[MS-NTHT]:
NTLM Over HTTP Protocol Specification

Revision Summary

Date	Revision History	Revision Class	Comments
03/14/2007	1.0		Version 1.0 release
04/10/2007	1.1		Version 1.1 release
05/18/2007	1.2		Version 1.2 release
06/08/2007	1.2.1	Editorial	Revised and edited the technical content.
07/10/2007	1.2.2	Editorial	Revised and edited the technical content.

Date	Revision History	Revision Class	Comments
08/17/2007	1.2.3	Editorial	Revised and edited the technical content.
09/21/2007	1.2.4	Editorial	Revised and edited the technical content.
10/26/2007	2.0	Major	Converted document to unified format and updated the technical content.
01/25/2008	2.0.1	Editorial	Revised and edited the technical content.

Table of Contents

1 Introduction

Microsoft provides support for NT LAN Manager (NTLM) (as specified in [MS-NLMP]) authentication in Microsoft Internet Explorer and Microsoft Internet Information Services (IIS) that uses the HTTP Protocol (for more information, see [RFC2616]) in addition to other standard authentication mechanisms. This provides the benefits of the NTLM Authentication Protocol for Web applications when other authentication mechanisms (such as those specified in [RFC4559] and [RFC2617]) are not available.

Support for NTLM authentication is as specified in [RFC4559], using native NTLM Authentication Protocol (as specified in [MS-NLMP]) data units instead of encoded tokens (as specified in [RFC4178]). The tokens are still transmitted using base64 encoding. This document calls out the differences in the Microsoft implementation from what is specified in [RFC4559], where applicable.

1.1 Glossary

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

The following terms are specific to this document:

Backus-Naur Form (BNF): Used to describe grammars. Also referred to as "Augmented **BNF**" in [RFC2616] section 2.1.

Client: Used as described in [RFC2616] section 1.3.

Proxy: Used as described in [RFC2616] section 1.3.

Server: Used as described in [RFC2616] section 1.3.

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

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-DTYP] Microsoft Corporation, "Windows Data Types", March 2007.

[MS-NLMP] Microsoft Corporation, "NT LAN Manager (NTLM) Authentication Protocol Specification", July 2006.

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

[RFC2616] Fielding, R., 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., Lawrence, S., Leach, P., Luotonen, A., and Stewart, L., "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999, http://www.ietf.org/rfc/rfc2617.txt

[RFC4178] Zhu, L., Leach, P., Jaganathan, K., and Ingersoll, W., "The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism", RFC 4178, October 2005, http://www.ietf.org/rfc/rfc4178.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.ietf.org/rfc/rfc4559.txt

1.2.2 Informative References

There are no informative references for this protocol.

1.3 Protocol Overview (Synopsis)

The NTLM over HTTP Protocol authentication variant is similar to the SPNEGO HTTP (as specified in [RFC4559]) authentication mechanism. Both are used to authenticate a Web client to a Web server. Although SPNEGO HTTP (as specified in [RFC4559]) works with both Kerberos and NTLM authentication, the NTLM over HTTP Protocol authentication variant only works with NTLM. The Kerberos protocol is not supported.

1.4 Relationship to Other Protocols

This document is a companion to the SPNEGO HTTP authentication document, as specified in [RFC4559]. It uses the augmented **BNF**, as specified in [RFC4559] section 4, and relies on both the non-terminals defined in that document and other aspects of the specification HTTP/1.1, as specified in [RFC2617]. For more information, see [RFC2616].

1.5 Prerequisites/Preconditions

NTLM over HTTP Protocol authentication assumes the following in addition to any assumptions specified in [MS-NLMP].

1. The Web server is operating in an environment with a database of user identities, and the NT LAN Manager (NTLM) Authentication Protocol, as specified in [MS-NLMP], is available to authenticate those users.

2. The Web client has implemented the NT LAN Manager (NTLM) Authentication Protocol, as specified in [MS-NLMP], so that it can participate in user authentication to the Web server.

1.6 Applicability Statement

NTLM HTTP authentication is used in environments where SPNEGO-based Kerberos and NTLM HTTP authentication, as specified in [RFC4559], is not available, and the Web client and server support NTLM authentication, as specified in [MS-NLMP].

1.7 Versioning and Capability Negotiation

Versioning and capability negotiation is handled by the HTTP protocols specified in [RFC2617] (for more information, see [RFC2616]). This protocol has no additional versioning or capability negotiation.

1.8 Vendor-Extensible Fields

NTLM over HTTP Protocol authentication contains no vendor-extensible fields.

