[MS-OXPOP3]:

Post Office Protocol Version 3 (POP3) Extensions

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **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 can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs**. To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are 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 as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications documentation does 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 documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Support. For questions and support, please contact dochelp@microsoft.com.

Preliminary Documentation. This particular Open Specifications document provides documentation for past and current releases and/or for the pre-release version of this technology. This document provides final documentation for past and current releases and preliminary documentation, as applicable and specifically noted in this document, for the pre-release version. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. Because this documentation might change between the pre-release version and the final

version of this technology, there are risks in relying on this 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	New	Initial Availability.
6/27/2008	1.0	Major	Initial Release.
8/6/2008	1.01	Minor	Edited technical content.
9/3/2008	1.02	Minor	Revised and edited technical content.
12/3/2008	1.03	Minor	Minor editorial fixes.
3/4/2009	1.04	Minor	Revised and edited technical content.
4/10/2009	2.0	Major	Updated technical content and applicable product releases.
7/15/2009	3.0	Major	Revised and edited for technical content.
11/4/2009	3.1.0	Minor	Updated the technical content.
2/10/2010	3.2.0	Minor	Updated the technical content.
5/5/2010	3.2.1	Editorial	Revised and edited the technical content.
8/4/2010	4.0	Major	Significantly changed the technical content.
11/3/2010	5.0	Major	Significantly changed the technical content.
3/18/2011	6.0	Major	Significantly changed the technical content.
8/5/2011	6.1	Minor	Clarified the meaning of the technical content.
10/7/2011	6.1	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	7.0	Major	Significantly changed the technical content.
4/27/2012	7.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	7.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	8.0	Major	Significantly changed the technical content.
2/11/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.
7/26/2013	9.0	Major	Significantly changed the technical content.
11/18/2013	9.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
10/30/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.
3/16/2015	10.0	Major	Significantly changed the technical content.
5/26/2015	11.0	Major	Significantly changed the technical content.
9/14/2015	11.0	None	No changes to the meaning, language, or formatting of the technical content.
6/13/2016	11.0	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2016	11.0	None	No changes to the meaning, language, or formatting of the technical content.
7/24/2018	12.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Post Office Protocol Version 3 (POP3) extensions specify extensions to the Post Office Protocol Version 3 (POP3). The following extensions are specified:

- The NT LAN Manager (NTLM) authentication mechanism for the POP3 protocol. This is a proprietary extension that is used with the POP3 **AUTH** command.
- The delegate access mechanism for the POP3 protocol.

For the purposes of this document, the NTLM authentication mechanism for POP3 is referred to in subsequent sections as "the NTLM POP3 Extension", and the delegate access mechanism for POP3 is referred to as "the POP3 Delegate Access extension".

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

alias: An alternate name that can be used to reference an object or element.

ASCII: The American Standard Code for Information Interchange (ASCII) is an 8-bit character-encoding scheme based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text. ASCII refers to a single 8-bit ASCII character or an array of 8-bit ASCII characters with the high bit of each character set to zero.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [RFC5234].

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

best body: The text format that provides the richest representation of a message body. The algorithm for determining the best-body format is described in [MS-OXBBODY].

connection-oriented NTLM: A particular variant of **NTLM** designed to be used with connection-oriented remote procedure call (RPC), as described in [MS-NLMP].

delegate: A user or resource that has permissions to act on behalf of another user or resource.

delegator: A user or resource for which another user or resource has permission to act on its behalf.

domain: A set of users and computers sharing a common namespace and management infrastructure. At least one computer member of the set must act as a domain controller (DC) and host a member list that identifies all members of the domain, as well as optionally hosting the Active Directory service. The domain controller provides authentication of members, creating a unit of trust for its members. Each domain has an identifier that is shared among its members. For more information, see [MS-AUTHSOD] section 1.1.1.5 and [MS-ADTS].

mailbox: A message store that contains email, calendar items, and other Message objects for a single recipient.

Multipurpose Internet Mail Extensions (MIME): A set of extensions that redefines and expands support for various types of content in email messages, as described in [RFC2045], [RFC2046], and [RFC2047].

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

NTLM message: A message that carries authentication information. Its payload data is passed to the application that supports embedded NTLM authentication by the NTLM software installed on the local computer. NTLM messages are transmitted between the client and server embedded within the application protocol that is using NTLM authentication. There are three types of NTLM messages: NTLM NEGOTIATE_MESSAGE, NTLM CHALLENGE_MESSAGE, and NTLM AUTHENTICATE_MESSAGE.

NTLM software: Software that implements the **NT LAN Manager (NTLM) Authentication Protocol**.

plain text: Text that does not have markup. See also plain text message body.

POP3 response: A message sent by a POP3 server in response to a message from a POP3 client. The structure of this message, as specified in [RFC1939], is as follows: <+OK> <response text><CR><LF> or <-ERR> <response text><CR><LF>.

user principal name (UPN): A user account name (sometimes referred to as the user logon name) and a domain name that identifies the domain in which the user account is located. This is the standard usage for logging on to a Windows domain. The format is: someone@example.com (in the form of an email address). In Active Directory, the userPrincipalName attribute of the account object, as described in [MS-ADTS].

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.

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

[MS-OXBBODY] Microsoft Corporation, "Best Body Retrieval Algorithm".

[RFC1734] Myers, J., "POP3 AUTHentication Command", RFC 1734, December 1994, http://www.rfc-editor.org/rfc/rfc1734.txt

[RFC1939] Myers, J., and Rose, M., "Post Office Protocol - Version 3", STD 53, RFC 1939, May 1996, http://www.rfc-editor.org/rfc/rfc1939.txt

[RFC2045] Freed, N., and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, http://www.rfc-editor.org/rfc/rfc2045.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

[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

[RFC822] Crocker, D.H., "Standard for ARPA Internet Text Messages", STD 11, RFC 822, August 1982, http://www.ietf.org/rfc/rfc0822.txt

1.2.2 Informative References

None.

1.3 Overview

Client applications that connect to the Post Office Protocol - Version 3 (POP3) service can use either standard **plain text** password authentication, as described in [RFC1939], or NT LAN Manager (**NTLM**) authentication.<1>

The NTLM POP3 Extension specifies how a POP3 client and POP3 server can use the NT LAN Manager (NTLM) Authentication Protocol, as described in [MS-NLMP], so that the POP3 server can authenticate the POP3 client. NTLM is a challenge/response authentication protocol that depends on the application layer protocols to transport NTLM packets from client to server, and from server to client.

This specification defines how the POP3 **AUTH** command, as described in [RFC1734], is used to perform authentication by using the NTLM Authentication protocol. The POP3 **AUTH** command standard defines an extensibility mechanism for arbitrary authentication protocols to be plugged in to the core protocol.

This specification defines an embedded protocol in which NTLM authentication data is first transformed into a **base64 encoding** representation, and then formatted by padding with POP3 keywords as defined by the AUTH mechanism. The base64 encoding and the formatting are very rudimentary, and solely intended to make the NTLM data fit the framework described in [RFC1734]. The following figure shows the sequence of transformations that are performed on an **NTLM message** to produce a message that can be sent over POP3.

