[MS-SRPL]:
Directory Replication Service (DRS) Protocol Extensions for SMTP

Revision Summary

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

Date	Revision History	Revision Class	Comments
07/10/2007	1.2.2	Editorial	Revised and edited the technical content.
08/17/2007	1.2.3	Editorial	Revised and edited the technical content.
09/21/2007	1.2.4	Editorial	Revised and edited the technical content.
10/26/2007	1.2.5	Editorial	Revised and edited the technical content.
01/25/2008	1.2.6	Editorial	Revised and edited the technical content.

Table of Contents

1 Introduction

As specified in [MS-ADTS], **domain controllers (DCs)** use the Directory Replication Service (DRS) Remote Protocol (as specified in [MS-DRSR]) to replicate their configurations, schema, and **domain naming contexts (domain NCs)** to other domain controllers. Domain controllers are usually configured to use DRS over a **remote procedure call (RPC)** transport mechanism; however, in some environments, RPC transport is unsuitable (for example, if firewalls in the network between the domain controllers are configured to block the ports used by RPC).

This document defines the extensions to the DRS protocol for transport over the **Simple Mail Transfer Protocol (SMTP)**. These DRS Protocol Extensions for SMTP provide an alternate transport for the DRS protocol that may allow domain controllers to perform **replication** in environments where the RPC transport mechanism is unsuitable. As specified in this document, the DRS Protocol Extensions for SMTP encapsulate the DRS messages into MIME attachments (as specified in [RFC2045]) that are then sent through e-mail between domain controllers by using SMTP (as specified in [RFC2821]). This document does not define extensions or changes to the SMTP protocol itself.

1.1 Glossary

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

Abstract Syntax Notation One (ASN.1)
Active Directory (AD)
base64
Certificate
Certificate Authority (CA) (or Certification Authority)
Certificate Template
Certification
Digital Signature
Domain Controller (DC)
Domain Naming Context (Domain NC)
Forest
Global Catalog (GC)
Globally Unique Identifier (GUID)
GUID-Based DNS Name
Hash Function
Interface Definition Language (IDL)
Little-Endian
Marshal
Microsoft Interface Definition Language (MIDL)
Naming Context (NC)
Network Data Representation (NDR)
Object Identifier (OID)
Padding
Private Key
Public Key
Remote Procedure Call (RPC)
Replication
Root CA
Schema Naming Context (Schema NC)
Server Object
Universally Unique Identifier (UUID) or Globally Unique Identifier (GUID)

The following terms are specific to this document:

Active Directory Replication: The process by which two domain controllers update their copies of a naming context to contain the same information. This process is conducted by exchanging the updates that each domain controller has made to its local copy of the naming context.

Address: In the context of mail communication over SMTP, the address is the content of the **To** or the **From** field. The sender and receiver of a mail message are identified by their addresses, each of which consists of a Fully Qualified Domain Name (FQDN) portion and a user-name portion that uniquely identify the recipient within the FQDN. The FQDN portion may indicate a computer or a domain on which that user name exists.

ASN.1: See **Abstract Syntax Notation One (ASN.1)**.

CA: See Certificate Authority (CA).

Certificate Enrollment: See Certification.

DC: See Domain Controller.

Delivery Status Notification (DSN): A DSN is an SMTP message that describes the progress of delivery of another SMTP message. The SMTP MTA sends a DSN message to the sender when delivery is delayed or obstructed.

Directory Replication Service (DRS): The module of **Active Directory (AD)** that carries out replication of naming contexts between domain controllers. It uses the DRS Remote Protocol, as specified in [MS-DRSR].

Directory System Agent (DSA): The module of **Active Directory (AD)** that answers LDAP requests and manages the storage and replication of naming contexts that are stored on the domain controller.

DRS: See Directory Replication Service.

Enroll/Enrollment: See Certification.

Full Master: A domain controller with a readable and writable copy of the naming context for a domain.

GUID: See Globally Unique Identifier and Universal Unique Identifier (UUID).

KCC: See Knowledge Consistency Checker.

Key Length: The number of bits in the key that is used in an encryption algorithm.

Knowledge Consistency Checker (KCC): The module of **Active Directory (AD)** that maintains the topology of site links between domain controllers and that computes which of the domain controllers should replicate, what transport mechanism to use, and on what schedule to replicate.

Mail Transfer Agent (MTA): A client or server computer that provides a mail transport service, such as SMTP.

MTA: See Mail Transfer Agent.

NC: See Naming Context.

RC4: A commonly used stream cipher that was invented by Ronald Rivest in 1987.

Relative Distinguished Name (RDN): As specified in [X500], the portion of a distinguished name that is unique to an organization unit but might not be unique inside a domain.

Serialize: See Marshal.

Simple Mail Transfer Protocol (SMTP): A TCP/IP protocol (as specified in [RFC2821]) that is used to send and receive e-mail.

SMTP: See Simple Mail Transfer Protocol.

Tampering: Modification of data by anyone other than the intended recipient.

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.

[C706] The Open Group, "DCE 1.1: Remote Procedure Call", C706, August 1997, http://www.opengroup.org/public/pubs/catalog/c706.htm

[FIPS186] National Institute of Standards and Technology, "Federal Information Processing Standards Publication 186-2: Digital Signature Standard (DSS)", January 2000, http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf

[IEEE1363] Institute of Electrical and Electronics Engineers, "Standard Specifications for Public-Key Cryptography", 1363-2000, August 1999, http://grouper.ieee.org/groups/1363/

[ITUX680] ITU-T, "Abstract Syntax Notation One (ASN.1): Specification of Basic Notation", Recommendation X.680, July 2002, http://www.itu.int/ITU-T/studygroups/com17/languages/X.680-0207.pdf

[MS-ADSC] Microsoft Corporation, "Active Directory Schema Classes", 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-DSSP] Microsoft Corporation, "Directory Services Setup Remote Protocol Specification", July 2006.

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

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

[MS-WCCE] Microsoft Corporation, "Windows Client Certificate Enrollment Protocol Specification", July 2006.

[PKCS1] RSA Laboratories, "PKCS#1 Version 2.1: RSA Cryptography Standard", PKCS #1, June 2002, http://www.rsa.com/rsalabs/node.asp?id=2125

[RC4] The RC4 Encryption Algorithm. RSA Data Security, Inc., http://www.rsa.com/node.aspx?id=1204

Note To obtain this stream cipher that is licensed by RSA Data Security, you need to contact this company.

[RFC1034] Mockapetris, P., "Domain Names–Concepts and Facilities", RFC 1034, November 1987, http://www.ietf.org/rfc/rfc1034.txt

[RFC1321] Rivest, R. "The MD5 Message-Digest Algorithm", RFC 1321, April 1992, http://www.ietf.org/rfc/rfc1321.txt

[RFC1778] Howes, T., Kille, S., Yeong, W., and Robbins, C., "The String Representation of Standard Attribute Syntaxes", RFC 1778, March 1995, http://www.ietf.org/rfc/rfc1778.txt

[RFC2045] Freed, N., et al., "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996, http://ietf.org/rfc/rfc2045.txt

[RFC2046] Freed, N. and Borenstein, N., "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996, http://ietf.org/rfc/rfc2046.txt

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996, http://ietf.org/rfc/rfc2047.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

[RFC2315] Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", RFC 2315, March 1998, http://www.ietf.org/rfc/rfc2315.txt

[RFC2510] Adams, C. and Farrell, S., "Internet X.509 Public Key Infrastructure Certificate Management Protocols", RFC 2510, March 1999, http://www.ietf.org/rfc/rfc2510.txt

[RFC2821] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001, http://www.ietf.org/rfc/rfc2821.txt

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, http://www.ietf.org/rfc/rfc2822.txt

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, http://www.ietf.org/rfc/rfc3280.txt

[SCHNEIER] Schneier, B., "Applied Cryptography, Second Edition", John Wiley and Sons, 1996, ISBN: 0471117099.

If you have any trouble finding [SCHNEIER], please check here.

[SP800-32] National Institutes of Standards and Technology, "Introduction to Public Key Technology and the Federal PKI Infrastructure", SP800-32, February 2001, http://csrc.nist.gov/publications/nistpubs/800-32/sp800-32.pdf

[UNICODE4.0] The Unicode Consortium, "Unicode 4.0.0", http://www.unicode.org/versions/Unicode4.0.0/

[X500] ITU-T, "Information Technology - Open Systems Interconnection - The Directory: Overview of Concepts, Models and Services", Recommendation X.500, August 2005, http://www.itu.int/rec/T-REC-X.500-200508-I/en

Note There is a charge to download the specification.

[X509] ITU-T, "Information Technology - Open Systems Interconnection - The Directory: Public-Key and Attribute Certificate Frameworks", Recommendation X.509, August 2005, http://www.itu.int/rec/T-REC-X.509/en

Note There is a charge to download the specification.

[X660] ITU-T, "Information Technology - Open Systems Interconnection - Procedures for the Operation of OSI Registration Authorities: General Procedures and Top Arcs of the ASN.1 Object Identifier Tree", Recommendation X.660, August 2004, http://www.itu.int/rec/T-REC-X.660/en

Note There is a charge to download the specification.

[X680] ITU-T, "Abstract Syntax Notation One (ASN.1): Specification of Basic Notation", Recommendation X.680, July 2002, http://www.itu.int/rec/T-REC-X.680/en

Note There is a charge to download the specification.

1.2.2 Informative References

[CRYPTO] Menezes, A., Vanstone, S., and Oorschot, P., "Handbook of Applied Cryptography", 1997, http://www.cacr.math.uwaterloo.ca/hac/

[MSADRTTR] Microsoft Corporation, "Active Directory Replication Topology Technical Reference," April 2005, http://technet2.microsoft.com/WindowsServer/en/Library/1f3bb1c1-ba8a-4b4e-9f23-f240566e3d661033.mspx

[MSFT-PKI] Microsoft Corporation, "Best Practices for Implementing a Microsoft Windows Server 2003 Public Key Infrastructure", July 2004, http://technet2.microsoft.com/WindowsServer/en/library/091cda67-79ec-481d-8a96-03e0be7374ed1033.mspx

[MSFT-TEMPLATES] Microsoft Corporation, "Implementing and Administering Certificate Templates in Windows Server 2003", July 2004, http://technet2.microsoft.com/WindowsServer/en/library/c25f57b0-5459-4c17-bb3f-2f657bd23f781033.mspx

If you have any trouble finding [MSFT-TEMPLATES], please check here.

