[MS-SAMS]:
Security Account Manager (SAM) Remote Protocol Specification (Server-to-Server)

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's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). 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.
- **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
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	2.0	Major	Updated and revised the technical content.
07/10/2007	2.0.1	Editorial	Revised and edited the technical content.
08/17/2007	2.1	Minor	Revised content based on Trustee feedback.
09/21/2007	2.1.1	Editorial	Revised and edited the technical content.
10/26/2007	2.2	Minor	Updated deep references to MS-SAMR.
01/25/2008	2.2.1	Editorial	Revised and edited the technical content.
03/14/2008	2.2.2	Editorial	Revised and edited the technical content.
06/20/2008	3.0	Major	Updated and revised the technical content.
07/25/2008	3.0.1	Editorial	Revised and edited the technical content.
08/29/2008	3.0.2	Editorial	Revised and edited the technical content.
10/24/2008	3.0.3	Editorial	Revised and edited the technical content.
12/05/2008	4.0	Major	Updated and revised the technical content.
01/16/2009	5.0	Major	Updated and revised the technical content.
02/27/2009	5.0.1	Editorial	Revised and edited the technical content.
04/10/2009	6.0	Major	Updated and revised the technical content.
05/22/2009	7.0	Major	Updated and revised the technical content.
07/02/2009	8.0	Major	Updated and revised the technical content.
08/14/2009	9.0	Major	Updated and revised the technical content.
09/25/2009	9.1	Minor	Updated the technical content.
11/06/2009	9.1.1	Editorial	Revised and edited the technical content.
12/18/2009	9.1.2	Editorial	Revised and edited the technical content.
01/29/2010	9.1.3	Editorial	Revised and edited the technical content.
03/12/2010	9.1.4	Editorial	Revised and edited the technical content.

Date	Revision History	Revision Class	Comments
04/23/2010	9.1.5	Editorial	Revised and edited the technical content.
06/04/2010	9.1.6	Editorial	Revised and edited the technical content.
07/16/2010	9.1.6	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	9.1.6	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	9.1.6	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	9.1.6	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	9.1.6	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	9.1.6	No change	No changes to the meaning, language, or formatting of the technical content.

Contents

1 Introduction

It is useful to review the Active Directory Technical Specification, as specified in [MS-ADTS], the Netlogon Remote Protocol Specification, as specified in [MS-NRPC], and the Security Account Manager (SAM) Remote Protocol Specification (Client-to-Server), as specified in [MS-SAMR] before reading this document to understand the context and dependencies for this protocol.

This document specifies the Security Account Manager (SAM) Remote Protocol (Server-to-Server). **Domain controllers (DCs)** use this protocol to forward time-critical database changes to the **primary domain controller (PDC)**, and to forward time-critical database changes from a **read-only domain controller (RODC)** to a **writable NC replica** within the same **domain** outside the normal replication protocol. This protocol is used only between **Active Directory** servers in the same domain. Beginning with the Windows Server 2008 Beta 2 operating system, this protocol was extended to forward certain non–time-critical write operations from an RODC to a writable NC replica.

The SAM Remote Protocol (Server-to-Server) is motivated by the requirement to propagate a subset of database changes to the PDC more quickly than the Directory Replication Service (DRS) Remote Protocol (as specified in [MS-DRSR]). This rapid propagation is used for sensitive information when the delay imposed by standard Active Directory replication creates either an unwelcome burden on the user or creates a risk to the enterprise. An example of the former is a password change operation; if the password is not made available rapidly, a user can experience unpredictable authentication failures as the new password is tried against domain controllers that have not yet replicated it. An example of the latter is when an account is locked out due to multiple password failures; the lockout condition, and, equally important, the lockout-cleared condition, must be propagated rapidly throughout the domain.

Windows Server® 2008 operating system introduced a new type of domain controller, the RODC. Extensions to the protocol are motivated by the requirement to support the chaining of certain write operations such as when a machine sends a request to an RODC to change its password, as specified in [MS-NRPC] section 1.3.4. The RODC cannot service the database change but instead sends the change request over the SAM Remote Protocol (Server-to-Server) to a Windows Server 2008 or Windows Server® 2008 R2 operating system DC that applies the database change. Similarly, there are a few messages specified in the SAM Remote Protocol (Client-to-Server) that request database changes, and, when an RODC receives these, they are forwarded to a writable NC replica. Because these messages are part of [MS-SAMR], they are described in detail within that specification. However, when received at an RODC, they are forwarded to another DC; therefore, the message processing to affect the forwarding is described in this specification. Within this specification, these methods are referred to as forwarded SAM Remote Protocol (Client-to-Server) messages.

The complement of this behavior, which is not addressed in this protocol, is the logic within the Active Directory servers, which synchronizes server replication state with that of the PDC when a corresponding failure occurs. That is, there is code within the Active Directory components to pull the current state of an account from the PDC when an authentication failure occurs, using the fact that if the password was changed recently, the PDC has a more up-to-date copy.

This protocol is used in Microsoft Windows® 2000 Server operating system, Windows Server® 2003 operating system, Windows Server 2008, and Windows Server 2008 R2 (but not in Microsoft Windows NT® 4.0 operating system). The motivation for this protocol does not exist in Windows NT 4.0 because, in version 4.0, only one DC can accept updates; therefore, a centralized DC with the most up-to-date information exists naturally. Because Windows 2000 Server, Windows Server 2003, Windows Server 2008, and Windows Server 2008 R2 support multi-master updates across all DCs, this protocol was invented to address the lack of a centralized, most-up-to-date source of password information.

1.1 Glossary

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

Active Directory
domain
domain controller (DC)
globally unique identifier (GUID)
little-endian
LM hash
NT hash
originating update
primary domain controller (PDC)
read-only domain controller (RODC)
relative identifier (RID)
requestor
responder
security identifier (SID)
writable NC replica

The following terms are specific to this document:

NC replica: A tree of objects whose root object is identified by the naming context (NC), which is a dsname. See **primary domain controller (PDC)**.

primary domain controller (PDC) emulator: This term was introduced in the Windows 2000 Server version of the Windows operating system. There is no meaningful distinction between the phrase "PDC emulator" and the acronym "**PDC**" other than to note that the phrase is not used with respect to Windows NT 4.0.

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-ADA1] Microsoft Corporation, "Active Directory Schema Attributes A-L", June 2007.

[MS-ADA2] Microsoft Corporation, "Active Directory Schema Attributes M", July 2006.

[MS-ADA3] Microsoft Corporation, "Active Directory Schema Attributes N-Z", July 2006.

[MS-ADTS] Microsoft Corporation, "Active Directory Technical Specification", July 2006.

[MS-DRSR] Microsoft Corporation, "Directory Replication Service (DRS) Remote Protocol Specification", July 2006.

[MS-DTYP] Microsoft Corporation, "Windows Data Types", January 2007.

[MS-ERREF] Microsoft Corporation, "Windows Error Codes", January 2007.

[MS-KILE] Microsoft Corporation, "Kerberos Protocol Extensions", July 2006.

[MS-NRPC] Microsoft Corporation, "Netlogon Remote Protocol Specification", March 2007.