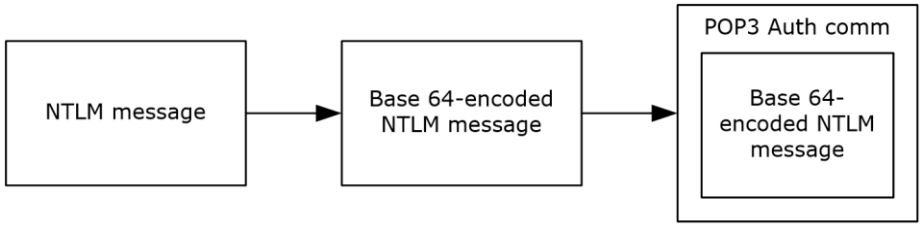

Figure 1: Relationship between NTLM message and POP3: NTLM Authentication Protocol message

This document specifies a pass-through protocol that does not specify the structure of NTLM information. Instead, the protocol relies on the software that implements the NTLM Authentication Protocol (as described in [MS-NLMP]) to process each NTLM message that is to be sent or received.

This specification defines a client role and a server role.

When POP3 performs an NTLM authentication, it has to interact with the NTLM subsystem appropriately. The following is an overview of this interaction.

If acting as a POP3 client:

1. The NTLM subsystem returns the first NTLM message to the client, to be sent to the server.
2. The client applies the base64 encoding and POP3-padding transformations mentioned earlier and described in detail later in this document to produce a POP3 message and send this message to the server.
3. The client waits for a response from the server. When the response is received, the client checks to determine whether the response indicates the end of authentication (success or failure) or that authentication is continuing.
4. If the authentication is continuing, the response message is stripped of the POP3 padding, base64 decoded, and passed into the NTLM subsystem, at which point the NTLM subsystem might return another NTLM message that has to be sent to the server. Steps 2 through 4 are repeated until authentication either succeeds or fails.

If acting as a POP3 server:

1. The server waits to receive the first POP3 authentication message from the client.
2. When a POP3 message is received from the client, the POP3 padding is removed, the message is base64 decoded, and the resulting NTLM message is passed into the NTLM subsystem.
3. The NTLM subsystem returns a status that indicates whether authentication completed successfully or failed, or whether more NTLM messages have to be exchanged to complete the authentication.
4. If the authentication is continuing, the NTLM subsystem returns an NTLM message that has to be sent to the client. This message is base64 encoded, and the POP3 padding is applied and sent to the client. Steps 2 through 4 are repeated until authentication either succeeds or fails.

The sequence that follows shows the typical flow of packets between client and server after NTLM authentication has been selected:

1. The POP3 client sends an NTLM **NEGOTIATE_MESSAGE** message (as described in [MS-NLMP]) embedded in a POP3 packet to the server.
2. On receiving the POP3 packet with an NTLM **NEGOTIATE_MESSAGE** message, the POP3 server sends an NTLM **CHALLENGE_MESSAGE** message (as described in [MS-NLMP]) embedded in a POP3 packet to the client.
3. In response, the POP3 client sends an NTLM **AUTHENTICATE_MESSAGE** message (as described in [MS-NLMP]) embedded in a POP3 packet.
4. The server then sends a **POP3 response** to the client to complete the authentication process successfully.

The **NTLM NEGOTIATE_MESSAGE**, **NTLM CHALLENGE_MESSAGE**, and **NTLM AUTHENTICATE_MESSAGE** message packets contain NTLM authentication data that has to be processed by the **NTLM software** that is installed on the local computer. The manner in which NTLM messages are to be retrieved and processed is described in [MS-NLMP].

This specification defines the delegate access mechanism that is used by a POP3 client.

Implementers of this specification have to conform to POP3, as described in [RFC1734] and [RFC1939], the **MIME** base64 encoding method, as described in [RFC2045], and the NTLM Authentication Protocol, as described in [MS-NLMP].

1.4 Relationship to Other Protocols

The NTLM POP3 Extension uses the POP3 AUTH extension mechanism, as specified in [RFC1734], and is an embedded protocol. Unlike stand-alone application protocols, such as Telnet or HTTP, packets for this specification are embedded in POP3 commands and server responses.

POP3 specifies only the sequence in which a POP3 server and POP3 client are required to exchange NTLM messages to authenticate the client to the server successfully. It does not specify how the client obtains NTLM messages from the local **NTLM software**, or how the POP3 server processes NTLM messages. The POP3 client and POP3 server implementations depend on the availability of an implementation of the NTLM Authentication Protocol (as specified in [MS-NLMP]) to obtain and process NTLM messages and on the availability of the **base64 encoding** and decoding mechanisms (as specified in [RFC2045]) to encode and decode the **NTLM message**s embedded in POP3 packets.

For conceptual background information and overviews of the relationships and interactions between this and other protocols, see [MS-OXPROTO].

1.5 Prerequisites/Preconditions

Because POP3 depends on NTLM to authenticate the client to the server, both server and client have access to an implementation of the NTLM Authentication Protocol (as specified in [MS-NLMP]) that is capable of supporting **connection-oriented NTLM**.

1.6 Applicability Statement

The NTLM POP3 Extension is used only when implementing a POP3 client that has to authenticate to a POP3 server by using NTLM authentication.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- Security and Authentication methods: The NTLM POP3 Extension supports the NTLMv1 and NTLMv2 authentication methods, as specified in [MS-NLMP].
- Capability Negotiation: POP3 does not support negotiation of which version of the NTLM authentication protocol to use. Instead, the NTLM authentication protocol version is configured on both the client and the server before authentication. NTLM authentication protocol version mismatches are handled by the NTLM Authentication Protocol implementation, not by POP3.

The client discovers whether the server supports NTLM AUTH through the **AUTH** command, issued without any arguments, at which point the server responds with a list of supported authentication mechanisms followed by a line that contains only a period (.). If NTLM is supported, the server includes the word "NTLM" in the list.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The NTLM POP3 Extension does not establish transport connections. Instead, NTLM POP3 Extension messages are encapsulated in POP3 commands and responses. The way in which NTLM POP3 Extension messages are encapsulated in POP3 commands is specified in section 2.2.

2.2 Message Syntax

The NTLM POP3 Extension messages are divided into the following two categories:

1. AUTH extensions
2. Delegate Access

2.2.1 AUTH Extensions

The first category of POP3 messages is messages that fall within the AUTH extensibility framework. These messages are specified in [RFC1734] and [RFC1939]. Some messages have parameters that have to be customized by the extensibility mechanism (such as NTLM). The following customizations are introduced in this specification:

- [RFC1734] section 2 defines the syntax of the **AUTH** command to initiate authentication. The parameter "mechanism" is defined to be the string "NTLM" for the NTLM POP3 Extension. The command to initiate an NTLM conversation by a client in **Augmented Backus-Naur Form (ABNF)**, as specified in [RFC5234], is specified as follows. This is referred to as the **POP3_AUTH_NTLM_Initiation_Command** in this specification.

```
"AUTH NTLM" CRLF
```

- If NTLM is supported, the POP3 server responds with a POP3 message to indicate that NTLM is supported, which is specified in [RFC1939]. The syntax of this command in ABNF form is specified as follows. This is referred to as the **POP3_NTLM_Supported_Response** command in this specification.

```
+ SP CRLF
```

- If NTLM is not supported, the POP3 server returns a failure status code as defined by [RFC1734] and [RFC1939]. The only data in this message that is useful is the "-ERR" string. The remaining data is human-readable data and has no bearing on the authentication. The syntax of this command in ABNF form is specified as follows. This is referred to as the **POP3_AUTH_NTLM_Fail_Response** command in this specification.

```
-ERR SP <human_readable_string> CRLF
```

- At every point of time during the authentication exchange, the client MUST parse the responses in the messages sent by the server and interpret them as defined by [RFC1734]. The responses define various states such as success in authenticating, failure to authenticate, and any other arbitrary failures that the software can encounter.