[MSSS] Microsoft Corporation, "Serialization Services", http://msdn2.microsoft.com/en-us/library/aa378670.aspx

[PUBKEY] RSA Laboratories, "Crypto FAQ: Chapter 2 Cryptography: 2.1 Cryptographic Tools: 2.1.1 What Is Public-Key Cryptography?", http://www.rsa.com/rsalabs/node.asp?id=2165

[RSAFAQ] RSA Laboratories, "Frequently Asked Questions About Today's Cryptography, Version 4.1", May 2000, http://www.rsa.com/rsalabs/faq/files/rsalabs_faq41.pdf

1.3 Protocol Overview (Synopsis)

As specified in [MS-ADTS], **domain controllers (DCs)** use the Directory Replication Service (DRS) Remote Protocol (as specified in [MS-DRSR]) to replicate their configurations, schema, and **domain**

naming context (domain NC) to other domain controllers. Domain controllers are usually configured to use DRS over an RPC transport mechanism; however, in some environments, RPC transport is unsuitable (for example, if firewalls in the network between the domain controllers are configured to block the ports used by RPC).

This document defines the extensions to the DRS Protocol for transport over the Simple Mail Transfer Protocol (SMTP). These DRS Protocol Extensions for SMTP provide an alternate transport for the DRS Protocol that may allow domain controllers to perform replication in environments where the RPC transport mechanism is unsuitable. As specified in this document, the DRS Protocol Extensions for SMTP encapsulate the DRS messages into MIME attachments (as specified in [RFC2045]) that are then sent in e-mail between **domain controllers** by using SMTP (as specified in [RFC2821]).

The DRS Protocol Extensions for SMTP specified in this document are not a general transport mechanism. They can be used only for the transport of a subset of DRS messages during replication between domain controllers. As specified in section 1.5, there are additional conditions that the configurations of the domain controllers must meet before the DRS Protocol Extensions for SMTP can be used to replicate state between the domain controllers.

When two domain controllers replicate, the domain controller that is initiating the replication is referred to as the client, and the other domain controller is referred to as the server. The basic steps of a replication are as follows:

1. The client domain controller sends a "get replicated change" request to the server domain controller.
2. The client domain controller sends a "get replicated change" request to the server domain controller.
3. The server domain controller accepts the "get replicated change" request from the client domain controller and identifies new updates for this client.
4. The server domain controller sends a "get replicated change" response to the client domain controller that is carrying those updates.
5. The client domain controller accepts the "get replicated change" response from the server domain controller and incorporates non-redundant updates from the server.

When using the DRS Protocol Extensions for SMTP, clients and servers asynchronously process batches of "get replicated change" messages. For example, a client may make multiple requests to the server before receiving a response, and a client is free to process replies at a later time than when the request was sent.

The following figure outlines the processing steps performed by the DRS Protocol Extensions for SMTP as it transports a "get replicated change" message (either a request or a response) to the other domain controller involved in the replication. The details of these steps are specified in section 3. When a domain controller receives an SMTP message, the steps are performed in the reverse order, starting with the SMTP MTA, and proceeding to obtaining the DRS Replication Data, which is then given to the DRS Protocol.

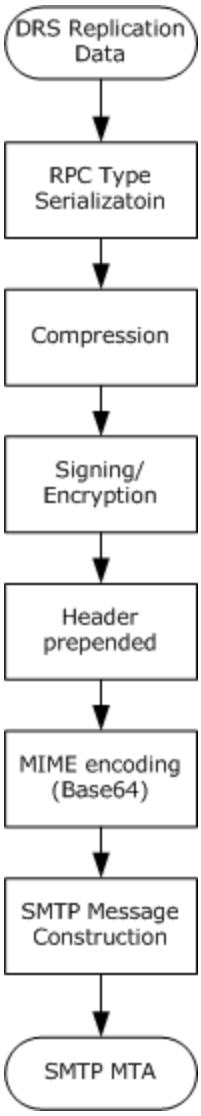


Figure 1: Transporting a "get replicated change" message to the other domain controller involved in a replication

The result is an SMTP message structured as shown in the following figure. The message is given to the SMTP MTA for delivery to the remote domain controller.

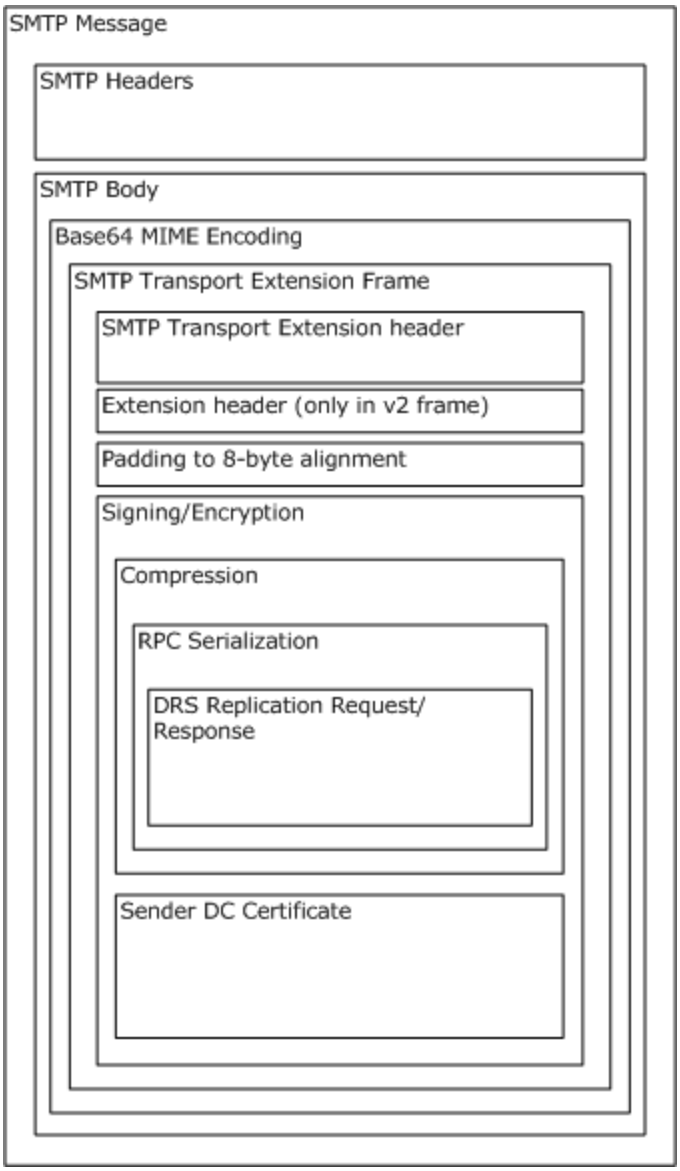


Figure 2: SMTP message given to the SMTP MTA for delivery to the remote domain controller

The specification of the DRS Protocol Extensions for SMTP depends on the terminology and concepts that are specified in [MS-ADTS] and [MS-DRSR]. For illustrative examples, see [MSADRTTR]. Summarizing information as specified in [MS-ADTS], all domain controllers are configured to be part of a **forest**. Each domain controller stores two or more **naming contexts (NCs)**, where an NC is a conceptual directory that maps names to attributes. Domain controllers use the Directory Replication Service Protocol (as specified in [MS-DRSR]) to maintain consistency between NCs that are stored on multiple domain controllers.

The properties and configuration of a forest are defined by the values in a **configuration NC** and a **schema naming context (schema NC)**. Each domain controller maintains a copy of its forest's configuration NC and schema NC. Changes made to any copy of these NCs, at any domain

controller, are replicated to the copies at all other domain controllers in the forest. Domain controllers also store a domain NC for one or more domains. A domain controller may be configured to have a read/write copy of the domain NC, in which case the domain controller is a **full master** for the domain, or it may have a read-only copy of the NC, in which case it is a **global catalog**. As part of the configuration NC for a forest, each domain controller in the forest is assigned to a **site** (conceptually, a site is a geographic region).

The following figure illustrates an example of a forest that contains two sites (Site1 and Site2). The domain controllers in Site2 (DC2-1 and DC2-2) are full master for the Domain2 NC and global catalogs for the Domain1 NC. The domain controller DC1 in Site1 is a full master for the Domain1 NC. A scenario in which this situation might exist is the operation of a domain controller on a submarine that makes contact with its base only infrequently. The submarine would be configured as Site1 and the base as Site2.

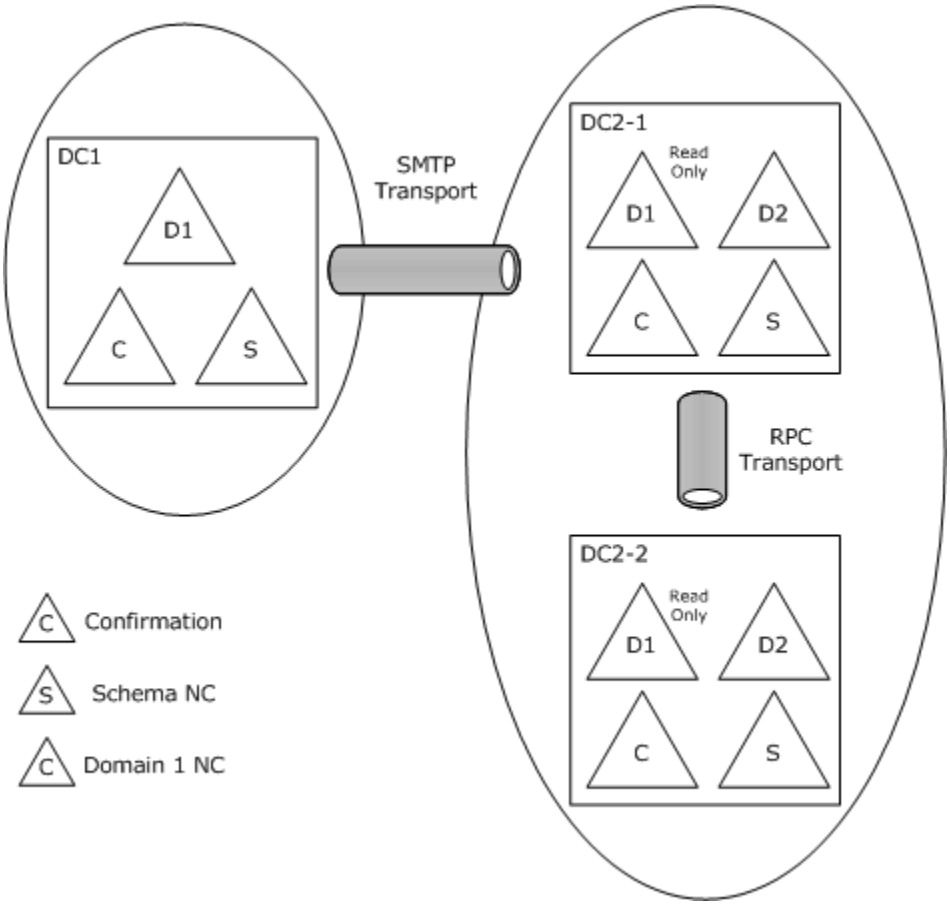


Figure 3: Forest that contains two sites (Site1 and Site2)

The configuration state in the forest's configuration NC dictates what transports may be used by DRS during replication. In the example in Figure 3, the two domain controllers in Site2 are configured to use RPC transport for their replication using DRS, as specified in [MS-DRSR], and DC1 and DC2-1 are configured to use SMTP transport for their replication using DRS, as specified in this document.