[MS-RPCE] Microsoft Corporation, "Remote Procedure Call Protocol Extensions", July 2006.

[MS-SAMR] Microsoft Corporation, "Security Account Manager (SAM) Remote Protocol Specification (Client-to-Server)", July 2006.

[RFC1274] Barker, P., and Kille, S., "The COSINE and Internet X.500 Schema", RFC 1274, November 1991, http://www.ietf.org/rfc/rfc1274.txt

[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

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

1.3 Overview

The SAM Remote Protocol (Server-to-Server) includes four sets of messages. The first set specifies messages that a domain controller (DC) sends to a primary domain controller (PDC) within the same domain to communicate select state changes in the **requestor's** database. The second and third sets specify messages that a read-only domain controller (RODC) forwards to a writable NC replica within the same domain to affect state changes in the **responder's** database. The fourth set, forwarded SAM Remote Protocol (Client-to-Server) messages, is different in that it uses a different transport protocol, as specified in [MS-SAMR].

This protocol is different from the Directory Replication Service (DRS) Remote Protocol (specified in [MS-DRSR]) in that (1) it is a "push" model from requestor to responder, and (2) it is used to communicate very limited, non-extensible, predefined state changes.

With the exception of the third set, this protocol is implemented within the Netlogon Remote Protocol [MS-NRPC] and requires that a Netlogon session be established between the requestor and the responder.

1.4 Relationship to Other Protocols

Much of this protocol sits directly on top of the Netlogon Remote Protocol, as specified in [MS-NRPC], and uses the NetrLogonSendToSam method to transmit the data. This method is specifically designed for this protocol. NetrLogonSendToSam is specified in [MS-NRPC] section 3.5.5.8.4. The remaining messages of this protocol use [MS-SAMR]; the message processing descriptions explicitly mention the use of [MS-SAMR] in these cases.

1.5 Prerequisites/Preconditions

The transport must be operational on both the requestor and the responder side of the protocol.

The responder must be aware of the domain **security identifier (SID)** of the domain it is servicing.

1.6 Applicability Statement

This protocol is applicable for DC to PDC (within the same domain) communication and for RODC to DC (within the same domain) communication. The message processing for the NetrLogonSendToSam method (as specified in [MS-NRPC] section 3.5.5.8.4) enforces that this protocol is used only for communication from a non-PDC DC to a PDC, or from an RODC to a DC.

1.7 Versioning and Capability Negotiation

There is no versioning or capability negotiation in this protocol. However, details about how to handle version or capability differences for both requestor and responder are specified in section 3.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

This protocol does not use any additional standards assignments other than what is specified in [MS-NRPC] section 1.9.

2 Messages

2.1 Transport

The transport for this protocol, unless otherwise noted, MUST be the NetrLogonSendToSam method, as specified in [MS-NRPC] section 3.5.5.8.4.

There is no additional configuration required over the Netlogon RPC protocol requirements. Details about transport for the Netlogon Remote Protocol are specified in [MS-NRPC] section 2.1.

Unless otherwise noted, security services for the SAM Remote Protocol (Server-to-Server) (including authentication, authorization, secrecy, and integrity) MUST be handled at the transport layer (that is, the Netlogon Remote Protocol).

Those messages that do not use the NetrLogonSendToSam method (in particular, forwarded SAM Remote Protocol (Client-to-Server) messages) MUST use transport identical to what is described in [MS-SAMR] section 2.1. Details about these messages are specified in section 2.2.5.

2.2 SAM Server-to-Server Request Message Syntax

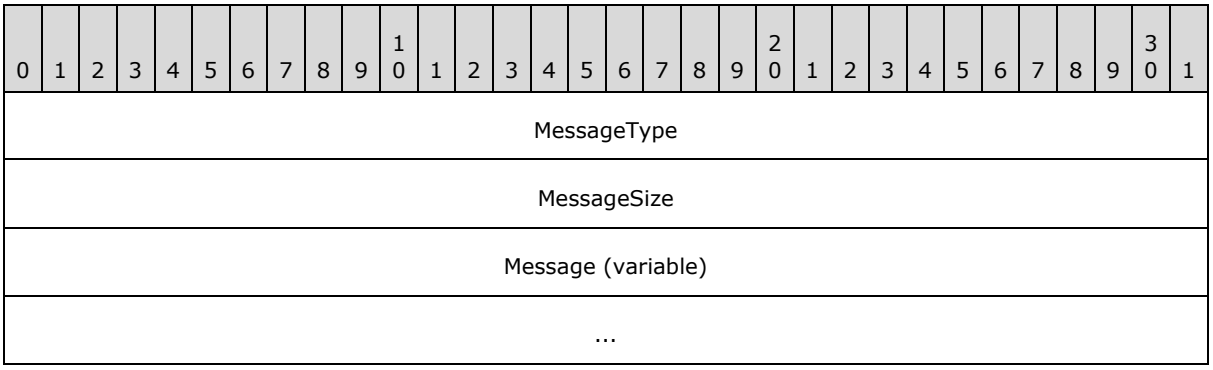
This section specifies the format of the bytes in the protocol. Bytes are presented as variable-length structures.

- Details about a base format that is used for all messages are specified in Base Request Message (section 2.2.1).
- Details about specific messages are specified in PasswordUpdate Request Message (section 2.2.2), ResetBadPwdCount Request Message (section 2.2.3), and PasswordUpdateForward Request Message (section 2.2.4).
- Details about forwarding password-change request messages are specified in section 2.2.5.
- The format of the return value and values used in this protocol is specified in Return Codes (section 2.2.8).

2.2.1 Base Request Message

Each request message is a formatted sequence of bytes. All 32-bit values interpreted as unsigned integers MUST be encoded in **little-endian** format (for all messages).

The syntax of a request is represented by the following diagram.



MessageType (4 bytes): A 32-bit, unsigned integer that indicates the type of request message. The field MUST be one of the following values.

Value	Meaning
PASSWORD_UPDATE_MSG 0x00000000	Message is a PasswordUpdate request.
RESET_PWD_COUNT_MSG 0x00000001	Message is a ResetBadPwdCount request.
FWD_PASSWORD_UPDATE_MSG 0x00000002	Message is a PasswordUpdateForward request.<1>
FWD_LASTLOGON_TS_UPDATE_MSG 0x00000003	Message is a LastLogonTimeStampUpdatesForward request.