The client can send or receive any of the following messages during authentication (note that the syntax and meaning of all these messages are specified in [RFC1734]):

- **POP3_AUTH_NTLM_Blob_Response**. This message is partially defined in [RFC1734]. The '+' status code indicates ongoing authentication and also indicates that the <Base 64-encoded-NTLM-message> is to be processed by the authentication subsystem. In this case, the client MUST de-encapsulate the data and pass it to the NTLM subsystem.

```
+ SP <Base 64-encoded-NTLM-message> CRLF
```

- **POP3_AUTH_NTLM_Fail_Response**. This message is defined in [RFC1939] and indicates that the authentication has terminated unsuccessfully, either because the user name or password was incorrect or because of some other arbitrary error, such as a software or data corruption error.

```
-ERR SP <human-readable-string> CRLF
```

- **POP3_AUTH_NTLM_Succeeded_Response**. This message is defined in [RFC1939] and indicates that the authentication negotiation has completed with the client successfully authenticating to the server.

```
+OK SP <human-readable-string> CRLF
```

- **POP3_AUTH_NTLM_Cancelled_Response**. This message is defined in [RFC1939] and indicates that the authentication negotiation has been canceled with the client.

```
-ERR SP <human-readable-string> CRLF
```

- **NTLM message**s encapsulated by the client and sent to the server are referred to as **POP3_AUTH_NTLM_Blob_Command** in this specification. They have the following syntax defined in ABNF, and they conform to the prescription as specified in [RFC1734].

```
<Base 64-encoded-NTLM-message> CRLF
```

- The client is able to cancel the authentication request by issuing a **POP3_AUTH_NTLM_Cancellation_Command**. This has the following syntax defined in ABNF:

```
"*" CRLF
```

2.2.2 POP3 Delegate Access

The POP3 Delegate Access extension extends the **USER** command, as specified in [RFC1939] section 7. The POP3 Delegate Access extension provides part of the logon mechanism for **delegate**s to access a **delegator**'s **mailbox**. There are four formats for using delegate access with POP3. In every case, the part after the last "/" of the user string is the mailbox identity in either **alias** or **user principal name (UPN)** format. The four formats are as follows:

- "USER" SP domain/delegateuseralias/principalalias
- "USER" SP domain/delegateuseralias/principalupn
- "USER" SP delegateuserupn/principalalias
- "USER" SP delegateuserupn/principalupn

The **domain** part of the **USER** string represents the delegate's domain.

The "delegateuserupn" part of the **USER** string represents the UPN of the delegate, which is composed of the user's identifier and domain, as specified in [RFC822] section 6.1.

The "delegateuseralias" part of the **USER** string represents the e-mail alias of the delegate.

The "principaluserupn" part of the **USER** string represents the UPN of the delegator, which is composed of the delegator's identifier and domain, as specified in [RFC822] section 6.1.

The "principaluseralias" part of the **USER** string represents the e-mail alias of the delegator.

3 Protocol Details

3.1 POP3 Client Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

3.1.1.1 Client POP3 State Model

The following figure shows the client POP3 state model.

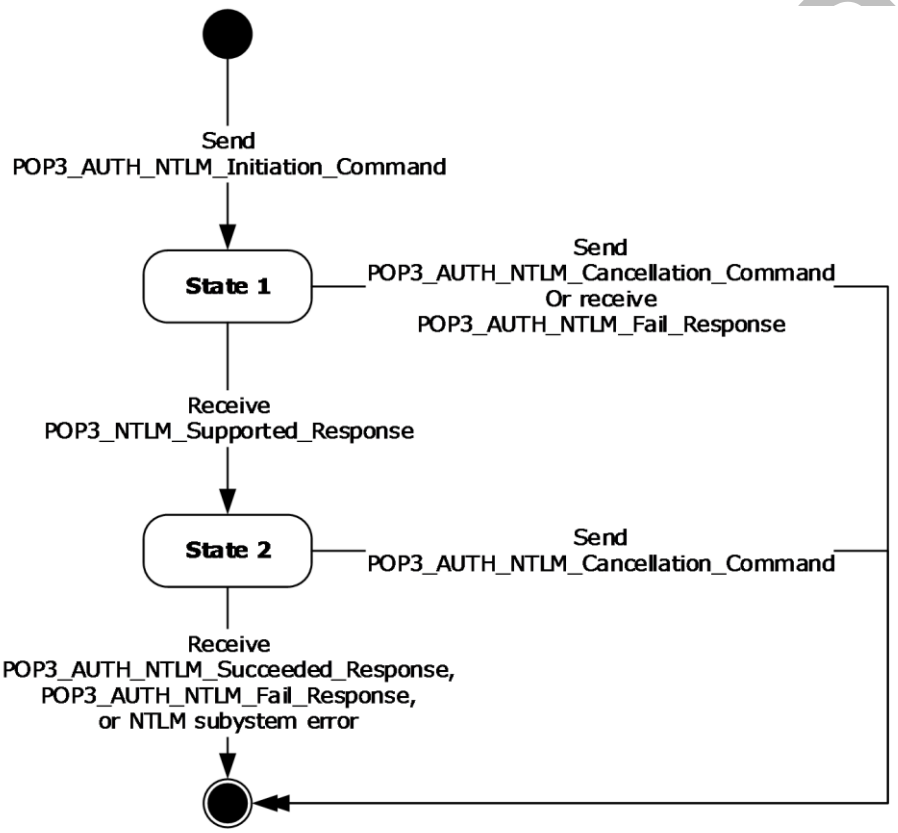

Figure 2: Client POP3 state model

The abstract data model for the NTLM POP3 Extension includes the following states:

1. Start

   This is the state of the client before the **POP3_AUTH_NTLM_Initiation_Command** has been sent.

2. State 1: **sent_authentication_request**

This is the state of the client after the **POP3_AUTH_NTLM_Initiation_Command** has been sent.

3. State 2: **inside_authentication**

   This is the state entered by a client after it has received a **POP3_NTLM_Supported_Response** message. In this state, the client initializes the NTLM subsystem by performing the following steps:

   - Encapsulates the **NTLM message**, returned by the NTLM subsystem, into a **POP3_AUTH_NTLM_Blob_Command** message and sends the challenge message to the server. Waits for a response from the server.
   - De-encapsulates the received **POP3_AUTH_NTLM_Blob_Response** message data (if any) from the server and converts it to NTLM messages data.
   - Passes the NTLM message data to the NTLM subsystem.
   - Encapsulates the NTLM authenticate message, returned by the NTLM subsystem, into a **POP3_AUTH_NTLM_Blob_Command** message.
   - Sends the **POP3_AUTH_NTLM_Blob_Command** message to the server.

   This state terminates when:

   - A **POP3_AUTH_NTLM_Succeeded_Response** message or **POP3_AUTH_NTLM_Fail_Response** message is received.
   - Any failure is reported by the NTLM subsystem.

4. Stop: **completed_authentication**

   This is the state of the client on exiting the **inside_authentication** state or the **sent_authentication_request** state. The rules for how the **inside_authentication** state is exited are defined in section 3.1.5. The behavior of POP3 in this state is not in the scope of this specification. It represents the end state of the authentication protocol.