The choices regarding which domain controllers should replicate and on what schedule they should replicate are made by the **Knowledge Consistency Checker**, as specified in [MS-ADTS]. For a set of informative examples of replication topology, see [MSADRTTR].

1.4 Relationship to Other Protocols

DRS Protocol Extensions for SMTP is a means of encapsulating **serialized** DRS RPC messages and transporting them inside an SMTP mail message.

The following figure illustrates the relationship between the DRS Protocol Extensions for SMTP and other protocols.

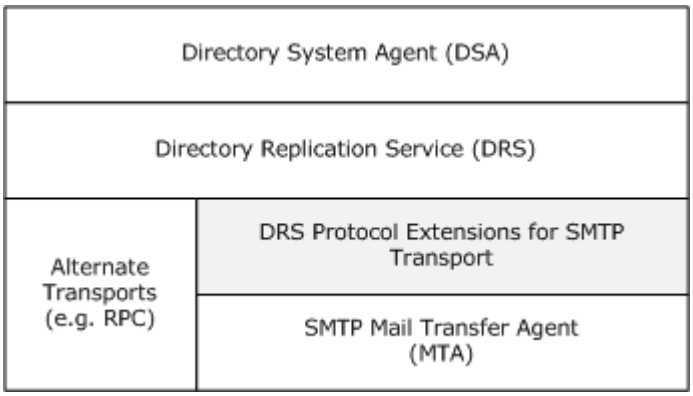


Figure 4: Relationship between DRS Protocol Extensions for SMTP and other protocols

The **Directory System Agent (DSA)** implements the functionality of a domain controller and is specified in [MS-ADTS]. The Directory Replication Service (DRS) Remote Protocol (as specified in [MS-DRSR]) controls how a naming context (NC) is replicated between domain controllers. DRS transports its messages between domain controllers by using the DRS Protocol Extensions for SMTP specified in this document or by using another transport, such as RPC (as specified in [MS-RPCE]).

In carrying out the processing steps specified in section 3, the DRS Protocol Extensions for SMTP uses the following additional protocols. Messages sent by DRS are serialized and deserialized by using the RPC serialization encode and decode process using type serialization, as specified [MS-RPCE] section 2.2.6. The DRS compression algorithm (as specified in [MS-DRSR] section 4.10.5.15) is used to compress the payload. The compressed result is optionally encrypted using **RC4** (as specified in [RC4]); it is always encapsulated in a cryptographically signed request structure, as specified in [MS-WCCE] and [PKCS1]. The hash algorithm that is used for the signature is specified in [RFC1321]. MIME encoding (as specified in [RFC2045], [RFC2046], and [RFC2047]) is used to represent binary information in a format suitable for inclusion in an SMTP message. The MIME encoded message is sent as the body of an SMTP message,as specified in [RFC2822]. An SMTP **mail transfer agent (MTA)** (as specified in [RFC2821]) is used to transport SMTP messages to the remote domain controller.

1.5 Prerequisites/Preconditions

The domain controller requires the ability to send and receive SMTP messages. Any SMTP mail transfer agent (as specified in [RFC2821]) MAY be used.<1>

The final choice of replication transport is made by the KCC, on a per-NC replica basis, as specified in [MS-ADTS]. The following requirements are necessary, but not sufficient, conditions for determining actual use of the SMTP transport.

If two domain controllers are to use the DRS Protocol Extensions for SMTP for replication, the following criteria MUST be met.

- The configuration NC on each domain controller MUST already specify the existence of a Windows **Active Directory (AD)** forest, and both domain controllers MUST be members of this forest.
- Each domain controller must have a Domain Controller **certificate**, and all Domain Controller **certificates** MUST be signed by the same **certificate authority**. Domain Controller **certificates** are as specified in section 2.3. **Certificate enrollment** and storage are specified in [MS-WCCE].
- The domain controllers MUST be configured to be in different sites.
- The configuration NC for the forest MUST specify that the DRS Protocol Extensions for SMTP may be used for replication between the domain controllers. The replication transport is governed by the configuration of connection, site link, and intersite transport objects, as specified in [MS-ADTS].
- One of the following statements MUST apply to the NC being replicated. The intuition behind these requirements is that replication between two full-master replicas of the same domain NC is not permitted via the DRS Protocol Extensions for SMTP to enforce an administrative best practice.
  - The NC is the configuration NC.
  - The NC is the schema NC.
  - Both domain controllers hold NC replicas of the same application NC.
  - Both domain controllers hold a partial read-only replica of the same NC (for example, both domain controllers are global catalogs).
  - One domain controller holds a writable full replica of its domain NC, and the other domain controller holds a partial read-only copy of that domain NC (for example, the other domain controller is a global catalog).
- The configuration NC must contain a **server object** for each domain controller. Both server objects must contain a mailAddress attribute, and the mailAddress MUST be a syntactically valid SMTP recipient (as specified in [RFC2822]).

1.6 Applicability Statement

The DRS Protocol Extensions for SMTP are used by domain controllers, in a forest, when they are replicating **Active Directory (AD)** contents by using the Directory Replication Service (DRS) Remote Protocol, as specified in [MS-DRSR].

The DRS Protocol Extensions for SMTP are appropriate for linking isolated, regional domains to their forest. The DRS Protocol Extensions for SMTP are appropriate for participation in global forest replicas, such as the configuration NC, schema NC, and the global catalog.

The DRS Protocol Extensions for SMTP cannot be used for replication between domain controllers that are part of the same site. The extensions cannot be used to replicate a domain between two domain controllers that are full masters of that domain. They can be used only to replicate a domain between a full master for the domain and a global catalog for that domain or between two global catalogs for that domain.

The DRS Protocol Extensions for SMTP specified in this document are not a general transport mechanism. They are defined only for transport of the IDL_DRSGetNcChanges request and response messages that are part of the DRS Remote Protocol, as specified in [MS-DRSR].<2>

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas.

- Protocol versions: The DRS Protocol Extensions for SMTP are available in two versions. They differ in the length of their preamble and whether a capability extension is carried. A receiver determines what version of the protocol is in use by the value of the **dwMsgVersion** field, as specified in sections 2.2.3 and 2.2.4.
- Capability negotiation: There is no capability negotiation when the version 1 (V1) frame format of this protocol is used. The version 2 (V2) frame format includes an explicit vector of capabilities.

Domain controllers that are running Windows 2000 MUST be sent only version 1 frames. Domain controllers that are running Windows Server 2003 or Windows Server 2008 SHOULD be sent version 2 frames, but they MAY be sent version 1 frames.<3>

Version 2 of the DRS Protocol Extensions for SMTP frame carries the DRS_EXTENSIONS_INT protocol element, as described in [MS-DRSR] section 5.27. These capabilities perform identical functions as the core DRS Remote Protocol; they are not interpreted by the DRS Protocol Extensions for SMTP. In the case of the DRS Protocol Extensions for SMTP, these capabilities are present in every message, in contrast to the core DRS Remote Protocol, where they are exchanged only on the first IDL_DRSBind message.

1.8 Vendor-Extensible Fields

There are no vender-extensible fields.

1.9 Standards Assignments

Parameter	Value	Reference
Well-known TCP/IP port for Simple Mail Transfer Service (SMTP)	25	[RFC2437]

2 Messages

The following sections specify how DRS Protocol Extensions for SMTP messages are transported and DRS Protocol Extensions for SMTP message syntax.

2.1 Transport

The DRS Protocol Extensions for SMTP use SMTP (as specified in [RFC2821]) as a transport.

The endpoint for the protocol is the mailbox that receives DRS SMTP messages on the target domain controller. This mailbox is identified by an addr-spec (as specified in [RFC2822] section 3.4.1) that includes both a local-part and domain. Each domain controller publishes its preferred mailAddresses (section 2.4.2.1) in the directory. The DRS layer provides to the DRS Protocol Extensions for SMTP the mailAddresses to be used as the SMTP sender and recipient. The particular local-part and domain used in the mailAddress are opaque to this protocol.<4>

A domain controller MAY interpret SMTP **delivery status notifications (DSNs)** for error reporting purposes.<5>

2.2 Message Syntax

Conceptually, the frame MAIL_REP_MSG used by the DRS Protocol Extensions for SMTP is a backward-compatible structure that has evolved over two successive product versions.

The DRS Protocol Extensions for SMTP message frame MUST be in the form of a **MAIL_REP_MSG_V1** message or a **MAIL_REP_MSG_V2** message, as specified in the following sections.

2.2.1 DRS_MSG

The data carried by this protocol MUST be a **Network Data Representation (NDR)**–encoded blob that contains one of the following DRS Remote Protocol structures, as specified in [MS-DRSR].

- DRS_MSG_GETCHGREQ_V4
- DRS_MSG_GETCHGREQ_V7
- DRS_MSG_GETCHGREPLY_V1
- DRS_MSG_GETCHGREPLY_V6

Other **DRS** message structures MUST NOT be carried as payload of the DRS Protocol Extensions for SMTP.