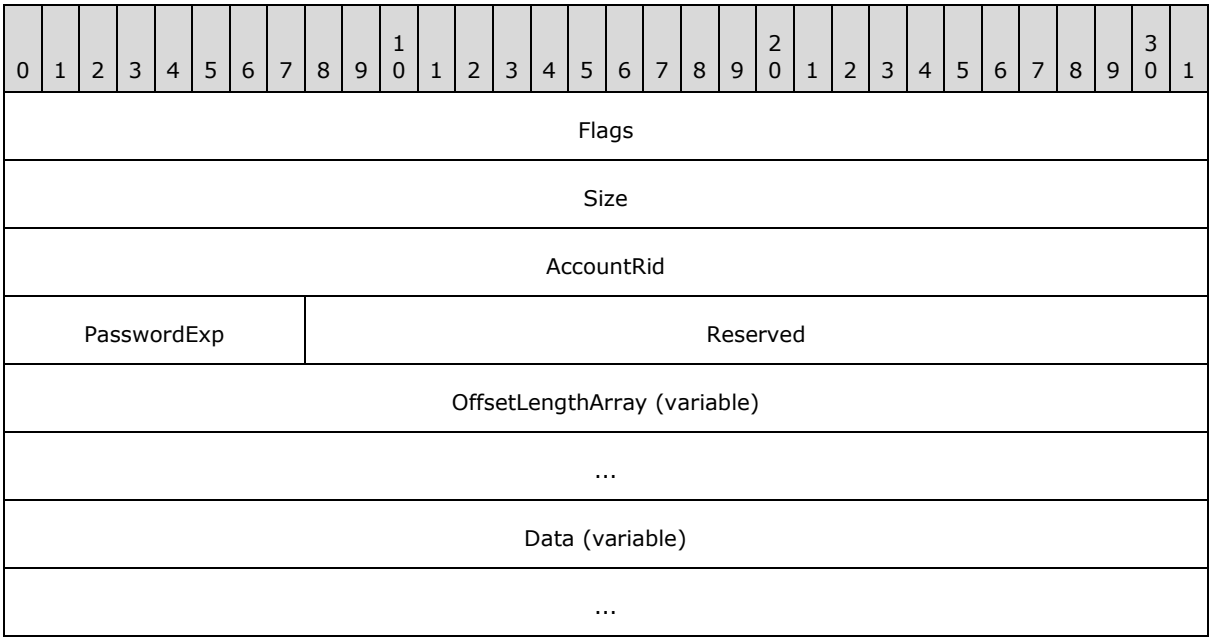
MessageSize (4 bytes): A 32-bit, unsigned integer that specifies the size of the **Message** field, in bytes.

Message (variable): A BLOB of data that constitutes a request message. This field MUST be **MessageSize** bytes long. The interpretation of the data in the **Message** field depends on the type of the request, which is indicated by the **MessageType** field. Message types are described in sections 2.2.2, 2.2.3, 2.2.4, and 2.2.7.

2.2.2 PasswordUpdate Request Message

The PasswordUpdate request message requests a change in password-related attributes for the directory object specified in the message. The message processing details are specified in section 3.3.5.2.

The layout of the PasswordUpdate request message is shown in the following diagram. This message MUST be carried in the **Message** field of the structure defined in section 2.2.1.



Flags (4 bytes): A bitmask with the following values defined. All bits that can be set, as specified below, can be set in any combination by the requestor with the exception of LM and NT; these bits MUST both be set or both be cleared. The responder MUST ignore the LM bit if it is set and the NT bit is not set.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Y	X	LM	NT	UN	PE	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

- **Y (Reserved)**: This bit SHOULD be set to zero and MUST be ignored on receipt.<2>
- **X (Reserved)**: These bits MUST be set to zero.
- **LM (FLAG_LM_HASH)**: This bit MUST be set if the message contains an LM hash. The hash value MUST be 16 bytes long.
- **NT (FLAG_NT_HASH)**: This bit MUST be set if the message contains an NT hash. The hash value MUST be 16 bytes long.
- **UN (FLAG_ACCOUNT_UNLOCKED)**: This bit MUST be set if the target account is to be unlocked. The corresponding **Offset** and **Length** fields in **Data** and **OffsetLengthArray** MUST be zero on request and MUST be ignored on receipt.
- **PE (FLAG_MANUAL_PWD_EXPIRY)**: This bit MUST be set if the **PasswordExp** field is valid for this request. The corresponding **Offset** and **Length** fields in **Data** and **OffsetLengthArray** MUST be zero on request and MUST be ignored on receipt.<3>

Size (4 bytes): A 32-bit, unsigned integer that contains the number of bytes in the PasswordUpdate request message, starting with (and including) the **Flags** field to (and including) the variable length of **OffsetLengthArray**. This information (the size) can be inferred on receipt from the bits set in the **Flags** field; for more information, see the description for **OffsetLengthArray**. This field is useful to quickly determine the start of the **Data** section.

AccountRid (4 bytes): A 32-bit, unsigned integer that contains a **relative identifier (RID)** for a SID structure.

PasswordExp (1 byte): This byte is only tested against zero, so all non-zero values are equivalent within the protocol. This field MUST be used to infer the length of the time period for which a password is valid. Details about message processing are specified in section 3.3.5.2.

Reserved (3 bytes): This portion of the message MUST be filled with zeros and MUST be ignored on receipt.

OffsetLengthArray (variable): An array of 8-byte elements used as offset and length descriptors for the data associated with this request message. Elements in this array correspond to bits in the **Flags** field. The number of elements in this array MUST be equal to the position of the most significant bit that is set in the **Flags** field. The entries MUST be in the same order as the bits in the **Flags** field. For example, if just bit 5 (FLAG_MANUAL_PWD_EXPIRY) is set, the length of the array is 6 (elements), or 48 bytes and the elements are ordered from least significant to most significant.

For an illustration of the relationship between the **Flags** field and the **OffsetLengthArray** element, see the protocol example in section 4.1.

Each **OffsetLengthArray** element contains two 32-bit unsigned integers that MUST consist of the following fields:

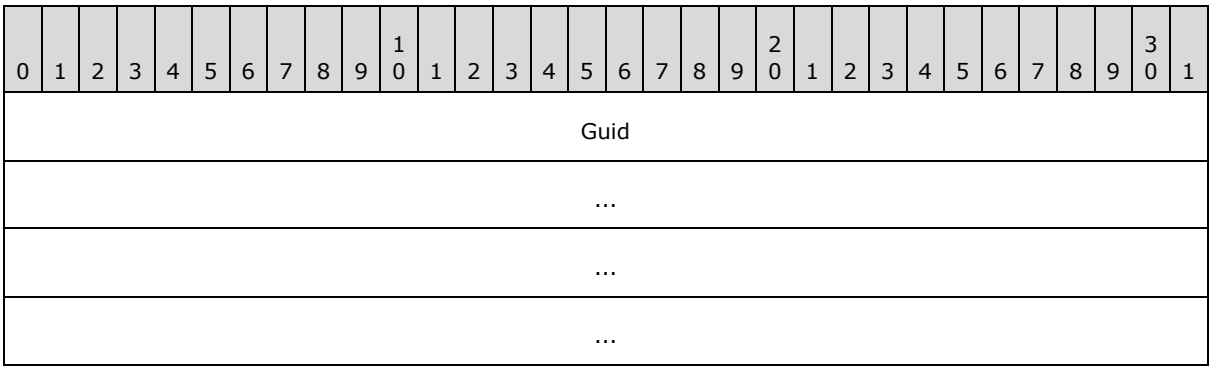
- **Offset**: The offset, in bytes, from the start of the **Data** field to the first byte of the data that corresponds to this particular element. The offset MUST be double-byte aligned.
- **Length**: The length, in bytes, of the data that corresponds to this particular element. The length MUST be double-byte aligned.

If a **Flags** bit is not set, or is set but has no data associated with it (for example, the **UN** flag), both **Offset** and **Length** MUST be zero in the corresponding **OffsetLengthArray** element and MUST be ignored on receipt.