3.1.1.2 NTLM Subsystem Interaction

During the **inside_authentication** phase, the POP3 client invokes the NTLM subsystem and uses **connection-oriented NTLM**, as specified in [MS-NLMP].

The following is a description of how POP3 uses NTLM. All NTLM messages are encapsulated as specified in section 2.2. [MS-NLMP] describes the data model, internal states, and sequencing of NTLM messages in greater detail, as follows:

1. The client initiates the authentication by invoking NTLM, after which NTLM returns the NTLM **NEGOTIATE_MESSAGE** message (as specified in [MS-NLMP]) to be sent to the server.
2. Subsequently, the exchange of **NTLM messages** goes on as defined by the NTLM protocol, with the POP3 client encapsulating the NTLM messages before sending them to the server, and de-encapsulating POP3 messages to obtain the NTLM message before giving it to NTLM.
3. The NTLM protocol completes authentication, either successfully or unsuccessfully, as follows:

   - The server sends the **POP3_AUTH_NTLM_Succeeded_Response** message to the client. On receiving this message, the client transitions to the **completed_authentication** state and treats the authentication attempt as successful.

- The server sends the **POP3_AUTH_NTLM_Fail_Response** message to the client. On receiving this message, the client transitions to the **completed_authentication** state and treats the authentication attempt as failed.
- Failures reported from the NTLM package (which can occur for any reason, including incorrect data being passed in, or implementation-specific errors) are reported to the client by NTLM and cause the client to transition to the **completed_authentication** state.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

When the client cancels authentication, it sends a **POP3_AUTH_NTLM_Cancellation_Command** message to the server, as specified in section 2.2.1.

3.1.5 Message Processing Events and Sequencing Rules

The NTLM POP3 Extension is driven by a series of message exchanges between a POP3 server and a POP3 client. The rules that govern the sequencing of commands and the internal states of the client and server are defined by a combination of [RFC1734] and [MS-NLMP]. Section 3.1.1 defines how the rules specified in [RFC1734] and [MS-NLMP] govern POP3 authentication.

If the client receives a message that is not expected for its current state, the client MUST cancel the authentication process and transition to the **completed_authentication** state.

3.1.5.1 Receiving a POP3_NTLM_Supported_Response Message

The expected state is **sent_authentication_request**.

On receiving this message, a client MUST generate the first **NTLM message** by calling the NTLM subsystem. The NTLM subsystem then generates the **NTLM NEGOTIATE_MESSAGE** message, as specified in [MS-NLMP]. The NTLM message is then encapsulated as defined previously and sent to the server.

The state of the client is changed to **inside_authentication**.

3.1.5.2 Receiving a POP3_AUTH_NTLM_Fail_Response Message

The expected state is **sent_authentication_request** or **inside_authentication**.

On receiving this message, a client MUST end the NTLM authentication attempt and change the state to **complete_authentication**. The client can then take any action it considers appropriate; this extension does not mandate any specific course of action.

3.1.5.3 Sending a POP3_AUTH_NTLM_Blob_Command Message

The expected state is **inside_authentication**.

This section defines the processing of **POP3_AUTH_NTLM_Blob_Command** messages. These **NTLM messages** sent by the client are encapsulated as follows to conform to the AUTH mechanism:

1. Base64 encode the NTLM message data. This is needed because NTLM messages contain data outside the **ASCII** character range, whereas POP3 only supports ASCII characters.

2. Send the base64 encoded string.

3. Suffix the <CR> and <LF> characters (ASCII values 0x0D and 0x0A), as required by POP3.

The **ABNF** definition of a client message is as follows:

<Base 64-encoded-NTLM-message><CR><LF>

De-encapsulation of these messages by the client adheres to the reverse logic, as follows:

1. Remove the <CR> and <LF> characters (ASCII values 0x0D and 0x0A).

2. Base64 decode the POP3 data to produce the original NTLM message data.

3.1.5.4 Receiving a POP3_AUTH_NTLM_Blob_Response Message

The expected state is **inside_authentication**.

On receiving this message, a client MUST de-encapsulate it to obtain the embedded NTLM message, and then pass it to the NTLM subsystem for processing. The NTLM subsystem can then either report an error, or report success and return an **NTLM message** to be sent to the server.

3.1.5.4.1 Error from NTLM

If the NTLM subsystem reports an error, the client MUST change its internal state to **completed_authentication** and conclude that the authentication has failed. The client can then take any action it considers appropriate; this specification does not mandate any specific course of action.

Typical actions in this scenario are to try other (non-authentication-related) POP3 commands or to disconnect the connection.

3.1.5.4.2 NTLM Reports Success and Returns an NTLM Message

The **NTLM message** SHOULD be encapsulated and sent to the server. No change occurs in the state of the client.

3.1.5.5 Receiving a POP3_AUTH_NTLM_Succeeded_Response Message

Expected state: **inside_authentication**.

The POP3 client MUST change its internal state to **completed_authentication** and conclude that the authentication has succeeded. The client can then take any action it considers appropriate. This specification does not mandate any specific course of action.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 POP3 Server Details

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

3.2.1.1 Server POP3 State Model

The following figure shows the server POP3 state model.

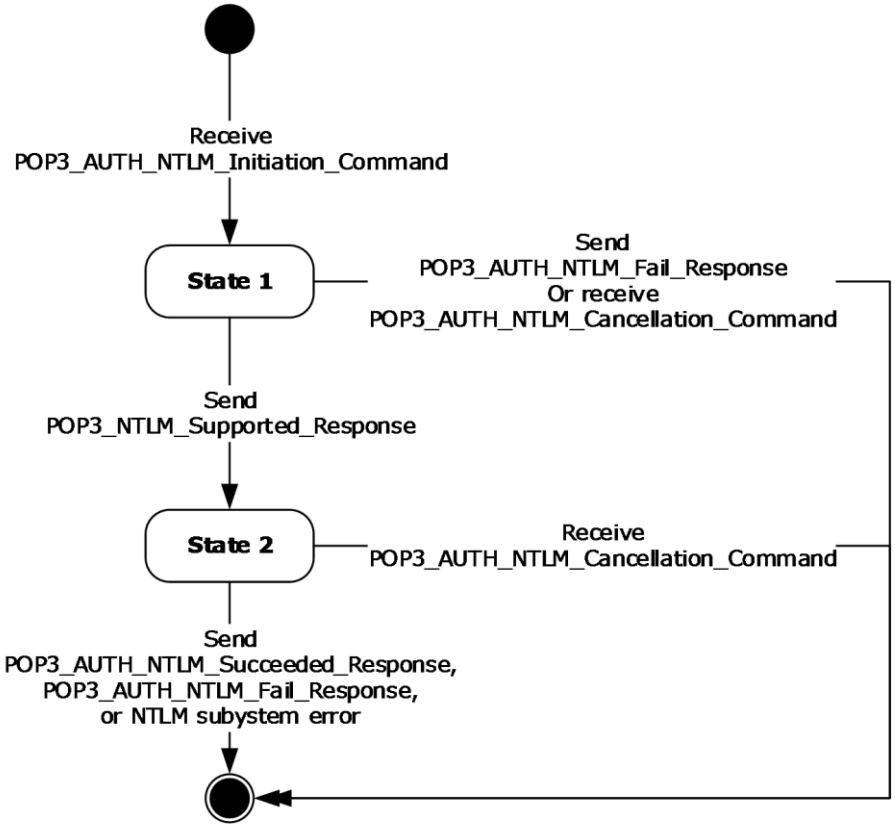

Figure 3: Server POP3 state model

The abstract data model for NTLM POP3 Extension includes the following states:

1. Start

   This is the state of the server before the **POP3_AUTH_NTLM_Initiation_Command** command has been received.