1.9 Standards Assignments

Parameter	Value	Reference
HTTP auth-scheme	NTLM	[RFC2617] section 1.2

2 Messages

The following sections specify how NTLM over HTTP Protocol messages are transported and message syntax.

2.1 Transport

NTLM over HTTP Protocol messages are carried in the HTTP authentication exchanges as authentication data (auth-data), as specified in [RFC4559] sections 4.1 and 4.2.

2.2 Message Syntax

The use of NTLM over HTTP Protocol authentication is indicated by an HTTP authentication scheme (auth-scheme) NTLM. The authentication parameters (auth-params) that are exchanged are base64-encoded messages. For more details about auth-scheme and auth-params, see [RFC2617] section 1.2.

2.2.1 WWW-Authenticate Response Header

If the server receives a request for an access-protected object and an acceptable Authorization header has not been sent, the server MUST respond with a "401 Unauthorized" status code and a WWW-Authenticate header, per the framework in [RFC2616]. The initial WWW-Authenticate header MUST NOT carry any auth-data. For more details about the text in this section, see [RFC2616], and specifically for the 401 status code, see [RFC2616] section 10.4.2.

The NTLM scheme operates as follows:

```
challenge= "NTLM" auth-data
auth-data = 1#( [ntlm-data] )
```

The meaning of the directive values that are used above is as follows:

```
ntlm-data
```

This directive contains the base64 encoding of a CHALLENGE_MESSAGE, as specified in [MS-NLMP] section 2.2.1.2.

2.2.2 Authorization Request Header

Upon receipt of the response containing a WWW-Authenticate header from the server, the client is expected to retry the HTTP request with the Authorization header, per the framework in [RFC2616]:

```
credentials= "NTLM" auth-data2

auth-data2= 1#( [ntlm-data] )
```

The meaning of the directives values that are used above is as follows:

```
ntlm-data
```

This directive contains the base64 encoding of either an AUTHENTICATE_MESSAGE, as specified in [MS-NLMP] section 2.2.1.3, or a NEGOTIATE_MESSAGE, as specified in [MS-NLMP] section 2.2.1.1.

Any return code other than a client error HTTP 401 status code (for more information, see [RFC2616] section 10.4.2), represents successful authentication. If the client is not able to access the requested resource and the response status code is not HTTP 401, the problem is HTTP protocol-specific (for more information, see [RFC2616] section 10).

2.2.3 Proxy-Authenticate Response Header

If the client must authenticate itself to a proxy and an acceptable Proxy-Authorization header has not been sent, the proxy MUST respond with a "407 Proxy Authentication Required" status code (for more information, see [RFC2616] section 10.4.8) and a "Proxy-Authenticate" header, per the framework in [RFC2616]. The initial Proxy-Authenticate header MUST NOT carry any auth-data.

The NTLM scheme operates as follows:

```
challenge= "NTLM" auth-data3

auth-data3= 1#( [ntlm-data] )
```

The meaning of the directives values that are used above is as follows:

```
ntlm-data
```

This directive contains the base64 encoding of a CHALLENGE_MESSAGE, as specified in [MS-NLMP] section 2.2.1.1.

2.2.4 Proxy-Authorization Request Header

Upon receipt of the response containing a Proxy-Authenticate header from the proxy, the client is expected to retry the HTTP request with the Proxy-Authorization header, per the framework in [RFC2616]:

```
credentials= "NTLM" auth-data4

auth-data4= 1#( [ntlm-data] )
```

The meaning of the values of the directives that are used above is as follows:

```
ntlm-data
```

This directive contains the base64 encoding of either an AUTHENTICATE_MESSAGE, as specified in [MS-NLMP] section 2.2.1.3, or a NEGOTIATE_MESSAGE, as specified in [MS-NLMP] section 2.2.1.1.

Any return code other than a client error HTTP 407 status code ([RFC2616] section 10.4.2), represents successful authentication. If the client is not able to access the requested resource and the response status code is not HTTP 407, the reason is HTTP protocol-specific. For details, see [RFC2616] section 10.

3 Protocol Details

The following sections specify details of the NTLM Over HTTP Protocol, including an abstract data model and message processing rules.

3.1 Common Details

3.1.1 Abstract Data Model

The abstract data model follows the specifications in [RFC4559].

3.1.2 Timers

None.

3.1.3 Initialization

There is no protocol-specific initialization.

3.1.4 Higher-Layer Triggered Events

There are no higher-layer triggered events in common to all parts of this protocol.