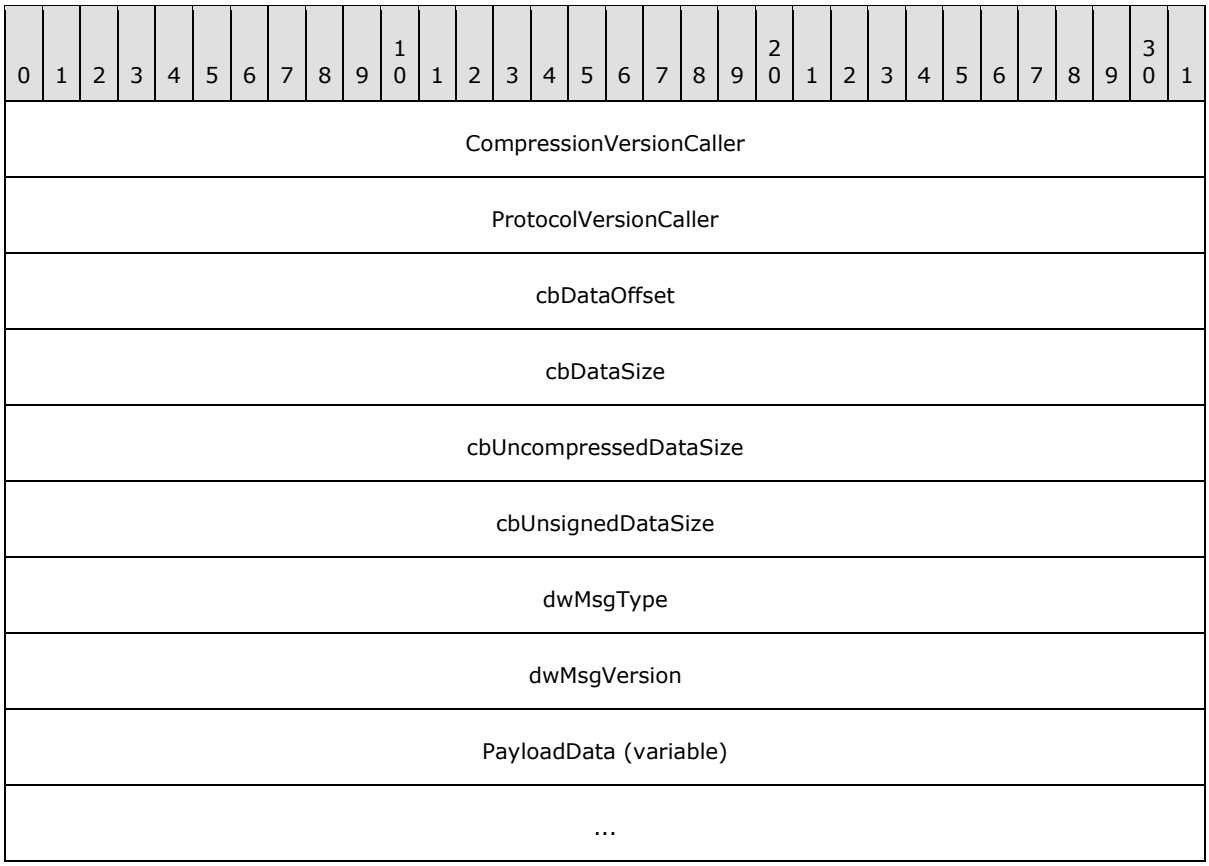
2.2.2 CURRENT_PROTOCOL_VERSION

The following constants are used by the DRS Protocol Extensions for SMTP.

Constant/value	Description
CURRENT_PROTOCOL_VERSION 0x0000000B	This constant specifies the current version of the DRS Protocol Extensions for SMTP.<6>

2.2.3 MAIL_REP_MSG_V1

This structure defines the V1 format for a DRS Protocol Extensions for SMTP frame. This structure is not part of the RPC data stream and, thus, is not present in the supplied **Interface Definition Language (IDL)**. The RPC data stream from the higher-layer DRS protocol is encapsulated by this structure and is carried within the payload data field. This frame is "hand-**marshaled**" by the protocol code, as specified in sections 3.2 and 3.3. It appears at the beginning of the attachment data sent using SMTP. All numeric header fields MUST be in the **little-endian** format.



CompressionVersionCaller: A 32-bit, unsigned integer that indicates the compression algorithm that is used for the data in this message. This field MUST be set to a valid value for the enumerated type DRS_COMP_ALG_TYPE, as specified in [MS-DRSR]. If the header field dwMsgType.CP is 0, this field MUST be sent as 0 and ignored on receipt.

ProtocolVersionCaller: A 32-bit, unsigned integer that indicates the protocol version for this message. This field MUST be set to the value of the **CURRENT_PROTOCOL_VERSION**.

cbDataOffset: In a V1 frame-type, the field SHOULD be sent as 0, and MUST be ignored on receipt. The value 32, the V1 header size, MAY be used.<7>

cbDataSize: A 32-bit, unsigned integer that indicates the size of the payload data (not including this header), starting with the first byte of Payload data, in bytes.

cbUncompressedDataSize: A 32-bit, unsigned integer that indicates the size of Send-Message-Serialized-Data (as specified in section 3.2.1), not including this header, before

compression, in bytes. If the header field dwMsgType.CP is 0, this field MUST be sent as 0 and ignored on receipt.

cbUnsignedDataSize: A 32-bit, unsigned integer that indicates the size of Send-Message-Compressed-Data (as specified in section 3.2.1), not including this header, before encryption, in bytes.

dwMsgType: An unsigned 32-bit field that specifies message type options. This value is a combination of one or more of the following bit fields. Bits not specified below MUST be set to 0 by the sender, and MUST be ignored by the receiver.

Value	Meaning
RQ (Request) 0x01000000	If set, indicates that this is a Request Message. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
RP (Response) 0x02000000	If set, indicates that this is a Response Message. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
SN (Signed) 0x00000020	If set, indicates that this message is signed. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
SL (Sealed) 0x00000040	If set, indicates that this message is sealed. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
CP (Compressed) 0x00000080	If set, indicates that this message is compressed. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.

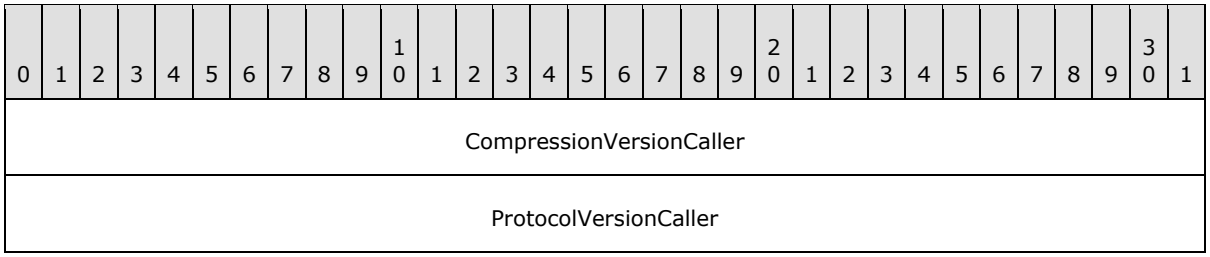
dwMsgVersion: dwMsgVersion: A 32-bit, unsigned integer that indicates whether this DRS Message is a V1 request or a V1 response. The value of this field MUST be one of the following.

Value	Meaning
0x000000001	This message contains a V1 response.
0x000000004	This message contains a V1 request.

PayloadData: Variable-length region that contains the Send-Message-Payload,as specified in section 3.2.1.

2.2.4 MAIL_REP_MSG_V2

This structure defines the V2 format for a DRS Protocol Extensions for SMTP frame. It appears at the beginning of the attachment data sent using SMTP. All numeric header fields MUST be in the little-endian format.



cbDataOffset
cbDataSize
cbUncompressedDataSize
cbUnsignedDataSize
dwMsgType
dwMsgVersion
dwExtFlags
cbExtOffset
ExtCapabilityVector (variable)
...
Padding (variable)
...
PayloadData (variable)
...

CompressionVersionCaller: A 32-bit, unsigned integer that indicates the compression algorithm that is used for the data in this message. This field MUST be set to a valid value for the enumerated type DRS_COMP_ALG_TYPE, as specified in [MS-DRSR]. If the header field dwMsgType.CP is 0, this field MUST be sent as 0 and ignored on receipt.

ProtocolVersionCaller: A 32-bit, unsigned integer that indicates the protocol version for this message. This field MUST be set to the value of the **CURRENT_PROTOCOL_VERSION**.

cbDataOffset: A 32-bit, unsigned integer that indicates the offset in bytes to the Payload Data in this message, calculated from the beginning of this frame (from the first byte of the **CompressionVersionCaller** field). This field MUST be a multiple of 8 bytes for alignment.

cbDataSize: A 32-bit, unsigned integer that indicates the size of the Payload Data (not including this header, starting with the first byte at **cbDataOffset**), in bytes.

cbUncompressedDataSize: A 32-bit, unsigned integer that indicates the size of Send-Message-Serialized-Data (as specified in section 3.2.1), not including this header, before compression, in bytes.

cbUnsignedDataSize: A 32-bit, unsigned integer that indicates the size of Send-Message-Compressed-Data (as specified in section 3.2.1), not including this header, before encryption, in bytes.

dwMsgType: An unsigned 32-bit field that specifies message type options. This value is a combination of one or more of the following bit fields. Bits not specified below MUST be set to zero by the sender and MUST be ignored by the receiver.

Value	Meaning
RQ (Request) 0x01000000	If set, indicates that this is a Request Message. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
RP (Response) 0x02000000	If set, indicates that this is a Response Message. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
SN (Signed) 0x00000020	If set, indicates that this message is sealed. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
SL (Sealed) 0x00000040	If set, indicates that this message is sealed. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.
CP (Compressed) 0x00000080	If set, indicates that this message is compressed. This field is one of 32 single-bit flags that are included in the **dwMsgType** field.

dwMsgVersion: A 32-bit, unsigned integer that indicates whether this DRS Message is a V2 request or a V2 response. The value of this field MUST be one of the following.

Value	Meaning
0x000000006	This message contains a V2 response.
0x000000007	This message contains a V2 request.

dwExtFlags: A 32-bit, unsigned integer that contains the **dwFlags** field of the **DRS_EXTENSIONS_INT** structure, as specified in [MS-DRSR] section **5.35**. The **dwFlags** field appears as bytes 5–8 of the structure, whose bytes are numbered starting from 1.

cbExtOffset: A 32-bit, unsigned integer that indicates the offset, from the start of the frame (from the first byte of the **CompressionVersionCaller** field), in bytes, to the **ExtCapabilityVector** field. This field MUST be a multiple of 8-bytes for alignment. This field MUST be 0x00000028.