2. State 1: **received_authentication_request**

   This is the state of the server after the **POP3_AUTH_NTLM_Initiation_Command** command has been received.

3. State 2: **inside_authentication**

   This is the state entered by a server after it has sent a **POP3_NTLM_Supported_Response** message. In this state, the server initializes the NTLM subsystem by performing the following steps:

   - Waits for a message from the client.
   - De-encapsulates the received **POP3_AUTH_NTLM_Blob_Command** message data from the client and obtains the embedded **NTLM message** data.
   - Passes the NTLM message data to the NTLM subsystem.
   - Encapsulates the NTLM message returned by the NTLM subsystem into a **POP3_AUTH_NTLM_Blob_Response** message.
   - Sends the **POP3_AUTH_NTLM_Blob_Response** message to the client.

   This state terminates when:

   - The NTLM subsystem reports completion with either a success or failed authentication status, at which point it sends the client the **POP3_AUTH_NTLM_Succeeded_Response** message or the **POP3_AUTH_NTLM_Fail_Response** message, as specified in [RFC1734].
   - The client sends a **POP3_AUTH_NTLM_Cancellation_Command** message to the server, at which point the server sends a **POP3_AUTH_NTLM_Fail_Response** message to the client.
   - Any failure is reported by the NTLM subsystem.

4. Stop: **completed_authentication**

   This is the state of the server on exiting the **inside_authentication** state or the **received_authentication_request** state. The rules for how the **inside_authentication** state is exited are defined in section 3.2.5. The behavior of POP3 in this state is specified in [RFC1734]; it represents the end state of the authentication protocol.

3.2.1.2 NTLM Subsystem Interaction

During the **inside_authentication** state, the POP3 server invokes the NTLM subsystem and uses **connection-oriented NTLM**, as specified in [MS-NLMP].

The following is a description of how POP3 uses NTLM. For more details, see [MS-NLMP], which describes the data model and sequencing of NTLM packets in greater detail.

1. On receiving the NTLM **NEGOTIATE_MESSAGE** message (as specified in [MS-NLMP]), the server passes it to the NTLM subsystem and is returned the NTLM **CHALLENGE_MESSAGE** message (as specified in [MS-NLMP]), if the NTLM **NEGOTIATE_MESSAGE** message was valid.
2. Subsequently, the exchange of **NTLM messages** goes on as defined by the NTLM protocol, with the POP3 server encapsulating the NTLM messages that are returned by NTLM before sending them to the client.
3. When the NTLM protocol completes authentication, either successfully or unsuccessfully, the NTLM subsystem notifies POP3 and the following occurs:

   - On successful completion, the server exits the **inside_authentication** state, enters the **completed_authentication** state, and send the **POP3_AUTH_NTLM_Succeeded_Response** message to the client. On receiving this message, the client transitions to the **completed_authentication** state.

- If a failure occurs because of an incorrect password error, as described in [MS-NLMP], the server enters the **completed_authentication** state and sends the client a **POP3_AUTH_NTLM_Fail_Response** message.
- If a failure occurs on the server because of any reason other than the incorrect password error, the server enters the **completed_authentication** state and send the client a **POP3_AUTH_NTLM_Fail_Response** message. On receiving this message, the client enters the **completed_authentication** state.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

When the server receives a **POP3_AUTH_NTLM_Blob_Command** command that contains an **NTLM NEGOTIATE_MESSAGE** message (as specified in [MS-NLMP]), the message is passed to the NTLM system. If the NTLM system is successful in handling the message, the NTLM system returns an **NTLM CHALLENGE_MESSAGE** message (as specified in [MS-NLMP]). The NTLM system triggers the server to send a **POP3_AUTH_NTLM_Blob_Response** message that contains the NTLM **CHALLENGE_MESSAGE** message.

When the server receives a **POP3_AUTH_NTLM_Blob_Command** command that contains an **NTLM AUTHENTICATE_MESSAGE** message (as specified in [MS-NLMP]), the message is passed to the NTLM system. If the NTLM system is successful in handling the message, the NTLM system returns a confirmation that the client successfully logged on. The successful NTLM system logon triggers the server to send a **POP3_Authentication_Succeeded_Response** message. The server state is then changed to the **completed_authentication** state.

When the server receives a **POP3_AUTH_NTLM_ Blob_Command** command that contains an **NTLM AUTHENTICATE_MESSAGE** message, the message is passed to the NTLM system. If the NTLM system handles the **NTLM AUTHENTICATE_MESSAGE** message and the message has an incorrect user name or password, the NTLM system MUST terminate authentication. The NTLM system informs the server that authentication has been stopped, which triggers the server to send a **POP3_AUTH_Failed_Response** message to the client. The server state is then changed to the **completed_authentication** state.

If the NTLM system returns any failure status, the failure status MUST trigger the server to send a **POP3_AUTH_Failed_Response** message to the client.

3.2.5 Message Processing Events and Sequencing Rules

The NTLM POP3 Extension is driven by a series of message exchanges between a POP3 server and a POP3 client. The rules that govern the sequencing of commands and the internal states of the client and server are defined by a combination of [RFC1734] and [MS-NLMP]. Section 3.2.1 defines how the rules specified in [RFC1734] and [MS-NLMP] govern POP3 authentication.

If the server receives a message that is not expected for its current state, the server MUST cancel the authentication process and transition to the completed state.

3.2.5.1 Receiving a POP3_AUTH_NTLM_Initiation_Command Message

The expected state is **start**.

On receiving the **POP3_AUTH_NTLM_Initiation_Command** message, the server changes its state to **received_authentication_request**. If the server supports NTLM, it MUST reply with the **POP3_NTLM_Supported_Response** message and change its state to **inside_authentication**.

If the server does not support NTLM, it MUST respond with the **POP3_AUTH_NTLM_Fail_Response** message, and the internal state is changed to **completed_authentication**.

3.2.5.2 Receiving a POP3_AUTH_NTLM_Blob_Command Message

The expected state is **inside_authentication**.

On receiving this message, a server MUST de-encapsulate the message, obtain the embedded NTLM message, and pass it to the NTLM subsystem. The NTLM subsystem can:

1. Report success in processing the message and return an NTLM message to continue authentication.
2. Report that authentication completed successfully.
3. Report that authentication failed because of a bad user name or password, as specified in [MS-NLMP].
4. Report that the authentication failed because of some other software error or message corruption.

3.2.5.2.1 NTLM Returns Success, Returning an NTLM Message

The **NTLM message** MUST be encapsulated and sent to the client. The NTLM message sent to the client is the **POP3_AUTH_NTLM_Blob_Response** message that is specified in section 3.2.5.3. The internal state of the POP3 server remains unchanged.

3.2.5.2.2 NTLM Returns Success, Indicating Authentication Completed Successfully

The server MUST return the **POP3_AUTH_NTLM_Succeeded_Response** message and change its internal state to **completed_authentication**.