Data (variable): A variable-sized field that MUST hold the data associated with the request. The descriptors for this data are the (**Offset**, **Length**) fields that constitute each **OffsetLengthArray** element. The length of the **Data** section MUST be no less than the maximum of **Offset** + **Length** for all elements of the **OffsetLengthArray**. This field is double-byte aligned and each entry is ordered the same as the elements in **OffsetLengthArray**; any bytes added to achieve alignment MUST have no bits set.

2.2.3 ResetBadPwdCount Request Message

The ResetBadPwdCount request message instructs the responder to update the **badPwdCount** Active Directory attribute for a specified directory object. Message processing details are specified in section 3.3.5.3. The data format for the ResetBadPwdCount request message MUST be a **globally unique identifier (GUID)**.



Guid (16 bytes): A GUID, which is a 16-byte value, as specified in [MS-DTYP] section 2.3.2. The GUID MUST be the value of objectGUID for the specified directory object.

2.2.4 PasswordUpdateForward Request Message

The PasswordUpdateForward request message requests a change in password-related attributes for the directory object specified in the message. This message SHOULD be sent only from read-only domain controllers. Message processing details are specified in section 3.3.5.4.<4>

The layout of the PasswordUpdateForward request message is shown in the following diagram. This message MUST be carried in the **Message** field of the structure defined in section 2.2.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Flags																															
Size																															
AccountRid																															
PasswordExp									Reserved																						
OffsetLengthArray (variable)																															
...																															
Data (variable)																															
...																															

Flags (4 bytes): MUST be a bitmask with the following values defined.<5>

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
AN	CP	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

- **AN (FLAG_ACCOUNT_NAME)**: This bit MUST be set to one. The associated message data in the **OffsetLengthArray** and **Data** fields references a UTF-16 encoded string that represents the account name, represented as the value of the sAMAccountName attribute on the directory object for which the password is to be changed.
- **CP (FLAG_CLEAR_TEXT_PASSWORD)**: This bit MUST be set to one. The associated message data in the **OffsetLengthArray** and **Data** fields references a UTF-16 encoded string that represents the new password.
- **X (Reserved)**: These bits MUST be set to zero and MUST be ignored on receipt.

Size (4 bytes): A 32-bit, unsigned integer that MUST contain the number of bytes in the PasswordUpdateForward request message, starting (and including) the **Flags** field to (and including) the variable length of **OffsetLengthArray**. This information (the size) can be inferred on receipt from the bits set in the **Flags** field; for more information, see the description for **OffsetLengthArray**. This field is useful to determine quickly the start of the **Data** section.

AccountRid (4 bytes): This 32-bit field MUST be set to zero and MUST be ignored on receipt.

PasswordExp (1 byte): This byte MUST be set to zero and MUST be ignored on receipt.

Reserved (3 bytes): This portion of the message MUST be filled with zeros and MUST be ignored on receipt.

OffsetLengthArray (variable): An array of 8-byte elements used as offset and length descriptors for the data associated with this request message. Elements in this array correspond to bits in the **Flags** field. The number of elements in this array MUST be equal to the position of the most significant bit that is set in the **Flags** field. The entries MUST be in the same order as the bits in the **Flags** field. For example, if just bit 1 (FLAG_CLEAR_TEXT_PASSWORD) is set, the length of the array is 2 (elements), or 16 bytes and the elements are ordered from least significant to most significant.

For an illustration of the relationship between the **Flags** field and the **OffsetLengthArray** element, see the protocol example in section 4.1.

Each **OffsetLengthArray** element contains two 32-bit unsigned integers that MUST consist of the following fields:

- **Offset**: The offset, in bytes, from the start of the **Data** field to the first byte of the data that corresponds to this particular element. The offset is double-byte aligned.
- **Length**: The length, in bytes, of the data that corresponds to this particular element. The length is double-byte aligned.

Data (variable): A variable-sized field that MUST hold the data associated with the request. The descriptors for this data are the (**Offset**, **Length**) fields that constitute each **OffsetLengthArray** element. The length of the **Data** section MUST be no less than the maximum of **Offset** + **Length** for all elements of the **OffsetLengthArray**. This field is double-byte aligned and each entry is ordered the same as the elements in **OffsetLengthArray**; any bytes added to achieve alignment MUST have no bits set.

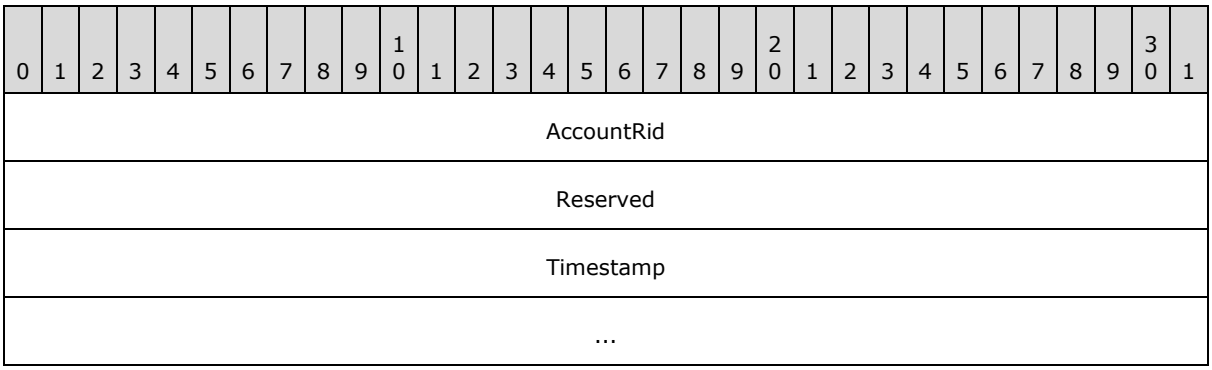
2.2.5 Forwarding Password-Change Request Messages

When an RODC receives a SamrOemChangePasswordUser2 or a SamrUnicodeChangePasswordUser2 request, as specified in [MS-SAMR] sections 3.1.5.10.2 and 3.1.5.10.3, the request MUST be forwarded to a writable NC replica. The way in which these messages are forwarded is specified in section 3.3.5.5.

2.2.6 LastLogonTimeStampUpdate Structure

The **LastLogonTimeStampUpdate** structure specifies an account and a timestamp.

The layout of the **LastLogonTimeStampUpdate** structure is shown in the following diagram.



AccountRid (4 bytes): A 32-bit, unsigned integer that contains a relative identifier (RID) for a SID structure.