3.1.5 Message Processing Events and Sequencing Rules

The WWW-Authenticate header is only sent from the server. The Authorization header is only sent by the client. (For details, see [RFC2617] and also see [RFC2616] sections 14.47 and 14.8.) Clients, servers, and proxys MUST be compliant with [RFC2617] and [RFC2616].

The Proxy-Authenticate header is only sent from the proxy. The Proxy-Authorization header is only sent by the client. (For more information, see [RFC2617] and [RFC2616] sections 14.33 and 14.34.) Clients, servers, and proxys MUST be compliant with [RFC2617] and [RFC2616].

3.1.6 Timer Events

None.

3.1.7 Other Local Events

There are no local events other than those specified in [RFC4559].

3.2 Server Details

3.2.1 Abstract Data Model

The abstract data model follows the specifications in [RFC4559].

3.2.2 Timers

None.

3.2.3 Initialization

There is no protocol-specific initialization.

3.2.4 Higher-Layer Triggered Events

There are no higher-layer triggered events in common to all parts of this protocol.

3.2.5 Message Processing Events and Sequencing Rules

The WWW-Authenticate header is only sent from the server. The Authorization header is only sent by the client. (For more information, see [RFC2617] and [RFC2616] sections 14.7 and 14.8.) Servers MUST be compliant with [RFC2617] and [RFC2616].

The Proxy-Authenticate header is only sent from the proxy. The Proxy-Authorization header is only sent by the client. (For more information, see [RFC2617] and [RFC2616] sections 14.33 and 14.34.) Servers MUST be compliant with [RFC2617] and [RFC2616].

3.2.6 Timer Events

None.

3.2.7 Other Local Events

There are no local events other than those specified in [RFC4559].

3.3 Client Details

3.3.1 Abstract Data Model

The abstract data model follows the specifications in [RFC4559].

3.3.2 Timers

None.

3.3.3 Initialization

There is no protocol-specific initialization.

3.3.4 Higher-Layer Triggered Events

There are no higher-layer triggered events associated with the client side of the protocol.

3.3.5 Message Processing Events and Sequencing Rules

The WWW-Authenticate header is only sent from the server. The Authorization header is only sent by the client. (For more information, see [RFC2617] and [RFC2616] sections 14.7 and 14.8.) Servers MUST be compliant with [RFC2617] and [RFC2616].

The Proxy-Authenticate header is only sent from the proxy. The Proxy-Authorization header is only sent by the client. (For more information, see [RFC2617] and [RFC2616] sections 14.33 and 14.34.) Servers MUST be compliant with [RFC2617] and [RFC2616].

3.3.6 Timer Events

None.

3.3.7 Other Local Events

There are no local events other than those specified in [RFC4559].

3.4 Proxy Details

3.4.1 Abstract Data Model

The abstract data model follows the specifications in [RFC4559].

3.4.2 Timers

None.

3.4.3 Initialization

There is no protocol-specific initialization.

3.4.4 Higher-Layer Triggered Events

There are no higher-layer triggered events associated with the proxy component of the protocol.

3.4.5 Message Processing Events and Sequencing Rules

The WWW-Authenticate header is only sent from the server. The Authorization header is only sent by the client. (For more information, see [RFC2617] and [RFC2616] sections 14.7 and 14.8.) Servers MUST be compliant with [RFC2617] and [RFC2616].

The Proxy-Authenticate header is only sent from the proxy. The Proxy-Authorization header is only sent by the client. (For more information, see [RFC2617] and [RFC2616] sections 14.33 and 14.34.) Servers MUST be compliant with [RFC2617] and [RFC2616].

3.4.6 Timer Events

None.

3.4.7 Other Local Events

There are no local events other than those specified in [RFC4559].

4 Protocol Examples

4.1 Server Examples

This scenario shows the messages exchanged when a Web client requests an access-protected document from a Web server using a GET method request at the URL: http://www.nowhere.org/dir/index.html.

```
C: GET dir/index.html
```

The first time the client requests the document, no Authorization header is sent; so the server responds with the following:

```
S: HTTP/1.1 401 Unauthorized
S: WWW-Authenticate: NTLM
```

The client obtains the local user credentials by using the [MS-NLMP] security package and then generates a new GET request to the server. The request contains an Authorization header with an NTLM NEGOTIATE_MESSAGE (as specified in [MS-NLMP] section 2.2.1.1) in ntlm-data:

```
C: GET dir/index.html
C: Authorization: NTLM tESsBmE/yNY3lb6a0L6vVQEZNqwQn0s8Unew
```