ExtCapabilityVector: The variable length region that contains the entire **DRS_EXTENSIONS_INT** structure, as specified in [MS-DRSR] section **5.35**. The contents of bytes 5–8 of this structure also appear in **dwExtFlags**.

Padding: Between 0 and 7 bytes, as required, to make sure that **PayloadData** begins on an 8-byte aligned boundary. If the length of this field is greater than 0 bytes, this field MUST be sent as 0 and ignored on receipt.

PayloadData: Variable-length region that contains the Send-Message-Payload (as specified in section 3.2.1). This field MUST begin at offset **cbDataOffset**.

2.3 Certificate Formats

An X.509 **certificate** (as specified in [X509]) that encapsulates a **public key** for the purpose of secure communication is a prerequisite for using the DRS Protocol Extensions for SMTP. Each domain controller participating in directory e-mail replication must have a **certificate** and **private key** that is available locally, that is unique to that computer, and that has been issued by a common **root CA**.

This **certificate** MAY be either a domain controller Replication **certificate** (as specified in section 2.3.1), or a Directory E-mail Replication **certificate**, as specified in section 2.3.2.<8>

The following **object identifiers (OIDs)** specify algorithms that are used for signing and sealing, as specified in PKCS #1 ([PKCS1]) and [SCHNEIER].

```
OID RSA MD5 (hash function) "1.2.840.113549.2.5"
OID RSA RC4 (encryption algorithm) "1.2.840.113549.3.4"
```

Both Domain Controller Replication **certificates** and Directory E-mail Replication **certificates** are X.509 **certificates** that contain the following X.509v1 fields.

- Version
- Serial Number
- Signature Algorithm
- Valid From
- Valid To
- Subject (distinguished name of the domain controller)
- Issuer
- Public Key

2.3.1 Domain Controller Replication Certificate

The Domain Controller Replication **certificate** is defined as an X.509 (as specified in [X509]) certificate with specific extensions and values, as described below.

A Domain Controller Replication **certificate** contains X.509v1 fields, as specified in section 2.3.

A Domain Controller Replication **certificate** also contains the following X.509v3 extensions identified in [RFC3280] section 4.2.1.

- Authority Key Identifier
- Subject Key Identifier
- Authority Information Access
- Key Usage (**Digital Signature**, Key Encipherment (a0))
- Subject Alternative Name

- The **Certificate** Subject Alternative Name section must contain the globally unique identifier (GUID) of the domain controller object in the directory and the Domain Name System (DNS) name. For example:
  - Other Name: 1.3.6.1.4.1.311.25.1 = ac 4b 29 06 aa d6 5d 4f a9 9c 4c bc b0 6a 65 d9
  - <DNS name of the domain controller>
- CDP (CRL Distribution Point)
- Enhanced Key Usage
  - Client Authentication (1.3.6.1.5.5.7.3.2)
  - Server Authentication (1.3.6.1.5.5.7.3.1)

A Domain Controller Replication **certificate** also contains the following X.509v3 extensions specific to Microsoft.

- Microsoft-defined X.509v3 extension for **certificate template** name

<9>

2.3.2 Directory E-mail Replication Certificate

The Directory E-mail Replication **certificate** is defined as an X.509 (as specified in [X509]) certificate with specific extensions and values, as described below.

A Directory E-mail Replication **certificate** contains X.509v1 fields, as specified in section 2.3.

A Directory E-mail Replication **certificate** also contains the following X.509v3 extensions, as specified in [RFC3280] section 4.2.1.

- Authority Key Identifier
- Subject Key Identifier
- Authority Information Access
- Key Usage
  - Digital Signature, Key Encipherment = (a0)
- Subject Alternative Name

  The **certificate** Subject Alternative Name section must contain the globally unique identifier (GUID) of the domain controller object in the directory and the Domain Name System (DNS) name. For example:
  - Other Name: 1.3.6.1.4.1.311.25.1 = ac 4b 29 06 aa d6 5d 4f a9 9c 4c bc b0 6a 65 d9
  - <DNS name of the domain controller>
- CDP (CRL Distribution Point)
- Enhanced Key Usage
  - Client Authentication (1.3.6.1.5.5.7.3.2)

- Server Authentication (1.3.6.1.5.5.7.3.1)

- Extended Key Usage
  - Directory E-mail Replication OID = 1.3.6.1.4.1.311.21.19

A Directory E-mail Replication **certificate** also contains the following X.509v3 extensions specific to Microsoft.

- Microsoft-defined X.509v3 extension for Application Policies
- Microsoft-defined X.509v3 extension for **certificate** template information

<10>

2.4 Active Directory Objects

2.4.1 Computer Object

The Computer object represents a computer in the **Active Directory (AD)** forest, and it is found by default at the following **relative distinguished name (RDN)** within the domain NC:

"CN=computername,CN=Computers"

For this RDN, "computername" is the host part of the computer's FQDN. As specified in section 2.3, the issued **certificate** MUST contain the GUID of the Computer object of that domain controller to be a valid domain controller **certificate**. When domain controllers exchange **certificates** during protocol operations (as specified in section 3), the domain controllers further verify that the **certificate** contains the GUID of a Computer object that has not been deleted.

The schema definition for the Computer object is specified in [MS-ADSC].

2.4.2 Server Object

This is the **Active Directory (AD)**Server object from the **Active Directory (AD)** Schema, as specified in [MS-ADTS] section 7.1.1.2.2.

The Server object represents a computer in the **Active Directory (AD)** forest that is a directory server. The Server object contains an nTDSDSA object (section 2.4.3) with configuration information for that server. The Server object is found at the following RDN within the configuration NC:

"CN=servername,CN=Servers,CN=sitename, CN=Sites"

For this RDN, "servername" is the host part of the computer's FQDN, and "sitename" is the administrator-specified name of the site to which the server belongs.

2.4.2.1 mailAddress Attribute

The mailAddress attribute of the Server object (section 2.4.2) that corresponds to a domain controller indicates the SMTP recipient **address** used by that server for the DRS Protocol Extensions for SMTP Transport.

The mailAddress is a Unicode string that MUST meet the requirements of an addr-spec, as specified in [RFC2822] section 3.4.1. This includes being in the form local-part@domain.

A directory server that is sending an update request via the DRS Protocol Extensions for SMTP determines the appropriate e-mail To address field by querying the value of this attribute for the

destination computer's Server object. The directory server sets the From address field by querying the value of this attribute for its own Server object.

2.4.3 nTDSDSA Object

The nTDSDSA object is the **Active Directory (AD)**Server object (section 2.4.2) from the **Active Directory (AD)** Schema, as specified in [MS-ADTS] section 7.1.1.2.2.

On a domain controller, the nTDSDSA object represents the replication agent, which is responsible for processing the DRS Protocol.

The nTDSDSA object has the RDN of "CN=NTDS Settings" and is a child of the Server object of the domain controller.

The GUID of this nTDSDSA object is invariant for the lifetime of the domain controller. The implementation MAY use this GUID value as an alternative identifier for the domain controller.<11>

2.4.3.1 msDs-Behavior-Version Attribute

The nTDSDSA object (section 2.4.3) class contains the msDs-Behavior-Version attribute. This attribute specifies the domain controller version. The contents of this attribute are as specified in [MS-ADTS] section 7.1.4.2.

3 Protocol Details

The higher layer is the Directory Replication Service (DRS) Remote Protocol (as specified in [MS-DRSR]). The lower layer is the SMTP MTA delivery function.

The DRS Protocol Extensions for SMTP serializes a DRS message and encloses that message in its own message envelope, which is called the DRS Protocol Extensions for SMTP frame. The DRS Protocol Extensions for SMTP first inserts the extension frame into a MIME attachment, then inserts the MIME attachment into an SMTP message, and finally gives the SMTP message to the lower-layer SMTP MTA for delivery to the recipient.

3.1 Common Details

3.1.1 Abstract Data Model

Each domain controller that uses the DRS Protocol Extensions for SMTP maintains the following state.

- SMTP-ADDR-DC-CERT-MAP (address): A dictionary that maps from the SMTP mail address of a domain controller to the Domain Controller **certificate** of that domain controller. The receiving role (section 3.3) populates the dictionary over time through requests that it receives.
- Local-DC-Mail-Address: This value is the SMTP address at which the DRS Protocol Extensions for SMTP on this domain controller expects to receive SMTP messages.
- Local-DC-Certificate: This value is the domain controller **certificate** for the local domain controller.

3.1.2 Timers

None.

3.1.3 Initialization

This protocol MUST initialize Local-DC-Mail-Address with the mail address of this domain controller, as taken from the configuration NC. The configuration NC MUST include the SMTP address of the domain controller.

This protocol MUST initialize Local-DC-Certificate with a **certificate** from the Public Key Infrastructure. This **certificate** MUST meet the criteria set forth in section 2.3.

This protocol MUST initialize SMTP-ADDR-DC-CERT-MAP with an entry for the local domain controller, as follows: <Local-DC-Mail-Address, Local-DC-Certificate>.

The implementation MAY populate the map with additional entries at initialization time, although this is not required for correct operation. As an alternative, the implementation MAY populate the map with knowledge of additional partner domain controllers as they are discovered during protocol operation.<12>

The SMTP MTA MUST be initialized as needed so that it delivers messages sent with a From address of Local-DC-Mail-Address. This protocol MUST perform any initialization required so that it will be able to receive SMTP messages that are sent to Local-DC-Mail-Address. For example, the protocol may need to register the domain of Local-DC-Mail-Address in the DNS in a fashion that allows it to receive SMTP messages that are sent to the domain.

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 Sending Role Details

This section defines the steps taken when the DRS Protocol Extensions for SMTP receive a message from the higher-layer Directory Replication Service (DRS) Remote Protocol to send to another domain controller. Because this document specifies a transport protocol, the processing steps are nearly identical for a domain controller acting as a server and a domain controller acting as a client. This document describes both roles in this section with the few differences between the roles specified in the text.

3.2.1 Abstract Data Model

When the Directory Replication Service (DRS) Remote Protocol invokes the DRS Protocol Extensions for SMTP, it provides the transport with the following information.