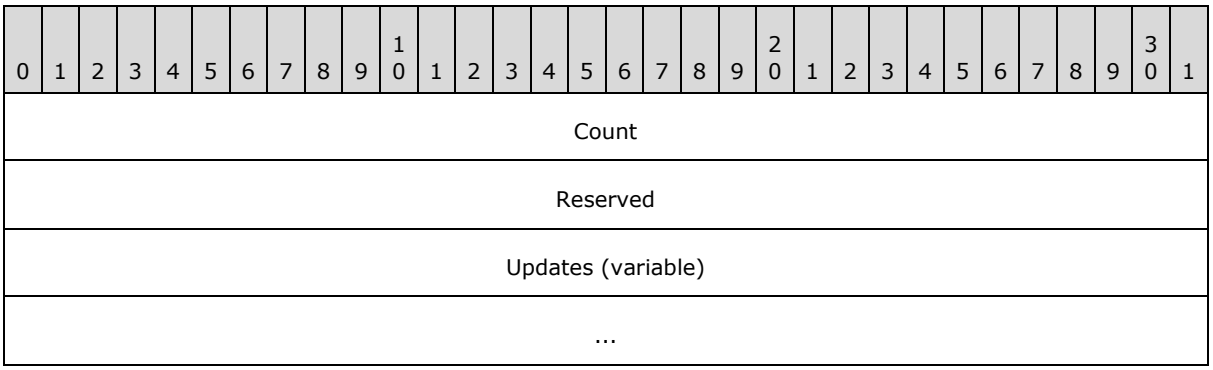
Reserved (4 bytes): A 32-bit, unsigned integer that MUST be set to zero.

Timestamp (8 bytes): A 64-bit signed integer containing the timestamp of the last logon time of the account specified by the **AccountRid** field.

2.2.7 LastLogonTimeStampUpdatesForward Request Message

The LastLogonTimeStampUpdatesForward request message requests an update to the **lastLogonTimeStamp** attribute for a specified set of user accounts. This message SHOULD be sent only from read-only domain controllers. Message processing details are specified in section 3.3.5.6.<6>

The layout of the LastLogonTimeStampUpdatesForward request message is shown in the following diagram. This message MUST be carried in the **Message** field of the structure defined in section 2.2.1.



Count (4 bytes): A 32-bit, unsigned integer that MUST contain the number of elements in the **Updates** field.

Reserved (4 bytes): A 32-bit, unsigned integer that MUST be set to zero.

Updates (variable): An array of LastLogonTimeStampUpdate structures.

2.2.8 Return Codes

The return code generated by the responder MUST be an NTSTATUS code (specified in [MS-ERREF] section 2.3). There are four distinguished values that can be returned by this protocol.

Return code	Symbolic name	Meaning
0x00000000	STATUS_SUCCESS	Success.
0xC0000058	STATUS_UNKNOWN_REVISION	The responder did not understand some part of the message sent; for example, the message type.
0xC0000059	STATUS_REVISION_MISMATCH	The responder supports the message type sent, but does not understand the format of the message; for example, expected flags are not set.
0xC0000064	STATUS_NO_SUCH_USER	The responder was not able to locate the directory object referenced in the message.

This protocol can return other error codes—for example, if a database operation fails. Other codes MUST be taken from the list specified in [MS-ERREF] section 2.3, or created (as specified in [MS-ERREF]). These other error codes MUST all be treated as equivalent by the client.

3 Protocol Details

3.1 Details Common to Both Requestor and Responder

3.1.1 Abstract Data Model

This protocol operates on a domain directory database, the data model for which is described in [MS-ADTS] section 3. For convenience, this section contains sufficient information from [MS-ADTS] section 3 to describe the message processing of this protocol.

The directory database is composed of a set of named objects. The name format is an X.500 name, as specified in [RFC1274]; therefore, the objects are arranged in a hierarchy by name. Each object name MUST be unique within the directory.

Each object possesses a collection of attributes. Each attribute is identified by a name specified in the attribute ldapDisplayName. For example, the X.500 name of the object is a single-valued attribute with the ldapDisplayName: distinguishedName. The complete list of attributes and associated constraints is specified in [MS-ADA1], [MS-ADA2], and [MS-ADA3].

Objects are retrieved from the directory database by specifying attribute-value constraints that the object attributes (and their values) MUST satisfy. Attribute values are updated by identifying the target object by distinguishedName and specifying the new set of attribute-value pairs.

Implementations must support creating, reading, updating, and deleting multiple objects, attributes, and attribute values with ACID (that is, atomic, consistent, isolated, and durable) properties. Such an update is referred to as a transaction in this specification.

Directory attributes and their semantics relevant to this protocol follow:

objectGUID: A GUID value (128 bit). Each object MUST have this attribute, and this attribute value MUST be unique within the universe. This value MUST be invariant throughout the object's lifetime; therefore, this attribute is a candidate primary key for database objects.

objectSid: The SID of an account.

pwdLastSet: The time (64-bit value) at which the **unicodePwd** value was last set; units MUST be 100 nanosecond time slices since January 1, 1601, midnight (GMT).

dbcsPwd: The **LM hash** of a clear-text password.

unicodePwd: The **NT hash** of a clear-text password.

badPwdCount: The number of logon attempts with a bad password since the last successful log on within the observation period specified in the attribute **lockoutObservationWindow**.

lockoutTime: The time at which the **badPwdCount** value exceeded the **lockoutThreshold** value; units are 100 nanosecond time slices since January 1, 1601, midnight (GMT).

lockoutThreshold: The number of invalid password attempts allowed before subsequent authentication attempts fail.

sAMAccountName: The logon name of an account.

lastLogonTimeStamp: The time (a 64-bit value) at which the user last logged on; units MUST be 100 nanosecond time slices since January 1, 1601, midnight (GMT).

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

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Requestor Details

3.2.1 Abstract Data Model

Details about the abstract data model are specified in section 3.1.1.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

3.2.4.1 Common to All Messages

None.

3.2.4.2 PasswordUpdate Request

This message is used to communicate the state change of a password for a directory entry to the PDC on a best-effort basis; if the PDC cannot be reached or the update fails for any reason, the resulting error MUST be ignored. This message MUST be triggered by the following database updates on a non-PDC; this message MUST NOT be triggered when updates are made on a PDC.

- **unicodePwd**: When this attribute is updated, a message with the FLAG_NT_HASH and FLAG_LM_HASH flags MUST be sent. The values of the NT hash and LM hash MUST be read from the requestor's database (**unicodePwd** and **dbcsPwd** attributes, respectively).

- **lockoutTime**: When this attribute is updated with a value of 0, a message with the FLAG_ACCOUNT_UNLOCKED flag MUST be sent.
- **pwdLastSet**: When this attribute is updated with a value of 0, a message with the FLAG_MANUAL_PWD_EXPIRY flag MUST be sent.

If more than one update occurs in a given transaction, the updates described above MUST be combined into one message so that the responder can make any updates, if necessary, in a single transaction.

3.2.4.3 ResetBadPwdCount Request

This message is used to communicate to the PDC if the **badPwdCount** attribute must be reset to 0. This message MUST be triggered by the following database update on a non-PDC.

When a non-PDC services a successful logon from either the Kerberos Protocol Extensions (as specified in [MS-KILE]) or the Netlogon Remote Protocol (as specified in [MS-NRPC]), the previous value of the local **badPwdCount** attribute was nonzero, and the local **badPwdCount** attribute is set to 0 as a result of the successful logon, the non-PDC MUST issue a ResetBadPwdCount request to the PDC on a best-effort basis; if the PDC cannot be reached or the update fails for any reason, the resulting error MUST be ignored.