The server decodes the ntlm-data that is contained in the auth-data2 base64-encoded data and passes this to its implementation of [MS-NLMP]. If the server accepts this authentication data from the client, it responds with an HTTP 401 code (for more information, see [RFC2616] section 10.2) and a WWW-Authenticate header with an NTLM CHALLENGE_MESSAGE (as specified in [MS-NLMP] section 2.2.1.2) in ntlm-data:

```
S: HTTP/1.1 401 Unauthorized
S: WWW-Authenticate: NTLM yNY3lb6a0L6vVQEZNqwQn0s8UNew33KdKZvG+Onv
```

The client decodes the ntlm-data that is contained in the auth-data base64-encoded data and passes this to its implementation of [MS-NLMP]. If this authentication data is valid, the client responds by reissuing the GET request with an Authorization header that contains an NTLM AUTHENTICATE_MESSAGE (as specified in [MS-NLMP] section 2.2.1.3) in ntlm-data:

```
C: GET dir/index.html
C: Authorization: NTLM kGaXHz6/owHcWRlvGFk8ReUZKHo=QEZNqwQn0s8U
```

The server decodes the ntlm-data that is contained in the auth-data2 base64-encoded data and passes this to its implementation of [MS-NLMP]. If the server accepts this authentication data from the client, it responds with an HTTP 2xx code (for more information, see [RFC2616] section 10.2) indicating success. The requested content is also included in the server response.

Note The base64 values used above are for illustrative purposes only and do not represent valid base64-encoded NTLM messages.

4.2 Proxy Examples

This scenario shows the messages that are exchanged when a Web client requests an access-protected document from a proxy using a GET method request at the URL: http://www.nowhere.org/dir/index.html.

```
C: GET dir/index.html
```

The first time the client requests the document, no Proxy-Authorization header is sent; so the proxy responds with the following:

```
S: HTTP/1.1 407 Proxy Authentication Required
S: Proxy-Authenticate: NTLM
```

The client obtains the local user credentials using the [MS-NLMP] security package and then generates a new GET request to the proxy. The request contains a Proxy-Authorization header with an NTLM NEGOTIATE_MESSAGE (as specified in [MS-NLMP] section 2.2.1.1) in ntlm-data:

```
C: GET dir/index.html
C: Proxy-Authorization: NTLM tESsBmE/yNY3lb6a0L6vVQEZNqwQn0s8Unew
```

The proxy decodes the ntlm-data that is contained in the auth-data2 base64-encoded data and passes this to its implementation of [MS-NLMP]. If the proxy accepts this authentication data from the client, it responds with an HTTP 407 code (for more information, see [RFC2616] section 10.2) and a Proxy-Authenticate header with an NTLM CHALLENGE_MESSAGE (as specified in [MS-NLMP] section 2.2.1.2) in ntlm-data:

```
S: HTTP/1.1 407 Proxy Authentication Required
S: Proxy-Authenticate: NTLM yNY3lb6a0L6vVQEZNqwQn0s8UNew33KdKZvG+Onv
```

The client decodes the ntlm-data that is contained in the auth-data base64-encoded data and passes this to its implementation of [MS-NLMP]. If this authentication data is valid, the client responds by reissuing the GET request with a Proxy-Authorization header that contains an NTLM AUTHENTICATE_MESSAGE (as specified in [MS-NLMP] section 2.2.1.3) in ntlm-data:

```
C: GET dir/index.html
C: Proxy-Authorization: NTLM kGaXHz6/owHcWRlvGFk8ReUZKHo=QEZNqwQn0s8U
```

The proxy decodes the ntlm-data that is contained in the auth-data2 base64-encoded data and passes this to its implementation of [MS-NLMP]. If the proxy accepts this authentication data from the client, it responds with an HTTP 2xx code (for more information, see [RFC2616] section 10.2) indicating success. The requested content is also included in the proxy response.

Note The base64 values used above are for illustrative purposes only and do not represent valid base64-encoded NTLM messages.

5 Security

The following sections specify security considerations for implementers of the NTLM over HTTP Protocol.

5.1 Security Considerations for Implementers

The NTLM Authentication Protocol (see [MS-NLMP]) does not provide any facilities for mutual authentication; therefore, server identities cannot be verified. Other security considerations are as specified in [RFC4559] section 6.

5.2 Index of Security Parameters

There are no security parameters in the NTLM over HTTP Protocol.

6 Appendix A: Windows Behavior

The information in this specification is applicable to the following versions of Windows:

- Windows Vista
- Windows Server 2003
- Windows XP
- Windows 2000
- Windows NT

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Windows behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that Windows does not follow the prescription.

There are no Windows behavior exceptions in this document.

7 Index