3.2.5.2.3 NTLM Returns Status, Indicating User Name or Password Was Incorrect

The server MUST return the **POP3_AUTH_NTLM_Fail_Response** message and change its internal state to **completed_authentication**.

3.2.5.2.4 NTLM Returns a Failure Status, Indicating Any Other Error

The server MUST return the **POP3_AUTH_NTLM_Fail_Response** message and change its internal state to **completed_authentication**.

3.2.5.3 Sending a POP3_AUTH_NTLM_Blob_Response Message

The expected state is **inside_authentication**.

This section defines the creation of **POP3_AUTH_NTLM_Blob_Response** messages. These are NTLM messages sent by the server, and they MUST be encapsulated as follows to conform to syntax specified by the AUTH mechanism:

1. Base-64-encode the **NTLM message** data.<2>
2. To the base64 encoded string, prefix the **POP3 response** code with a plus sign (+).
3. Suffix the <CR> and <LF> characters (**ASCII** values 0x0D and 0x0A) as required by POP3.

The **ABNF** definition of a server message is as follows:

+<SP><Base 64-encoded-NTLM-message><CR><LF>

De-encapsulation of these messages by the client adheres to the reverse logic, as follows:

1. Remove the <CR> and <LF> characters (ASCII values 0x0D and 0x0A).
2. Remove the POP3 response code (+) and the space following it.

Decode the base64 encoded POP3 data to produce the original NTLM message data.<3>

3.2.5.4 Receiving a POP3_AUTH_NTLM_Cancellation_Command Message

The expected state is **received_authentication_request** or **inside_authentication**.

On receiving the **POP3_AUTH_NTLM_Cancellation_Command** message, the server state is changed to **completed_authentication**. The server sends a **POP3_AUTH_NTLM_Failed_Response** message to the client.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

The following subsections describe operations used in a common scenario to illustrate the function of the Post Office Protocol Version 3 (POP3) Extensions.

4.1 POP3 Client Successfully Authenticating to a POP3 Server

This section illustrates the NTLM POP3 Extension with a scenario in which a POP3 client successfully authenticates to a POP3 server by using NTLM. The following figure shows a POP3 client authenticating to a POP3 server.

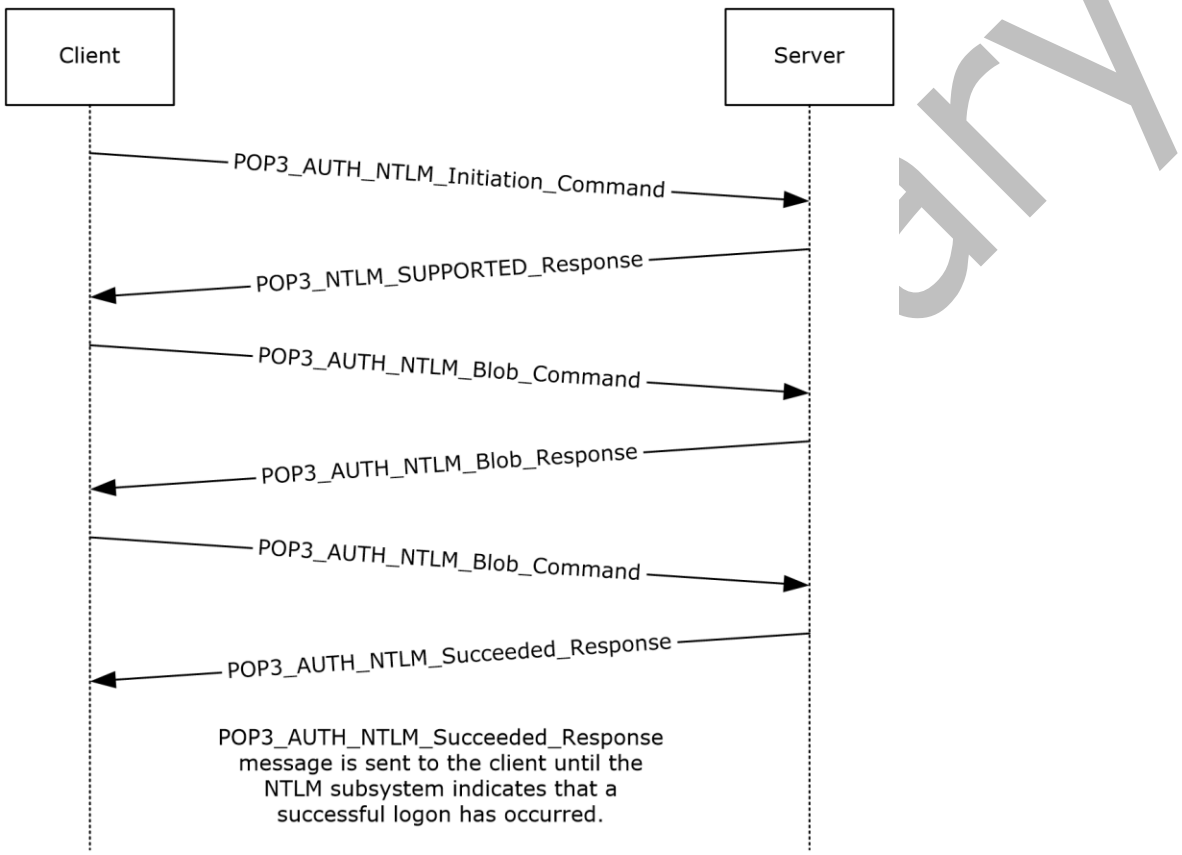

Figure 4: POP3 client successfully authenticating to POP3 server

1. The client sends a **POP3_AUTH_NTLM_Initiation_Command** command to the server. This command is described in [RFC1734] and does not carry any POP3-specific data. It is included in this example to provide a better understanding of the POP3 NTLM initiation command. The POP3 message is as follows:

   ```
   AUTH NTLM
   ```

2. The server sends the **POP3_NTLM_Supported_Response** message, which indicates that it can perform NTLM authentication. The POP3 message is as follows:

   ```
   +
   ```

3. The client sends a **POP3_AUTH_NTLM_Blob_Command** command that contains a base64 encoded NTLM **NEGOTIATE_MESSAGE** message (as described in [MS-NLMP]).

   The POP3 message is as follows:

   TlRMTVNTUAABAAAAB4IIogAAAAAAAAAAAAAAAAAAAAAFASgKAAAADw==The original NTLM message is as follows:

```
00000000:4e 54 4c 4d 53 53 50 00 01 00 00 00 07 82 08 a2     NTLMSSP......,.¢
00000010:00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     ................
00000020:05 01 28 0a 00 00 00 0f ..(.....
```

4. The server sends **a POP3_AUTH_NTLM_Blob_Response** message that contains a base64 encoded NTLM **CHALLENGE_MESSAGE** message (as described in [MS-NLMP]).