3.2.4.4 PasswordUpdateForward Request

This message is used to communicate a request to change the state of a password for a directory entry. This message MUST be triggered only at an RODC by receipt of a NetrServerPasswordSet or NetrServerPasswordSet2 message, as specified in [MS-NRPC] sections 3.5.5.4.6 and 3.5.5.4.5, with AccountType of WorkstationSecureChannel, ServerSecureChannel, or CdcServerSecureChannel. The message is sent to the DC with which the RODC has established its secure channel, as specified in [MS-NRPC]. If the forwarding operation returns an error, the operation fails and the error MUST be propagated back to the client that originated the password update request.

3.2.4.5 Forwarding a Password-Change Request

A request to change the state of a password for a directory entry is forwarded from an RODC to a writable NC replica. This behavior is triggered only at an RODC, and only by receipt of a SamrOemChangePasswordUser2 or a SamrUnicodeChangePasswordUser2 message, as specified in [MS-SAMR] sections 3.1.5.10.2 and 3.1.5.10.3. If the forwarding operation returns an error, the operation fails and the error MUST be propagated back to the client that originated the password change request.<7>

Upon receiving one of the password-change messages, the RODC MUST process the data from the message subject to all of the following constraints:

1. The RODC MUST locate a writable domain controller for the same domain.<8>
2. The RODC MUST open an RPC binding handle to the domain controller located above in the context of the requestor. The binding handle is specified in [MS-RPCE] section 2.
3. The RODC MUST send the same RPC message that it received to the domain controller, using the binding handle opened above and the same input parameters. For example, if the RODC received a SamrUnicodeChangePasswordUser2 message, the RODC sends a SamrUnicodeChangePasswordUser2 message with identical parameters except that the binding handle is the handle opened above. Details about the message parameters of SamrOemChangePasswordUser2 and SamrUnicodeChangePasswordUser2 are specified in [MS-SAMR] sections 3.1.5.10.2 and 3.1.5.10.3.

4. The RODC MUST initialize any output parameters with values returned from the target domain controller.

5. The RODC SHOULD return to the requestor the error status returned from the target domain controller.<9>

3.2.4.6 LastLogonTimeStampUpdatesForward Request

This message is used to update the **lastLogonTimeStamp** attribute for one or more directory entries. This message is forwarded from an RODC to a writable NC replica. This behavior is triggered only at an RODC, and only as a result of a logon attempt by the specified directory entries. Logon attempts by the directory entries SHOULD NOT be affected if the update request returns an error.

3.2.5 Message Processing Events and Sequencing Rules

All errors MUST be returned to the calling layer. Furthermore, implementations MUST NOT retry the operation upon failure because subsequent database updates (either at the requestor or responder) may render the information stale in the request. This directive applies to all messages.<10>

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Responder Details

3.3.1 Abstract Data Model

Details about the abstract data model are specified in section 3.1.1.

3.3.2 Timers

None.

3.3.3 Initialization

The responder MUST begin listening for incoming requests.

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

In the following sections, the notation "Message.PasswordUpdate" is used to refer to the PasswordUpdate (section 2.2.2) request message. By using this convention, Message.PasswordUpdate.Flags, for example, refers to the **Flags** field of the PasswordUpdate request message. Similarly, "Message.ResetBadPwdCount" refers to the ResetBadPwdCount (section 2.2.3) request message, and "Message.PasswordUpdateForward" refers to the PasswordUpdateForward (section 2.2.4) request message.

For each message in this section, an informative description of the message is presented first; then, a normative description about how the message must be processed is given.

3.3.5.1 Message Type

3.3.5.1.1 Non-Normative Description

The messages described in sections 3.3.5.2, 3.3.5.3, 3.3.5.4, and 3.3.5.6 contain a message-type field. Message processing for those messages begins with the responder examining the message type.

3.3.5.1.2 Normative Specification

If Message.MessageType is not one of the expected values, as defined in Base Request Message (section 2.2.1), the responder MUST return an error. Otherwise, the responder MUST process the message based on the message type as follows:

Message type	Message processing
PASSWORD_UPDATE_MSG	3.3.5.2
RESET_PWD_COUNT_MSG	3.3.5.3
FWD_PASSWORD_UPDATE_MSG	3.3.5.4
FWD_LASTLOGON_TS_UPDATE_MSG	3.3.5.6

3.3.5.2 PasswordUpdate Request Message

3.3.5.2.1 Non-Normative Description

This message is used by DCs to communicate to the PDC both the NT hash and the LM hash of a recently changed password. The target user is identified by a RID, and the NT hash and LM hash is sent in the clear (the transport encrypts on the wire). If the password also immediately expires (that is, the user must change the password upon the next logon), this is also communicated. As such, the server message processing changes the local database to reflect the requested message. This processing is only executed on a PDC.

3.3.5.2.2 Normative Specification

On receiving this message, the responder SHOULD<11> return STATUS_NOT_SUPPORTED if either the responder is not the PDC or the requestor is an RODC. Otherwise, the responder MUST process the data from the message subject to all of the following constraints. All of the following actions MUST be performed in the same transaction:

1. The responder SHOULD validate the integrity of the message with respect to embedded offsets and sizes. Responder implementations SHOULD return STATUS_INVALID_PARAMETER upon receiving malformed messages.<12>

2. If no bits are set in Message.PasswordUpdate.Flags, no action MUST be performed, and the responder MUST return STATUS_INVALID_PARAMETER to the requestor. If bits are set in Message.PasswordUpdateFlags that MUST be 0 (as specified in section 2.2.2), STATUS_REVISION_MISMATCH MUST be returned.

3. The responder SHOULD return STATUS_SUCCESS to the requestor, and, as a background operation, SHOULD<13> perform a "replicateSingleObject" operation with the DN of the DSA object of the requestor DC, and the DN of the object that has an **objectSid** attribute value that corresponds to the value constructed by concatenating the Message.PasswordUpdate.Rid field with the configured domain SID.<14> This operation is specified in [MS-ADTS] section 3.1.1.3.3.17.

4. If no errors occur during the message processing, the responder MUST return STATUS_SUCCESS; otherwise, the responder MUST return an error code, as specified in section 2.2.8.

5. If the background operation fails for any reason, the responder MUST perform the following tasks in the background.

    1. If the FLAG_NT_HASH flag is present in Message.PasswordUpdate.Flags, the responder MUST perform all of the following operations on the target object in the database.

        1. Update the **unicodePwd** attribute with the data value supplied in Message.PasswordUpdate.Data, as specified by the fourth array element in Message.PasswordUpdate.OffsetLengthArray.

        2. If the FLAG_LM_HASH flag is present in Message.PasswordUpdate.Flags, update the **dbcsPwd** attribute with the data value supplied in Message.PasswordUpdate.Data, as specified by the third array element in Message.PasswordUpdate.OffsetLengthArray.

        3. Update the **pwdLastSet** attribute to the current time.

    2. If the FLAG_ACCOUNT_UNLOCKED flag is present in Message.PasswordUpdate.Flags, the responder MUST update the **lockoutTime** attribute of the target object in the database to 0.

    3. If the FLAG_MANUAL_PWD_EXPIRY or FLAG_NT_HASH flag is present in Message.PasswordUpdate.Flags, and if Message.PasswordUpdate.PasswordExp is nonzero, the responder MUST update the **pwdLastSet** attribute of the target object in the database to 0.