- Send-Recipient-Mail-Address: An opaque Unicode string that contains the SMTP mail address of the recipient. The encoding for Unicode MIME is as specified in [RFC2047].
- Send-Message-Data: A sequence of bytes comprising a DRS Protocol message, as specified in section 2.2.1. The extension does not alter or interpret the content of the message during subsequent send processing in this protocol.
- Send-Message-Type: The value dictates the type of the message to send, either Request type or Response type.
- Send-Frame-Type: The value dictates the type of frame that this protocol constructs, either **MAIL_REP_MSG_V1** type or **MAIL_REP_MSG_V2** type.
- Send-Compression-Algorithm: The value dictates the compression method that this protocol uses when compressing the message. The value is type DRS_COMP_ALG_TYPE, as specified in [MS-DRSR] section 4.10.5.15.
- Send-Message-Version: A 32-bit integer quantity provided by the DRS Protocol layer that identifies the DRS structure version associated with Send-Message-Data. The value must be one of the structure version numbers specified in section 2.2.2. The extension inserts this value into the extension frame without interpretation.
- Send-Commentary: This value is a sequence of Unicode characters provided by the DRS Protocol layer. This value represents a human-readable descriptive summary of the intent of the message. The contents are opaque to this protocol.

This document uses the following working variables to represent intermediate representations of Send-Message-Data between processing steps.

- Send-Message-Serialized-Data
- Send-Message-Compressed-Data
- Send-Message-Data-Authenticated
- Send-Message-Payload
- Send-Message-Frame

Each variable represents a separate, contiguously allocated buffer.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

The Directory Replication Service (DRS) Remote Protocol layer invokes the DRS Protocol Extensions for SMTP after the construction of the DRS Protocol message, as follows:

- The domain controller, in the client role, is sending a request message. The DRS layer invokes the following send processing steps at the point indicated in the text of "Client Send Behavior," as specified in [MS-DRSR] section 4.10.4.
- The domain controller, in the server role, has received a request message, completed processing of the request, and is sending a response message. The DRS Protocol layer invokes the following described send-processing steps at the point indicated in the text of "Server Behavior," as specified in [MS-DRSR] section 4.10.5.

3.2.4.1 Serialization Processing

The DRS Protocol Extensions for SMTP MUST perform the following data marshaling procedure on the DRS Protocol message in Send-Message-Data. The extension MUST encode Send-Message-Data as an RPC IDL structured type by using Microsoft RPC Extension "Type Serialization Version 1," as specified in [MS-RPCE] section 2.2.6. (For additional examples, see [MSSS].) The result is Send-Message-Serialized-Data.

3.2.4.2 Compression Processing

The DRS Protocol Extensions for SMTP MUST perform the following data compression procedure on Send-Message-Serialized-Data. The extension MUST compress the sequence of bytes comprising the Send-Message-Serialized-Data according to the DRS compression algorithm indicated by Send-Compression-Algorithm. "TransformOutput" (as specified in [MS-DRSR] section 4.10.5.15) defines the processing steps for the DRS compression algorithms. The result is Send-Message-Compressed-Data.

3.2.4.3 Cryptographic Processing

The DRS Protocol Extensions for SMTP MUST perform a certificate service (as specified in [MS-WCCE]) cryptographic operation on Send-Message-Compressed-Data. All cryptographic operations MUST employ the **Abstract Syntax Notation One (ASN.1)** encoding, as specified in [MS-WCCE].

The certificate-based cryptographic operation consists of a conditional encryption step followed by an unconditional message-signature step. The extension MUST perform encryption on response messages and MUST NOT perform encryption on request messages.

If Send-Message-Type indicates the message is of type Response, the implementation MUST encrypt the compressed data prior to signing, as follows.

- The abstract working variable Send-Recipient-Certificate MUST be set to the value of SMTP-ADDR-DC-CERT-MAP (Send-Recipient-Mail-Address). Implementations make the **certificate** available in the map by either populating the **certificate** at initialization time or populating the **certificate** during previous receive-processing (see section 3.3.5.3).
- The extension MUST invoke certificate-based cryptographic encryption on Send-Message-Compressed-Data by using the key of Send-Recipient-Certificate. The encryption algorithm MUST be RSA RC4, as specified in [SCHNEIER].
- The extension MUST use the encrypted result as the input to the subsequent signature operation.

For a Response message, the result of the encryption step defined above MUST be cryptographically signed. For Request messages, the Send-Message-Compressed-Data MUST be cryptographically signed. The result of the cryptographic signature operation MUST be in "PKCS #7 Format," as specified in [RFC2315]. The hash function used in the signature operation MUST be RSA MD5. The result MUST include Local-DC-Certificate in the list of associated **certificates**. The result of the signing operation is Send-Message-Data-Authenticated.

3.2.4.4 Frame Message Processing

The following specifies the layout of the two defined frames in sections 2.2.3 and 2.2.4.

The variable Send-Frame-Type identifies the kind of frame that is required. The protocol MUST construct the frame according to the rules specified for that frame using the information that is provided in the abstract interface variables. The protocol uses the Send-Compression-Algorithm as the **CompressionVersionCaller** field. The protocol uses Send-Message-Data-Authenticated as the Send-Message-Payload. If the Send-Frame-Type indicates type **MAIL_REP_MSG_V2**, the protocol MUST insert the current value of DRS_EXTENSION (as specified in [MS-DRSR]) into the frame, as specified in section 2.2.4. The result is the Send-Message-Frame.

3.2.4.5 Lower-Layer SMTP MTA Interaction

The protocol prepares an SMTP message (as specified in [RFC2822]) as follows:

- The **To** field MUST be equal to the Send-Recipient-Mail-Address.
- The protocol MUST compute the **Subject** field by prepending the Unicode string "Intersite message for NTDS Replication:" to the Send-Commentary. Unicode MIME support for SMTP header fields is as specified in [RFC2047].
- The following MIME options (as specified in [RFC2045]) MUST be set in the headers of the SMTP message.
  - MIME-Version: 1.0 or higher

- Content-Transfer-Encoding: **base64**
- Content-Type: image/gif

- The protocol MUST encode Send-Message-Frame with MIME **base64** encoding [RFC2045].
- The protocol MUST use the **base64** encoded Send-Message-Frame as the body of the SMTP message.

The protocol gives the SMTP message to the SMTP MTA and directs it to perform a send operation to the Send-Recipient-Mail-Address.<13>

3.2.5 Message Processing Events and Sequencing Rules

The lower-layer SMTP delivery agent MAY return delivery status notifications (DSNs) for previously sent messages.<14>

3.2.6 Timer Events

None.

3.2.7 Other Local Events

The lower-layer SMTP MTA delivers SMTP messages on its own schedule using whatever network transport that it selects.

3.3 Receiving Role Details

This section specifies the behavior of the DRS Protocol Extensions for SMTP when the SMTP MTA receives an SMTP message. This section also defines the behavior for both servers and clients.

3.3.1 Abstract Data Model

This section defines the working variables that the protocol uses when performing in the receiving role. The protocol populates the following working variables during frame decoding, as described in subsequent sections of this topic.

- Sender-Mail-Address: This variable holds an opaque Unicode string that contains the SMTP mail address of the sender. The extension populates this field during the steps provided in section 3.3.5.1.
- Received-Message-Type: This variable indicates the type of message, which is either Request type or Response type.
- Received-Compression-Method: This variable indicates the type of compression used by the sender. The value is type **DRS_COMP_ALG_TYPE**, as specified in [MS-DRSR] section 4.10.2.13. The extension populates this field during the steps provided in section 3.3.5.2.
- Sender-DC-Certificate: This variable holds the **certificate** of the sending domain controller, as obtained during the cryptographic operation described in section 3.3.5.3.

The extension uses the following working variables to communicate intermediate data buffers between processing steps.

- Receive-Frame

- Receive-Data
- Receive-Message-Verified-Data
- Receive-Message-Deserialized-Data

Each variable represents a separate, contiguously allocated buffer. Each processing step defines the method of construction and specifies internal field alignment requirements, if any.

3.3.2 Timers

None.

3.3.3 Initialization

None.

3.3.4 Higher-Layer Triggered Events

There are no higher-layer triggered events for this role. The lower-layer SMTP MTA delivers messages to the DRS Protocol Extensions for SMTP, as described in the next section.

3.3.5 Message Processing Events and Sequencing Rules

Message processing in the DRS Protocol Extensions for SMTP begins when the SMTP MTA delivers an SMTP message to the server process for the DRS Protocol Extensions for SMTP. This operation MUST validate the frame that is received from the SMTP MTA, decode the frame into its constituent fields, and pass the resulting DRS data to the DRS Protocol layer.

3.3.5.1 SMTP Header Processing

The SMTP message MUST meet the following criteria. If any of the criteria are not met, the SMTP message MUST be dropped, and it MAY be logged. The SMTP header fields are specified in [RFC2822].

- The **To** field of the SMTP message MUST contain a single recipient, the Local-DC-Mail-Address.
- The SMTP message MUST contain a body section.
- The body of the message MUST use the following MIME options (as specified in [RFC2045]): Content-Transfer-Encoding = **base64**, Content-Type = image/gif.
- The **Subject** field MUST begin with the Unicode characters "Intersite message for NTDS Replication:"

<15>

If all criteria are met, the protocol MUST place the contents of the of the SMTP message **From** field in the Sender-Mail-Address working variable. The protocol then MUST extract the body from the SMTP message and decode the **base64** encoding. The decoded result is Receive-Frame.

3.3.5.2 Frame Message Processing

The implementation MUST ensure the validity of the Receive-Frame contents prior to their use. If any of the frame validation constraints described in section 3.3.5.6 are not met, the Receive-Frame MUST be dropped.

The protocol MUST use the contents of the Receive-Frame from the byte that begins at **cbDataOffset** for **cbDataSize** bytes as the Receive-Data payload.

The extension MUST set the working variable Received-Message-Type as follows:

- If **dwMsgType** flag RQ is set, Received-Message-Type equals Request.
- If **dwMsgType** flag RP is set, Received-Message-Type equals Response

The extension MUST set the working variable Received-Compression-Method to the value of frame field **CompressionVersionCaller**.