   The POP3 message is as follows:

```
+ TlRMTVNTUAACAAAAFAAUADgAAAAFgoqinziKqGYjdlEAAAAAAAAAAAGQAZABMAAAABQ
LODgAAAA9UAEUAUwBUAFMARQBSAFYARQBSAAIAFABUAEUAUwBUAFMARQBSAFYARQBSAA
EAFABUAEUAUwBUAFMARQBSAFYARQBSAAQAFABUAGUAcwB0AFMAZQByAHYAZQByAAMAFA
BUAGUAcwB0AFMAZQByAHYAZQByAAAAAAA=
```

   The NTLM message is as follows:

```
00000000:4e 54 4c 4d 53 53 50 00 02 00 00 00 14 00 14 00     NTLMSSP.........
00000010:38 00 00 00 05 82 8a a2 9f 38 8a a8 66 23 76 51     8....,Š¢Ÿ8Š¨f#vQ
00000020:00 00 00 00 00 00 00 00 64 00 64 00 4c 00 00 00     ........d.d.L...
00000030:05 02 ce 0e 00 00 00 0f 54 00 45 00 53 00 54 00     ..Î.....T.E.S.T.
00000040:53 00 45 00 52 00 56 00 45 00 52 00 02 00 14 00     S.E.R.V.E.R.....
00000050:54 00 45 00 53 00 54 00 53 00 45 00 52 00 56 00     T.E.S.T.S.E.R.V.
00000060:45 00 52 00 01 00 14 00 54 00 45 00 53 00 54 00     E.R.....T.E.S.T.
00000070:53 00 45 00 52 00 56 00 45 00 52 00 04 00 14 00     S.E.R.V.E.R.....
00000080:54 00 65 00 73 00 74 00 53 00 65 00 72 00 76 00     T.e.s.t.S.e.r.v.
00000090:65 00 72 00 03 00 14 00 54 00 65 00 73 00 74 00     e.r.....T.e.s.t.
000000a0:53 00 65 00 72 00 76 00 65 00 72 00 00 00 00 00     S.e.r.v.e.r.....
```

5. The client sends a **POP3_AUTH_NTLM_Blob_Command** message that contains a base64 encoded NTLM AUTHENTICATE_MESSAGE message (as described in [MS-NLMP]).

   The POP3 message is as follows:

```
TlRMTVNTUAADAAAAGAAYAGIAAAAYABgAegAAAAAAAABIAAAACAAIAEgAAAASABIAUAAA
AAAAAACSAAAABYKIogUBKAoAAAAPdQBzAGUAcgBOAEYALQBDAEwASQBFAE4AVABKMiQ4
djhcSgAAAAAAAAAAAAAAAAAAAAC7zUSgB0Auy98bRi6h3mwHMJfbKNtxmmo=
```

   The NTLM message is as follows:

```
00000000:4e 54 4c 4d 53 53 50 00 03 00 00 00 18 00 18 00     NTLMSSP….......
00000010:62 00 00 00 18 00 18 00 7a 00 00 00 00 00 00 00     b…….z……..
00000020:48 00 00 00 08 00 08 00 48 00 00 00 12 00 12 00     H……H……..
00000030:50 00 00 00 00 00 00 00 92 00 00 00 05 82 88 a2     P……'….,ˆ¢
00000040:05 01 28 0a 00 00 00 0f 75 00 73 00 65 00 72 00     ..(…..u.s.e.r.
00000050:4e 00 46 00 2d 00 43 00 4c 00 49 00 45 00 4e 00     N.F.-.C.L.I.E.N.
00000060:54 00 4a 32 24 38 76 38 5c 4a 00 00 00 00 00 00     T.J2$8v8\J……..
00000070:00 00 00 00 00 00 00 00 00 00 bb cd 44 a0 07 40     ……..…»ÍD .@
00000080:2e cb df 1b 46 2e a1 de 6c 07 30 97 db 28 db 71     .Ëß.F.¡Þl.0—Û(Ûq
00000090:9a 6a šj
```

6. The server sends a **POP3_AUTH_NTLM_Succeeded_Response** message. The POP3 message is as follow:

```
+OK User successfully logged on
```

4.2 POP3 Client Unsuccessfully Authenticating to a POP3 Server

This section illustrates the NTLM POP3 Extension with a scenario in which a POP3 client tries NTLM authentication to a POP3 server and the authentication fails. The following figure shows the unsuccessful attempt to authenticate to the POP3 server.

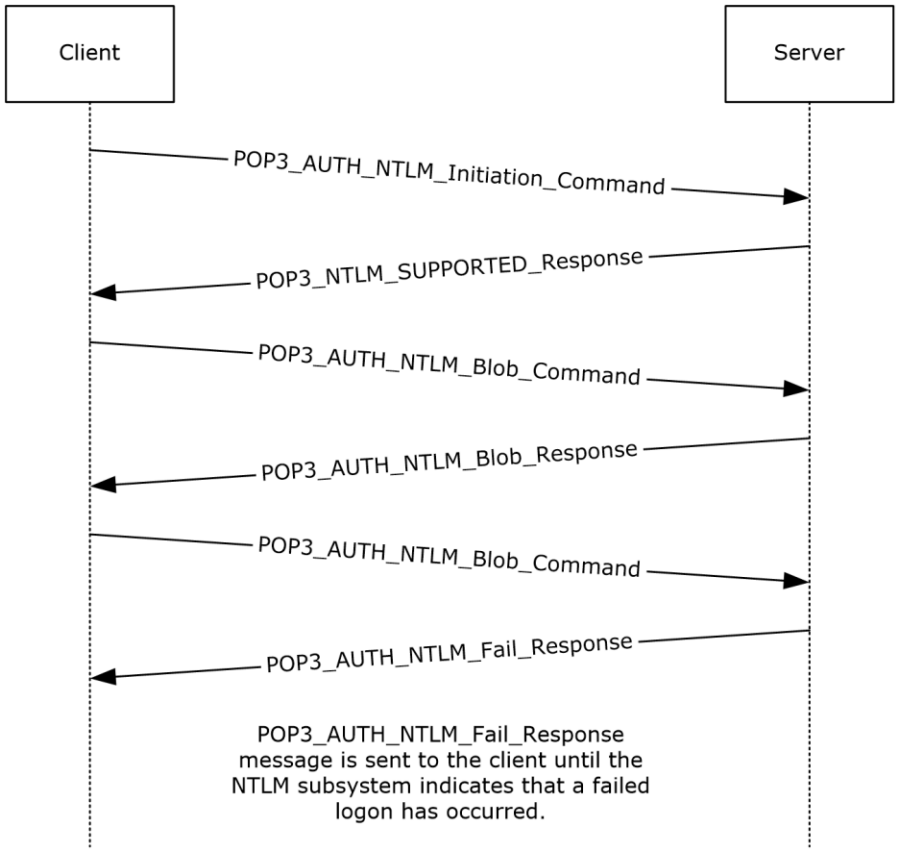

Figure 5: Client unsuccessfully authenticating to POP3 server

1. The client sends a **POP3_AUTH_NTLM_Initiation_Command** command to the server. This command is described in [RFC1734] and does not carry any POP3-specific data. It is included in this example to provide a better understanding of the POP3 NTLM initiation command. The POP3 message is as follows:

```
AUTH NTLM
```

2. The server sends the **POP3_NTLM_Supported_Response** message, which indicates that it can perform NTLM authentication. The POP3 message is as follows:

```
+
```

3. The client sends a **POP3_AUTH_NTLM_Blob_Command** command that contains a **base64 encoding** NTLM **NEGOTIATE_MESSAGE** message (as described in [MS-NLMP]).