    4. All updates MUST occur as originating updates.

    5. All errors MUST be ignored.

3.3.5.3 ResetBadPwdCount Request Message

3.3.5.3.1 Non-Normative Description

This message indicates that the **badPwdCount** attribute on a target user must be reset to zero. The target user is indicated by a GUID value that corresponds to the **objectGUID** attribute.

3.3.5.3.2 Normative Specification

Upon receiving this message, the responder SHOULD<15> return STATUS_NOT_SUPPORTED if either the responder is not the PDC or the requestor is an RODC. Otherwise, if there is an object in the database that has an **objectGUID** attribute value that corresponds to the value in the Message.ResetBadPwdCount.Guid field, the responder MUST update the **badPwdCount** attribute to zero. The database updates MUST be done in one transaction. If the database transaction succeeds, the responder MUST return STATUS_SUCCESS; otherwise, the responder MUST return an error code, as specified in section 2.2.8.

3.3.5.4 PasswordUpdateForward Request Message

3.3.5.4.1 Non-Normative Description

This message is used by RODCs to communicate the request to change a password. The target user is identified by the user account name, and the new password is sent in clear-text by this protocol (the transport encrypts the password on the wire). As such, the server message processing changes the local database to reflect the requested message.

3.3.5.4.2 Normative Specification

Upon receiving this message, the responder SHOULD<16> return STATUS_NOT_SUPPORTED if the requestor is not an RODC. Otherwise, the responder MUST process the data from the message subject to all of the following constraints. All of the following actions MUST be performed in the same transaction:

1. The responder SHOULD validate the integrity of the message with respect to embedded offsets and sizes. Responder implementations SHOULD return STATUS_INVALID_PARAMETER upon receiving malformed messages.<17>
2. If either FLAG_ACCOUNT_NAME or FLAG_CLEAR_TEXT_PASSWORD is not set, the responder SHOULD<18> return STATUS_REVISION_MISMATCH to the requestor.
3. If any reserved flag (marked as X in PasswordUpdateForward Request Message (section 2.2.4)) is set, the responder SHOULD return STATUS_REVISION_MISMATCH.<19>
4. If there is no object in the database that has a **sAMAccountName** attribute value that corresponds to the data value supplied in Message.PasswordUpdateForward.Data that is specified by the first array element in Message.PasswordUpdateForward.OffsetLengthArray, the responder MUST return STATUS_NOT_FOUND.
5. If the responder is not a writable NC replica in the same domain as the RODC, the responder MUST return an error.
6. If RODC is not allowed to cache credentials for the target user account, as specified in [MS-DRSR] section 4.1.10.5.15, the responder MUST return STATUS_ACCESS_DENIED.
7. The state changes for the **password** attributes in the database MUST be the same as those specified for clearTextPassword in [MS-SAMR].
8. If no errors occur during message processing, the responder MUST return STATUS_SUCCESS; otherwise, the responder MUST return an error code, as specified in section 2.2.8.
9. All updates MUST occur as originating updates.

3.3.5.5 Forwarding a Password-Change Request Message

3.3.5.5.1 Non-Normative Description

When an RODC receives a SamrOemChangePasswordUser2 or SamrUnicodeChangePasswordUser2 request, as specified in [MS-SAMR] sections 3.1.5.10.2 and 3.1.5.10.3, the request is forwarded to a writable NC replica in the same domain by initiating a transport session in the context of the caller and resending the message to that DC. The result of message processing at the DC is then returned to the requestor.

3.3.5.5.2 Normative Specification

Upon receiving this message, the responder MUST process the data from the message subject to the constraints specified in [MS-SAMR] sections 3.1.5.10.2 and 3.1.5.10.3.

3.3.5.6 LastLogonTimeStampUpdatesForward Request Message

3.3.5.6.1 Non-Normative Description

This message is used by RODCs to forward updates of the **lastLogonTimeStamp** attribute for one or more user accounts to a writable NC replica. The RODC must be configured to allow caching of the account credentials on the RODC.

3.3.5.6.2 Normative Specification

Upon receiving this message, the responder SHOULD<20> return STATUS_NOT_SUPPORTED if the requestor is not an RODC. Otherwise, the responder MUST process the data from the message subject to all of the following constraints:

1. The responder SHOULD validate the integrity of the message with respect to embedded offsets and sizes. Responder implementations SHOULD return STATUS_INVALID_PARAMETER upon receiving malformed messages.<21>
2. If the responder is not a writable NC replica in the same domain as the RODC, then the responder SHOULD<22> return an error.
3. For each individual update contained in the **Updates** field of the LastLogonTimeStampUpdatesForward message (section 2.2.7), the responder MUST do the following:
    1. If there is no object in the database that has an **objectSid** attribute value that corresponds to the value constructed by concatenating the LastLogonTimeStampUpdate.AccountRid field with the configured domain SID, skip this update and go to the next one.
    2. Verify that the RODC is allowed to cache credentials for the object found, as specified in [MS-DRSR] section 4.1.10.5.15; otherwise, skip this update request and go to the next one.
    3. Update the **lastLogonTimeStamp** attribute of the directory entry in accordance with the algorithm specified for that attribute in [MS-ADA1]. This MUST be an originating update.
4. All errors MUST be ignored.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

4 Protocol Examples

4.1 SAM Server-to-Server Request Example

The following example shows a requestor successfully making a PasswordUpdate (section 2.2.2) request.

The message flow is trivial: The requestor makes a PasswordUpdate request (arrow A in the following figure), and the responder returns a status code (arrow B in the figure), as specified in section 2.2.2.

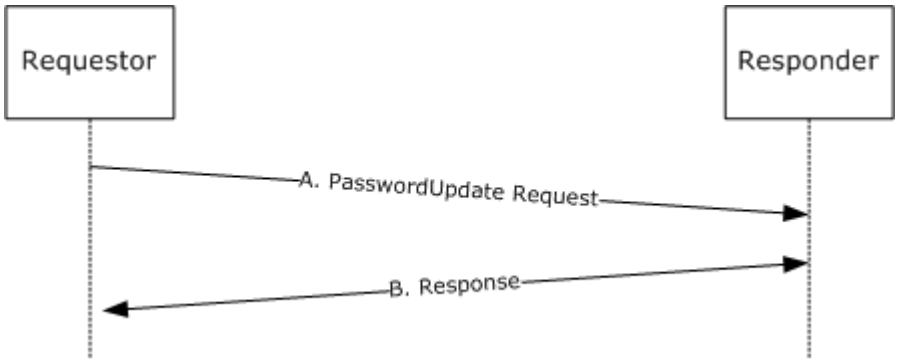


Figure 1: A requestor makes a PasswordUpdate request

The following content shows an example of a PasswordUpdate request message (arrow A in the preceding figure).

```
00000000  00 00 00 00 60 00 00 00-2C 00 00 00 40 00 00 00
00000010  F8 03 00 00 01 00 00 00-00 00 00 00 00 00 00 00
00000020  00 00 00 00 00 00 00 00-00 00 00 00 10 00 00 00
00000030  10 00 00 00 10 00 00 00-00 00 00 00 00 00 00 00
00000040  00 00 00 00 00 00 00 00-D3 58 D4 AC 2F 3C DA 54
00000050  3C FA 06 98 89 F4 AD 23-4C 23 A5 D3 67 46 2A F3
00000060  22 3D DC 54 58 34 EA 5E
```

The binary data shown above expresses the following information:

- This is a PasswordUpdate request (a value of 0 for the first 32-bit value).
- Size of the remaining data is 0x60 bytes.
- For the PasswordUpdate message:
  - The following flags are set:
    - FLAG_LM_HASH (0x04)
    - FLAG_NT_HASH (0x08)
    - FLAG_MANUAL_PWD_EXPIRY (0x20)
  - Because FLAG_MANUAL_PWD_EXPIRY represents bit 5 in the **Flag** field, there are six elements in the **OffsetLengthArray**.