3.3.5.3 Cryptographic Processing

The extension MUST perform a **certificate** service [MS-WCCE] cryptographic operation on the Receive-Data. All cryptographic operations MUST employ the **Abstract Syntax Notation One (ASN.1)** encoding, as specified in [MS-WCCE].

The Receive-Data value MUST be a structure of type "PKCS #7 Format," as specified in [MS-WCCE] section 2.2.1.4.2.

The PKCS7 structure MUST contain a set of associated **certificates** that have been provided by the sender for the benefit of the receiver. The list of associated **certificates** MUST contain one Domain Controller **certificate**, as specified in section 2.3. This **certificate** is the Sender-DC-Certificate.

The protocol MUST verify the validity of the Sender-DC-Certificate, as specified in section 3.3.5.7. If the Sender-DC-Certificate is not valid, the Receive-Frame MUST be dropped.

Certificate-based cryptographic operation consists of an unconditional signature verification step, followed by a conditional decryption step. An implementation MUST perform decryption on response type messages and MUST NOT perform decryption on request type messages.

The implementation MUST perform the signature verification operation on Receive-Data. The hash function that is used MUST be RSA MD5. If the verification fails, the implementation MUST discard the message.

If Received-Message-Type indicates a Response, the cryptographically verified data MUST next be decrypted. The decryption algorithm MUST be RSA RC4 and use the key of Local-DC-Certificate. The resulting plaintext is the Receive-Message-Verified-Data.

If Received-Message-Type indicates a Request, the verified data is the Receive-Message-Verified-Data.

The implementation MUST add an entry to SMTP-ADDR-DC-CERT-MAP if Received-Message-Type is Request. The entry takes the form <Sender-Mail-Address, Sender-DC-Certificate>.

If Received-Message-Type is Response, the sender's **certificate** MAY be included in SMTP-ADDR-DC-CERT-MAP.<16>

When the implementation updates the map, the following semantics are used: The abstract state SMTP-ADDR-DC-CERT-MAP(Sender-Mail-Address) MUST be set equal to the Sender-Certificate, and any value previously stored MUST be overwritten.

3.3.5.4 Decompression and Deserialization Processing

The protocol performs a decompression step, followed by a data-unmarshaling step.

The protocol MUST apply the decompression method indicated by Received-Compression-Method. Reference "DecompressReplyMessage," ([MS-DRSR] section 4.10.6.14) specifies the processing steps for the DRS compression algorithms.

The protocol MUST deserialize the expanded result as an RPC IDL structured type by using Microsoft RPC Extension "Type Serialization Version 1," as specified in [MS-RPCE] section 2.2.6. The result is Receive-Message-Deserialized-Data.

3.3.5.5 Higher-Layer DRS Protocol Interaction

The protocol provides Receive-Message-Deserialized-Data to the DRS Protocol layer for further interpretation.

The DRS Protocol Extensions for SMTP passes operation to the DRS Protocol layer at the following processing points.

- If Received-Message-Type is a Request, the DRS Protocol layer server-role processing MUST commence as specified at the top of "Server Behavior," [MS-DRSR] section 4.10.5.
- If Received-Message-Type is a Response, DRS client-role processing MUST commence as specified in "Client Receive Behavior," [MS-DRSR] section 4.10.6.

3.3.5.6 Extension Frame Decoding and Validation

This section defines specific frame validation constraints. The implementation MUST discard frames that are not valid.

The Receive-Frame is a **MAIL_REP_MSG_V1** type if **dwMsgVersion** is either 0x00000001 or 0x00000004. To be a valid **MAIL_REP_MSG_V1**, it MUST meet the following constraints:

- Receive-Frame length > sizeof(**MAIL_REP_MSG_V1**)
- **ProtocolVersionCaller** is equal to **CURRENT_PROTOCOL_VERSION**
- **cbDataOffset** is equal to 0
- Exactly one of the **dwMsgType** RP or RQ header flags MUST be set
- **CompressionVersionCaller** is a member of **DRS_COMP_ALG_TYPE**
- sizeof(**MAIL_REP_MSG_V1**) + **cbDataSize** is less than or equal to Receive-Frame length

The Receive-Frame is a **MAIL_REP_MSG_V2** type if **dwMsgVersion** is either 0x00000006 or 0x00000007. To be a valid **MAIL_REP_MSG_V2** frame, it MUST meet the following constraints:

- Receive-Frame length > sizeof(**MAIL_REP_MSG_V2**)
- **ProtocolVersionCaller** is equal to **CURRENT_PROTOCOL_VERSION**
- **cbDataOffset** is not equal to 0
- **cbDataOffset** is 8-byte-aligned
- **cbExtOffset** is 0 or is 8-byte-aligned
- Exactly one of the **dwMsgType** RP or RQ header flags MUST be set
- **CompressionVersionCaller** is a member of **DRS_COMP_ALG_TYPE**

- sizeof(MAIL_REP_MSG) + **cbDataOffset** + **cbDataSize** ≤ Receive-Frame length
- sizeof(MAIL_REP_MSG) + **cbExtOffset** ≤ Receive-Frame length
- **cbExtOffset** is not equal to **cbDataOffset**
- If **cbExtOffset** is equal to 0
  - Error
- Else If **cbExtOffset** < **cbDataOffset**, then
  - **cbExtOffset** is equal to sizeof(MAIL_REP_MSG_V2)
  - **cbDataOffset** - **cbExtOffset** is equal to size of DRS_EXTENSIONS_INT
- Else If **cbExtOffset** > **cbDataOffset**, then
  - Error

If the Receive-Frame is neither a **MAIL_REP_MSG_V1** nor a **MAIL_REP_MSG_V2**, it MUST be considered not valid.

3.3.5.7 Certificate Post-Processing

The Sender-DC-Certificate value, as a Domain Controller **Certificate** (section 2.3), MUST contain the GUID of an **Active Directory (AD)** object.

The receiving domain controller MUST verify the following:

- That the GUID identifies an **Active Directory (AD)** object of type Computer object (section 2.4.1). The Computer object MUST NOT be in the deleted object state.
- That the Computer object is acting in the domain controller state, as determined by the userAccountControl Bits, as specified in [MS-DRSR] section 5.166.
- That there is an **Active Directory (AD)** object of type Server object (section 2.4.2) associated with the Computer object. The Server object MUST NOT be in the deleted state.
- That the Server object must have a child object, which is the DRS replication agent NTDSDSA object (section 2.4.3) for the domain controller.

When the receiving domain controller makes this determination, it MUST use information in its local NC replicas. The receiving domain controller MAY establish this correspondence and conduct a liveness check by using implementation-specific references between the Computer object in a domain NC, which may or may not be present locally as an NC replica, and the Server object in the configuration NC, which is held locally.<17>

3.3.6 Timer Events

None.

3.3.7 Other Local Events

The lower-layer SMTP MTA receives SMTP messages on its own schedule. This document does not specify the configuration or operation of the SMTP MTA.

4 Protocol Examples

This section illustrates the operation of the DRS Protocol Extensions for SMTP specified in this document by tracing the steps of a single DRS Protocol exchange that is transported over the DRS Protocol Extensions for SMTP. The section describes the SMTP message that carries the DRS request and includes a decoding of the DRS Protocol Extensions for SMTP frame inside that SMTP message. Section 4.4 provides guidance about how to set up a test case in which domain controllers use the DRS Protocol Extensions for SMTP.

4.1 Data Transfer Via SMTP Replication

A single SMTP replication operation consists of four sub-operations as follows. Note that for the purposes of this section, the "client" is the domain controller that is requesting replicated data from a "server."

1. The client sends a request. The DRS engine hands the extension a blob that contains a "get changes" request. The extension performs the higher-layer triggered operation (section 3.2.4), encoding the request blob as a frame. The frame is then handed to the SMTP service. The frame, as an attachment to an SMTP mail message, is sent to the server.

2. The server receives the request. The SMTP service receives the mail message from the client and gives the frame to the extension. The extension performs the message processing operation (section 3.3.5), and then passes the blob to the DRS engine, which processes the request.

3. The server sends the response. After it processes the request, the DRS engine generates another DRS blob, which contains the response. The extension performs the higher-layer triggered operation and then passes the response to the SMTP service. The SMTP service sends the message to the client as an attachment to an SMTP mail message.

4. The client receives the response. The SMTP service receives the mail message from the client and gives the frame to the extension. The extension performs the message processing operation and then passes the blob to the DRS engine, which processes the response.

For the purpose of this example, DC1 (the "server") and DC3 (the "client") exist as described previously, configured for SMTP replication.

4.2 Sample SMTP Message

The following is a sample SMTP message that contains a DRS request message from DC3 to DC1. The fully-qualified domain names of the machines as registered in DNS are d2975006-04cb-4f9d-b797-0c1df78f16d6._msdcs.ddsys7x28.nttest.microsoft.com and daae90dd-b957-4671-a9ae-9fc3c0f2f446._msdcs.ddsys7x28.nttest.microsoft.com, respectively.