   The POP3 message is as follows:

```
TlRMTVNTUAABAAAAB4IIogAAAAAAAAAAAAAAAAAAAAFASgKAAAADw==
```

   The NTLM message is as follows:

```
00000000:4e 54 4c 4d 53 53 50 00 01 00 00 00 07 82 08 a2     NTLMSSP......,.¢
00000010:00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     ................
00000020:05 01 28 0a 00 00 00 0f ..(.....
```

4. The server sends a **POP3_AUTH_NTLM_Blob_Response** message that contains a base64 encoded NTLM **CHALLENGE_MESSAGE** message (as described in [MS-NLMP]).

   The POP3 message is as follows:

```
+ TlRMTVNTUAACAAAAFAAUADgAAAAFgoqieUWd5ES4Bi0AAAAAAAAAAGQAZABMAA
AABQLODgAAAA9UAEUAUwBUAFMARQBSAFYARQBSAAIAFABUAEUAUwBUAFMARQBSAF
YARQBSAAEAFABUAEUAUwBUAFMARQBSAFYARQBSAAQAFABUAGUAcwB0AFMAZQByAH
YAZQByAAMAFABUAGUAcwB0AFMAZQByAHYAZQByAAAAAAA=
```

   The NTLM message is as follows:

```
00000000:4e 54 4c 4d 53 53 50 00 02 00 00 00 14 00 14 00     NTLMSSP.........
00000010:38 00 00 00 05 82 8a a2 79 45 9d e4 44 b8 06 2d     8....,Š¢yE•äD¸.-
00000020:00 00 00 00 00 00 00 00 64 00 64 00 4c 00 00 00     ........d.d.L...
00000030:05 02 ce 0e 00 00 00 0f 54 00 45 00 53 00 54 00     ..Î.....T.E.S.T.
00000040:53 00 45 00 52 00 56 00 45 00 52 00 02 00 14 00     S.E.R.V.E.R.....
00000050:54 00 45 00 53 00 54 00 53 00 45 00 52 00 56 00     T.E.S.T.S.E.R.V.
00000060:45 00 52 00 01 00 14 00 54 00 45 00 53 00 54 00     E.R.....T.E.S.T.
00000070:53 00 45 00 52 00 56 00 45 00 52 00 04 00 14 00     S.E.R.V.E.R.....
00000080:54 00 65 00 73 00 74 00 53 00 65 00 72 00 76 00     T.e.s.t.S.e.r.v.
00000090:65 00 72 00 03 00 14 00 54 00 65 00 73 00 74 00     e.r.....T.e.s.t.
000000a0:53 00 65 00 72 00 76 00 65 00 72 00 00 00 00 00     S.e.r.v.e.r.....
```

5. The client sends a **POP3_AUTH_NTLM_Blob_Command** command that contains a base64 encoded NTLM **AUTHENTICATE_MESSAGE** message (as described in [MS-NLMP]).

   The POP3 message is as follows:

```
TlRMTVNTUAADAAAAGAAYAGIAAAAYABgAegAAAAAAAABIAAAACAAIAEgAAAASABIA
UAAAAAAAAACSAAAABYKIogUBKAoAAAAPdQBzAGUAcgBOAEYALQBDAEwASQBFAE4A
VAAOarJ6lZ5ZNwAAAAAAAAAAAAAAAAAAAACD9mD8jmWs4FkZe59/nNb1cF2HkL0C
GZw=
```

   The NTLM message is as follows:

```
00000000:4e 54 4c 4d 53 53 50 00 03 00 00 00 18 00 18 00     NTLMSSP.........
00000010:62 00 00 00 18 00 18 00 7a 00 00 00 00 00 00 00     b.......z.......
00000020:48 00 00 00 08 00 08 00 48 00 00 00 12 00 12 00     H.......H.......
00000030:50 00 00 00 00 00 00 00 92 00 00 00 05 82 88 a2     P.......'....,ˆ¢
00000040:05 01 28 0a 00 00 00 0f 75 00 73 00 65 00 72 00     ..(.....u.s.e.r.
00000050:4e 00 46 00 2d 00 43 00 4c 00 49 00 45 00 4e 00     N.F.-.C.L.I.E.N.
00000060:54 00 0e 6a b2 7a 95 9e 59 37 00 00 00 00 00 00     T..j²z•ţY7......
00000070:00 00 00 00 00 00 00 00 00 00 83 f6 60 fc 8e 65     ..........ƒö`üŢe
```

```
00000080:ac e0 59 19 7b 9f 7f 9c d6 f5 70 5d 87 90 bd 02      ¬àY.{ŸoeÖõp]‡•½.
00000090:19 9c .oe
```

6. The server sends a **POP3_AUTH_NTLM_Fail_Response** message. The POP3 message is as follows:

```
-ERR, Error: Command not valid
```

5 Security

The following sections specify security considerations for implementers of the NTLM POP3 Extension.

5.1 Security Considerations for Implementers

Implementers ought to familiarize themselves with the security considerations associated with using NTLM authentication. Those security considerations are described in [MS-NLMP].

5.2 Index of Security Parameters

Security parameter	Section
NTLM	2 and 3

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 updates to those products.

- Microsoft Exchange Server 2003
- Microsoft Exchange Server 2007
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Microsoft Exchange Server 2019 Preview
- Microsoft Office Outlook 2003
- Microsoft Office Outlook 2007
- Microsoft Outlook 2010
- Microsoft Outlook 2013
- Microsoft Outlook 2016
- Microsoft Outlook 2019 Preview

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates 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 1.3: POP3 servers support at least one authentication mechanism. Office Outlook 2003 supports **USER** and **PASS** verbs, as defined in [RFC1939]. Office Outlook 2007 supports the AUTH verb, as defined in [RFC1734]. Office Outlook 2003 and Office Outlook 2007 do not support the **APOP** command, as specified in [RFC1939] section 7. Both Office Outlook 2003 and Office Outlook 2007 support **AUTH NTLM**.

<2> Section 3.2.5.3: In Exchange 2003, messages received by using MAPI are converted to **MIME** the first time they are retrieved and subsequently stored. The MIME size can be different before the message is retrieved than after it is converted to MIME. This is necessary because NTLM messages contain data outside the **ASCII** character range, whereas POP3 supports only ASCII characters.

<3> Section 3.2.5.3: In Exchange 2003, the MIME stream is preserved. In Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016, and Exchange 2019 Preview the MIME stream is not preserved. Only content of the **best body** as specified in [MS-OXBBODY] is preserved. The order in which the best body part is selected is as follows:

1. Enriched Text Format
2. HTML
3. **Plain text**

Therefore, the MIME stream is regenerated every time a message is retrieved. The alternative body parts are regenerated on demand as the message is retrieved. In Exchange 2010, the MIME headers and body parts of messages are stored if the headers existed when the messages were delivered. In Microsoft Exchange Server 2010 Service Pack 1 (SP1), Exchange 2013, Exchange 2016, andExchange 2019 Preview, messages are not stored in MIME format. Messages are converted from MAPI to MIME before being sent to the client. When the client requests the size of a message before retrieving the actual message itself, the MIME size provided is the size associated with the message as a MAPI property. When the client retrieves the message, the message is converted from MAPI to MIME, and the message size calculated thereafter can be different from the size calculated from the MAPI property. Additionally, if the message is modified by MAPI, the size is likely to change again, as is the corresponding MIME size after actual conversion from MAPI to MIME.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

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.
- A document revision that captures changes to protocol functionality.

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 **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

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

Section	Description	Revision class
All	Updated supported products throughout document.	Major
6 Appendix A: Product Behavior	Updated list of products.	Major

8 Index