- The amount of header information for the PasswordUpdate message is 0x40 bytes.
- The RID of the target directory object is 0x3F8.
- The password should be expired immediately.
- The first two and last two elements of the **OffsetLengthArray** do not have corresponding data in the **Data** portion of the message.
- The offset of the LM hash value is 0 bytes from the end of the header, and the length of the value is 0x10; therefore, the LM hash value is 0xACD458D354DA3C2F9806FA3C23ADF489.
- The offset of the NT hash value is 0x10 bytes from the end of the header, and the length of the value is 0x10; therefore, the NT hash value is 0xD3A5234CF32A466754DC3D225EEA3458.

The PasswordUpdate response message (arrow B in the preceding figure) is STATUS_SUCCESS (value 0). This is simply the RPC return code of the Netlogon RPC method **NetrLogonSendToSam**.

5 Security

5.1 Security Considerations for Implementers

Because password data can be set through a message in this protocol, this is a highly sensitive protocol. It is critical that the implementation of the transport ensures the level of authentication, authorization, secrecy, and integrity supplied by the protocol specification of the Netlogon Remote Protocol. For more information, see [MS-NRPC] section 5.

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:

- Microsoft Windows® 2000 Server operating system
- Windows Server® 2003 operating system
- Windows Server® 2008 operating system
- Windows Server® 2008 R2 operating system

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: FWD_PASSWORD_UPDATE_MSG is processed only by domain controllers running Windows Server 2008 or Windows Server 2008 R2. This message type is sent only by read-only domain controllers running Windows Server 2008 or Windows Server 2008 R2.

<2> Section 2.2.2: This bit is always set to one by the requestor and ignored by the responder. The associated message data in the **OffsetLengthArray** and **Data** fields contain a UTF-16 encoded string (which is also ignored by the responder). There is no benefit to the requestor sending this value (the UTF-16 encoded string represents the account name of the user); therefore, for the purposes of this specification, it has not been made mandatory.

<3> Section 2.2.2: The flags previously specified are supported in Windows as indicated in the following table. No flags have been deprecated.

Symbolic name	Available in
FLAG_LM_HASH FLAG_NT_HASH	Windows 2000 Server, Windows Server 2003, Windows Server 2008, and Windows Server 2008 R2.
FLAG_ACCOUNT_UNLOCKED FLAG_MANUAL_PWD_EXPIRY	Windows 2000 Server SP3, Windows Server 2003, Windows Server 2008, and Windows Server 2008 R2.

<4> Section 2.2.4: This message is processed only on domain controllers running Windows Server 2008 or Windows Server 2008 R2.

<5> Section 2.2.4: The flags specified are supported in Windows as indicated in the following table. No flags have been deprecated.

Symbolic name	Available in
FLAG_ACCOUNT_NAME	Windows Server 2008 Windows Server 2008 R2

Symbolic name	Available in
FLAG_CLEAR_TEXT_PASSWORD	Windows Server 2008 Windows Server 2008 R2

<6> Section 2.2.7: This message is processed only on domain controllers running Windows Server 2008 or Windows Server 2008 R2.

<7> Section 3.2.4.5: The Windows implementation requires a writable domain controller running Windows Server 2008 or Windows Server 2008 R2. A definition of a writable domain controller is specified in [MS-ADTS]. The Windows implementation uses the domain controller locator service, as specified in [MS-ADTS], to locate the preferred domain controller.

<8> Section 3.2.4.5: The Windows implementation requires a writable domain controller running Windows Server 2008 or Windows Server 2008 R2. A definition of a writable domain controller is specified in [MS-ADTS]. The Windows implementation uses the domain controller locator service described in [MS-ADTS] to locate the preferred domain controller.

<9> Section 3.2.4.5: The Windows implementation attempts to replicate the change immediately from the target domain controller to the RODC as an optimization. A failure to replicate the changes is ignored by the RODC because standard Active Directory replication eventually replicates the change. Details are specified in [MS-DRSR] section 4.1.10 and section 4.1.10.1.3, the Replicate Single Object operation.

<10> Section 3.2.5: All status codes returned from the responder are ignored, unless otherwise stated.

<11> Section 3.3.5.2.2: Windows 2000 Server, Windows Server 2003, and Windows Server 2008 return STATUS_ACCESS_DENIED if either the responder is not the PDC or the requestor is an RODC.

<12> Section 3.3.5.2.2: Windows 2000 Server and Windows Server 2003 do not validate the syntactic correctness of messages, and the behavior for a malformed message is undefined.

<13> Section 3.3.5.2.2: Windows 2000 and Windows Server 2003 do not execute the replicate-single-object operation, and will only perform the password hash updates synchronously during message processing. If there is no object in the database that has an **objectSid** attribute value that corresponds to the value constructed by concatenating the Message.PasswordUpdate.Rid field with the configured domain SID, Windows 2000 and Windows Server 2003 will return STATUS_NO_SUCH_USER.

<14> Section 3.3.5.2.2: Windows 2000 Server and Windows Server 2003 do not perform this operation, and act as if it failed by following the steps after step 6.

<15> Section 3.3.5.3.2: Windows 2000 Server, Windows Server 2003, and Windows Server 2008 return STATUS_ACCESS_DENIED if either the responder is not the PDC or the requestor is an RODC.

<16> Section 3.3.5.4.2: Windows 2000 Server, Windows Server 2003, and Windows Server 2008 return STATUS_ACCESS_DENIED if the requestor is not an RODC.

<17> Section 3.3.5.4.2: Windows 2000 Server and Windows Server 2003 do not validate the syntactic correctness of messages, and the behavior for a malformed message is undefined.

<18> Section 3.3.5.4.2: Windows Server 2008 R2 does not return an error if either FLAG_ACCOUNT_NAME or FLAG_CLEAR_TEXT_PASSWORD is not set.

<19> Section 3.3.5.4.2: Windows 2000 Server, Windows Server 2003, and Windows Server 2008 ignore the presence of any reserved flags and will continue processing.

<20> Section 3.3.5.6.2: Windows 2000 Server, Windows Server 2003, and Windows Server 2008 return STATUS_ACCESS_DENIED if the requestor is not an RODC.

<21> Section 3.3.5.6.2: Windows 2000 Server and Windows Server 2003 do not validate the syntactic correctness of messages, and the behavior for a malformed message is undefined.

<22> Section 3.3.5.6.2: Windows Server 2008 returns STATUS_SUCCESS without performing any of the updates specified in this section.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

S

T

V