The headers for the SMTP message are as follows:

```
From: < IsmService@d2975006-04cb-4f9d-b797-
0c1df78f16d6._msdcs.ddsys7x28.nttest.microsoft.com>
To: <_IsmService@daae90dd-b957-4671-a9ae-
9fc3c0f2f446._msdcs.ddsys7x28.nttest.microsoft.com>
Subject: Intersite message for NTDS Replication: Get changes request for NC
CN=Configuration,DC=ddsys7x28,DC=nttest,DC=microsoft,DC=com from USNs <22749/OU,
22749/PU> with flags 0x300008d0
MIME-Version: 1.0
Content-Type: image/gif
Content-Transfer-Encoding: base64
AAAAAAsAAABIAAAAIA8AAAAAAAAgAgAAAQAAIAcAAAB/+/8fKAAAABwAAAB/+/8fFqqQfy2BLUKd
8GfV2VzRhrgCAAAAAAAAMIIPHAYJKoZIhvcNAQcCoIIPDTCCDwkCAQExDjAMBggqhkiG9w0CBQUA
```

```
MIICMwYJKoZIhvcNAQcBoIICJASCAiABEAg [rest of message elided]
```

4.3 DRS Protocol Extensions for SMTP Transport Frame

The following is the actual mail attachment from the sample SMTP message described in section 4.2 after **base64** decoding.

```
# offset:   value    comments
# MAIL_REP_MSG_V2 header
00000000: 0000 0000 CompressionVersionCaller (0)
00000004: 0b00 0000 ProtocolVersionCaller (11)
00000008: 4800 0000 cbDataOffset (72)
0000000c: 540d 0000 cbDataSize (3412)
00000010: 0000 0000 cbUncompressedDataSize (0)
00000014: d801 0000 cbUnsignedDataSize (472)
00000018: 0100 0020 dwMsgType (10000000000000000000000000000010) b0..31
0000001c: 0700 0000 dwMsgVersion (7)
00000020: 7ffb ff1f dwExtFlags
00000024: 2800 0000 cbExtOffset (40)
# begin DRS_EXTENSIONS extension vector
00000028: 1c00 0000
0000002c: 7ffb ff1f
00000030: e865 14d9
00000034: 5cbd 5c44
00000038: b776 dbcd
0000003c: e1db 2aec
00000040: b001 0000

# padding inserted according to section
CNDJ6nn5us4RjIIAqgBLqQsCAAAACAAAAA4AAABfAFIAZQBmADEANAAwADUANAA3ADMAMgAzAAAA
2.2.4
00000044: 0000 0000
# begin payload data
# offset:   value                                 value as ASCII char
00000048: 3082 0d50 0609 2a86                      0..P..*.
00000050: 4886 f70d 0107 02a0 820d 4130 820d 3d02  H.........A0..=.
00000060: 0101 310e 300c 0608 2a86 4886 f70d 0205  ..1.0...*.H.....
# the payload data is the SMTP-Message-Data-Authenticated
# as defined in section
CNDJ6nn5us4RjIIAqgBLqQsCAAAACAAAAA4AAABfAFIAZQBmADEANAAwADgANQA4ADEANQAxAAAA
3.2.1 above.  It is elided here.
00000d80: 1319 130e a38f be9c b97f b272 14f5 4f85  ...........r..O.
00000d90: 7a89 f8f2 b482 ac4c 4306 3dc5            z......LC.=.
# end payload data
```

4.4 Configuring SMTP Replication

As an aid for implementers who are attempting to set up and test the DRS Protocol Extensions for SMTP, this section provides an example of how to configure SMTP replication between two domain controllers. In the example, replication occurs between two domain controllers that are in the same forest, but in two different sites and domains. Only the configuration and schema partitions (which are common to all domains in the forest) are replicated via SMTP replication.

The relevant information with respect to the two domain controllers is as follows:

- DC1

- Domain controller name: DC1
- Domain: dc=corp,dc=contoso,dc=com
- Site: firstsite

- DC3
  - Domain controller name: DC3
  - Domain: dc=remote,dc=corp,dc=contoso,dc=com
  - Site: remotesite

With the domain controllers configured as described above, create a new site link with SMTP as the transport. Place both domain controllers in that site link. Be sure that any other site links that also contain the two domain controllers have a cost greater than that of the SMTP site link.

5 Security

The following sections specify security considerations for implementers of the DRS Protocol Extensions for SMTP.

5.1 Security Considerations for Implementers

As specified in sections 2.2.3, 2.2.4, and 3, information such as whether the message is a request or a response and which message version is present in both the DRS Protocol Extensions for SMTP headers and inside the serialized DRS message. The fields in the DRS Protocol Extensions for SMTP headers are sent without encryption or authentication, and they are subject to potential snooping and **tampering**. The implementation MUST consider that all header fields are potentially not valid until verified; in particular, the values of **cbDataOffset** and **cbExtOffset** MUST be validated to fall within the extent of the PayloadData. The implementation MUST ensure that buffer under-run, buffer over-run, or integer arithmetic overflow do not occur during decoding and subsequent processing of the frame.<18>

When data is encrypted by this protocol, the **key length** that is used is determined by the length of the key in the recipient's **certificate**. The Domain Controller Replication **certificate** has a key length of 56-bits and the Domain Controller E-mail **certificate** has a key length of 128-bits.<19>

5.2 Index of Security Parameters

Security parameter	Section
Encryption key length 56 Rc4AuxInfo.dwBitLen = 56	3.2.4.3
Encryption key length 128 Rc4AuxInfo.dwBitLen = 128	2.3.2 and 3.2.4.3
Certificate message signing	3.2.4.3
Certificate message sealing	3.2.4.3
Hash function szOID_RSA_MD5	3.2.4.3
(PKCS_7_ASN_ENCODING | CRYPT_ASN_ENCODING)	3.2.4.3
Encryption algorithm szOID_RSA_RC4	3.2.4.3

6 Appendix A: Windows Behavior

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

- Windows 2000
- Windows Server 2003
- Windows Server 2008

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

<1> Section 1.5: The Windows implementation uses the Microsoft SMTP service (SMTPSVC) as the delivery agent. SMTPSVC is an optional component in the Windows Server package. The SMTP service is independent of Microsoft Exchange; it is a stand-alone service with functionality similar to Sendmail.

<2> Section 1.6: **Active Directory (AD)** and the DRS Protocol Extensions for SMTP were first released for Windows 2000. The protocol is not applicable to earlier versions of Windows.

<3> Section 1.7: [MS-DRSR] determines the operating system in use by the peer domain controller, and thus what type of frame to send, by accessing the nTDSDSA object of the forest's configuration NC. Reference "nTDSDSA object" (as specified in [MS-ADTS] section 7.1.1.2.2.1.2.1.1) defines the attributes of this object. The Windows implementation uses the msDs-Behavior-Version attribute (as specified in [MS-ADTS] section 7.1.4.2) to determine the peer operating system.

<4> Section 2.1: Domain controllers that are running Windows automatically initialize the **mailAddress** field. Domain controllers that are running Windows set the local-part of the mail address to the mail recipient named "_IsmService". Domain controllers that are running Windows register a secondary domain for themselves in DNS by using the globally unique identifier (GUID) of their NTDSA object (as specified in [MS-ADTS]) as the most specific label. The format of the **GUID-based DNS name** for a domain controller is as specified in [MS-ADTS]. Domain controllers that are running Windows use this GUID-format DNS alias in the domain portion in their mailAddress.

<5> Section 2.1: A domain controller that is running Windows logs SMTP delivery status notifications (DSNs) for DRS Protocol Extensions for SMTP messages in the Windows Event Log.

<6> Section 2.2.2: All officially released versions of Windows use this protocol version number.

<7> Section 2.2.3: Windows 2000 systems set **cbDataOffset** to 0 in a V1 frame and ignore its contents on receipt. Windows Server 2003 systems set **cbDataOffset** to 32 in a V1 frame. However, Windows Server 2003 systems accept V1 frames with **cbDataOffset** equal to 0, or with **cbDataOffset** equal to 32.

<8> Section 2.3: A domain controller running Windows 2000 Server can process messages sent to it that use either type of **certificate** (Domain Controller Replication or Directory E-mail Replication); however, it will send only requests that use a Domain Controller Replication **certificate**. A domain controller that is running Windows Server 2003 or Windows Server 2008 can process messages sent to it that use either type of **certificate**. When sending requests, a domain controller running Windows Server 2003 or Windows Server 2008 prefers the Directory E-mail Replication **certificates** over the domain controller Replication **certificates**, if both are available. The type of **certificate** that is used when sending a request does not depend on the operating system of the receiving

domain controller. As specified below, the **certificate** that is used to sign the request is sent by the client as part of the request, and is used by the server to encrypt the response.

<9> Section 2.3.1: A computer running Windows Server will use Domain Controller Replication **certificates** that contain the following X.509v3 extensions specific to Windows.

- Certificate template name
- OID = 1.3.6.1.4.1.311.21.6.
- The **certificate** must have the template name extension with the value "DomainController" encoded in BMPSTRING format, as specified in [MS-WCCE] and [UNICODE4.0].

For more information about certificate template names and certificate templates, see [MSFT-TEMPLATES].

<10> Section 2.3.2: A computer running Windows Server will use Directory E-mail Replication **certificates** that contain the following X.509v3 extensions specific to Windows.

- Application Policies (Policy Identifier = Directory E-mail Replication Agent)
- **Certificate** template information
- Template = Directory e-mail Replication(1.3.6.1.4.1.311.21.8.3692315854.1256661383.1690418588.4201632533.1.29)
- Major version number
- Minor version number

<11> Section 2.4.3: The Windows implementation registers in the DNS a secondary host name for the domain controller that is based on the GUID of its nTDSDSA object. This alias is the GUID-based DNS name. The Windows implementation uses the GUID-based DNS name in the domain field of its mailAddress.

<12> Section 3.1.3: The Windows implementation adds dictionary entries for partner domain controllers dynamically during protocol operation.

<13> Section 3.2.4.5: Domain controllers running Windows use the following strings as the format for the Send-Commentary: The DRS layer fills the field "%ws" with the Unicode name of the NC replica, and fills the '%I64d' fields with the unsigned 64-bit quantities taken from USN_VECTOR, as specified in [MS-DRSR] section 5.170.

"Get changes request for NC %ws from USNs <%I64d/OU, %I64d/PU> with flags 0x%x".

"Get changes reply for NC %ws from USNs <%I64d/OU, %I64d/PU> to USNs <%I64d/OU, %I64d/PU>".

<14> Section 3.2.5: The Windows SMTPSVC component returns delivery status notifications (DSNs). Delivery status notifications that indicate a failure are logged.

<15> Section 3.3.5.1: A domain controller running Windows logs SMTP delivery status notifications (DSNs) for DRS Protocol Extensions for SMTP Transport messages in the Windows Event Log.

<16> Section 3.3.5.3: In the case of a Response, the Windows implementation does not add the sender's **certificate** to the map.

<17> Section 3.3.5.7: The Windows implementation uses the value of the serverReferenceBl attribute of the Server object in the configuration NC to establish this correspondence.

<18> Section 5.1: Microsoft's implementation validates the fields in the DRS Protocol Extensions for SMTP headers, but these fields are used only until the DRS message is authenticated, decrypted, and unmarshaled. After that, the data in the headers of the DRS Protocol Extensions for SMTP is no longer needed and is ignored.

<19> Section 5.1: All request messages sent by domain controllers that are running Windows 2000 include a Domain Controller Replication **certificate**; therefore, the response will be encrypted with a 56-bit key. Response data to a domain controller that is running Windows Server 2003 or Windows Server 2008 will be encrypted with either a 56-bit or 128-bit key, depending on whether the domain controller has been configured with a Domain Controller E-mail **certificate** or Domain Controller Replication **certificate**.

7 Index

T

V