a protocol server that is gated by forms based authentication.

Prior to requesting this document, the client attempt to determine the capabilities of the server:

```
C: OPTIONS /dir/
C: User-Agent: MSOffice 12
```

The server issues a response indicating that it is capable of forms based authentication:

```
S: HTTP/1.1 403 Forbidden
S: X-FORMS_BASED_AUTH_REQUIRED:
https://www.contoso.com/fbalogin.aspx?wreply=https://www.contoso.com/OnSuccess.aspx
S: X-FORMS_BASED_AUTH_RETURN_URL: https://www.contoso.com/OnSuccess.aspx
S: X-FORMS_BASED_AUTH_DIALOG_SIZE: 800x600.
```

The client then issues an HTTP request to the header specified in the X-FORMS_BASED_AUTH_REQUIRED URI, requesting HTML that the user can employ to establish his or her identity:

```
C: GET /fbalogin.aspx?wreply=https://www.contoso.com/OnSuccess.aspx
C: User-Agent: Mozilla/4.0
```

The server then replies with an HTML form that contains enough logic to establish the user's identity with the server. In this example, the server returns a simple form:

```
S: HTTP/1.1 200 OK
S: <body>
S: <form name="CredentialForm" method="post"
S: action="fbalogin.aspx?wreply=https://www.contoso.com/OnSuccess.aspx"
S: id="Creds">
S: <table>
S:         <tr><td>Login: </td></tr>
S:         <tr><td>Username: <input name="UsernameTextBox" type="text"
S:         id="UsernameTextBox" </td></tr>
S:         <tr><td>Password: <input name="PasswordTextBox"
S:         type="password"
S:         id="PasswordTextBox" /></td></tr>
S:         <tr><td><input type="submit" name="UsernamePasswordButton"
S:         value="Submit" id="UsernamePasswordButton" /></td></tr>
S: </table>
S: </form>
S: </body>
```

On receipt of the preceding HTML, the client instantiates a dialog box of the size that is specified in the initial response to the OPTIONS request. (In this example, that size is 800x600). After the HTML is rendered, the rich client follows the instructions dictated by the HTML form. This example assumes that the user entered the credentials "user:pass" for the user name and password, and then clicked the submit button.

```
C: POST /fbalogin.aspx?wreply=https://www.contoso.com/OnSuccess.aspx
C: User-Agent: Mozilla/4.0
C: UsernameTextBox=user&PasswordTextBox=pass
```

If the user's interactions with the HTML form allowed the server to establish the user's identity, the remote protocol server sets the identity as a cookie on the request and redirects the user back to the **return_url** specified in the response to the Protocol Discovery request.

```
S: HTTP/1.1 302 Object Moved
Location: https://www.contoso.com/OnSuccess.aspx
Set-Cookie: Authentication=<server-determined hash of the user's identity>
```

On seeing the redirect, the client determines that this URI matches that returned in response to the Protocol Discovery request. Because the URIs match, the client assumes success, follows the redirect, and closes the form that it was using to render the HTML

```
C: GET /OnSuccess.aspx
C: User-Agent: Mozilla/4.0
C: Cookie: Authentication=<server-determined hash of the user's identity>
```

The server can then respond with any finalization logic that is required:

```
S: HTTP/1.1 200 OK
S: Set-Cookie: FooCookie=bar
```

After this call completes, the client runs the series of HTTP transactions that is required to successfully open https://www.contoso.com/dir/document.docx. For more information about this series of transactions, see [MS-OCPROTO] section 2.1.2.1.2.

5 Security

5.1 Security Considerations for Implementers

Forms based authentication necessarily transmits the user's identity as plain text. Implementers are encouraged to use a secure channel, such as HTTPS, to avoid inadvertently exposing the user's identity.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

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

- The 2007 Microsoft Office system
- Microsoft Office 2010 suites
- Microsoft Office 2013
- Microsoft SharePoint Foundation 2010
- Microsoft SharePoint Foundation 2013
- Windows SharePoint Services 3.0

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

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

<1> Section 2.2.1: SharePoint 2007 Products and Technologies ignores the header and processes the request based solely on the user agent string.

<2> Section 2.2.1: The Office 2010 and the 2007 Office system client applications never send the X-FORMS_BASED_AUTH_ACCEPTED header and always rely on the user agent string to identify themselves as nonbrowser clients to a protocol server.

<3> Section 2.2.2: SharePoint 2007 Products and Technologies does not return the X-FORMS_BASED_AUTH_REQUIRED header . Rather, the protocol server returns the following extended error, as described in [MS-WEBDAVE] section 2.2.3:

```
X-MSDAVEXT_Error: 917656;
Access%20denied%2e%20%20Before%20opening%20files%20in%20this%20location%2c%20you%20must%20fir
st%20browse%20to%20the%20web%20site%20and%20select%20the%20option%20to%20login%20automaticall
y%2e
```

<4> Section 2.2.2: SharePoint 2007 Products and Technologies does not return the X-FORMS_BASED_AUTH_RETURN_URL header . Rather, the protocol server returns the following extended error, as described in [MS-WEBDAVE] section 2.2.3:

```
X-MSDAVEXT_Error: 917656;
Access%20denied%2e%20%20Before%20opening%20files%20in%20this%20location%2c%20you%20must%20fir
st%20browse%20to%20the%20web%20site%20and%20select%20the%20option%20to%20login%20automaticall
y%2e
```

<5> Section 3.1.5: SharePoint 2007 Products and Technologies does not explicitly return the X-FORMS_BASED_AUTH_REQUIRED header to the Protocol Discovery request, as detailed in section

2.2.2, that is made by a protocol client. Rather, the server returns the following extended error, as described in [MS-WEBDAVE] section 2.2.3:

X-MSDAVEXT_Error: 917656;
Access%20denied%2e%20%20Before%20opening%20files%20in%20this%20location%2c%20you%20must%20first%20browse%20to%20the%20web%20site%20and%20select%20the%20option%20to%20login%20automatically%2e

Upon receipt of this error, the protocol client issues a request to determine the web URL for the specified URL, as described in [MS-WEBSS].

Upon determination of the web URL, the client needs to consider the following URL to be the equivalent of the value for X-FORMS_BASED_AUTH_REQUIRED, as defined in section 2.2.2:

http://*server*/*weburl*/_layouts/Authenticate.aspx?Source=Error.aspx

where:

The *server* placeholder represents the address of the SharePoint 2007 Products and Technologies.

The *weburl* placeholder represents the value returned from the previous **UrlToWebUrl** request.

"/_layouts/Authenticate.aspx?Source=Error.aspx" is a hard-coded string.

Additionally, because the server returns the client to the Error.aspx page on successful authentication, the client considers the following URL to be equivalent to the value of X-FORMS_BASED_AUTH_RETURN_URL, as defined in section 2.2.2:

http://*server*/*weburl*/_layouts/Error.aspx

where:

The *server* placeholder is the address of the SharePoint 2007 Products and Technologies.

The *weburl* placeholder represents the value returned from the previous **UrlToWebUrl** request.

"/_layouts/Error.aspx" is a hard-coded string.

7 Change Tracking

This section identifies changes that were made to the [MS-OFBA] protocol document between the July 2013 and November 2013 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

- A document revision that incorporates changes to interoperability requirements or functionality.
- 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
2.2.1 Protocol Discovery Requests	Clarified the name of the ''FrontPage Server Extensions Remote Protocol" request.	N	Content updated.

8 Index

T

V