[MS-SIPRE]:
Session Initiation Protocol (SIP) Routing Extensions

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
04/04/2008	0.1		Initial Availability
04/25/2008	0.2	Major	Updated based on feedback
06/27/2008	1.0	Major	Updated and revised the technical content.
08/15/2008	1.01	Major	Revised and edited the technical content.
09/12/2008	1.02	Major	Revised and edited the technical content.
12/12/2008	2.0	Major	Updated and revised the technical content.
02/13/2009	2.01	Minor	Revised and edited the technical content.
03/13/2009	2.02	Minor	Revised and edited the technical content.
07/13/2009	2.03	Major	Revised and edited the technical content
08/28/2009	2.04	Editorial	Revised and edited the technical content
11/06/2009	2.05	Minor	Revised and edited the technical content
02/19/2010	2.06	Editorial	Revised and edited the technical content
03/31/2010	2.07	Major	Updated and revised the technical content
04/30/2010	2.08	Editorial	Revised and edited the technical content
06/07/2010	2.09	Editorial	Revised and edited the technical content
06/29/2010	2.10	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.10	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	3.0	Major	Significantly changed the technical content.
11/15/2010	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	3.0	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction 10
1.1 Glossary 10
1.2 References 12
1.2.1 Normative References 12
1.2.2 Informative References 14
1.3 Protocol Overview (Synopsis) 14
1.4 Relationship to Other Protocols 15
1.5 Prerequisites/Preconditions 15
1.6 Applicability Statement 15
1.7 Versioning and Capability Negotiation 15
1.8 Vendor-Extensible Fields 15
1.9 Standards Assignments 15

2 Messages 16
2.1 Transport 16
2.2 Message Syntax 16
2.2.1 SIP URI Parameter Extensions 16
2.2.1.1 SIP URI Parameter Extensions for Record-Route, Path, and Route Header Fields 17
2.2.1.2 SIP URI Parameter Extensions for Contact, Route Header and Request-URI Fields 18
2.2.1.3 SIP URI Parameter Extensions for Contact, Record-Route, Path, Route Header and Request-URI Fields 18
2.2.2 Syntax of Globally Routable User Agent URI 18
2.2.3 Record-Route Header Field Extension 19
2.2.4 Contact Header Field Extensions 19
2.2.5 Via Header Field Extensions 20
2.2.6 From and To Header Field Extensions 21
2.2.7 Location Profile Syntax 21
2.2.7.1 Location Profile Description Element 21
2.2.7.2 Location Profile Rule Element 21
2.2.8 Routing Script Preamble Syntax 22
2.2.8.1 Identification and Version 22
2.2.8.2 Target Element 23
2.2.8.3 List Element 23
2.2.8.4 Flags Element 23
2.2.8.5 Wait time Element 23
2.2.9 Ms-Sensitivity Header Field Syntax 24
2.2.10 Ms-Forking Header Field Syntax 24
2.2.11 Ms-Correlation-Id Header Field Syntax 24
2.2.12 Reason Header Field Extension 24
2.2.13 Content-Disposition Header Field Extension 25
2.2.14 Extensions for Federation and Public IM Connectivity 25
2.2.15 Extensions for Remote Users 25
2.2.16 History-Info Header Field extensions 26
2.2.17 P-Dialog-Recovery-Action Header Field Syntax 26
2.2.18 Option Tag extensions 26
2.2.19 Call Context Syntax 27
2.2.19.1 Id Element 27
2.2.19.2 From Element 27

2.2.19.3 To Element 28
2.2.19.4 Participants Element 29
2.2.19.5 Participant Element 29
2.2.19.6 Date element 30
2.2.19.7 ConversationId element 30
2.2.19.8 DataFormat element 30
2.2.19.9 ContextData element 30
2.2.19.10 Mode element 31
2.2.20 Ms-Call-Info Header Field Syntax 31
2.2.21 P-Agent-On-Behalf-Of Header Field Syntax 32
2.2.22 E911 Call Syntax 32

3 Protocol Details 33
3.1 EPID Mechanism 33
3.1.1 Abstract Data Model 33
3.1.2 Timers 33
3.1.3 Initialization 33
3.1.3.1 User Agent Initialization 34
3.1.4 Higher-Layer Triggered Events 34
3.1.4.1 User Agent Operation 34
3.1.5 Message Processing Events and Sequencing Rules 34
3.1.5.1 User Agent Operation 34
3.1.5.2 SIP Registrar Operation 34
3.1.5.3 SIP Proxy Operation 35
3.1.6 Timer Events 35
3.1.7 Other Local Events 35
3.2 SIP.INSTANCE Mechanism 35
3.2.1 Abstract Data Model 35
3.2.2 Timers 36
3.2.3 Initialization 36
3.2.3.1 User Agent Initialization 36
3.2.4 Higher-Layer Triggered Events 37
3.2.4.1 User Agent Operation 37
3.2.5 Message Processing Events and Sequencing Rules 37
3.2.5.1 SIP Registrar Operation 37
3.2.5.2 SIP Proxy Operation 37
3.2.6 Timer Events 38
3.2.7 Other Local Events 38
3.3 GRUU Mechanism 38
3.3.1 Abstract Data Model 38
3.3.2 Timers 38
3.3.3 Initialization 38
3.3.3.1 User Agent Initialization 38
3.3.4 Higher-Layer Triggered Events 38
3.3.4.1 User Agent Operation 38
3.3.5 Message Processing Events and Sequencing Rules 39
3.3.5.1 SIP Registrar Operation 39
3.3.5.2 SIP Proxy Operation 40
3.3.6 Timer Events 41
3.3.7 Other Local Events 41
3.4 Firewall and Network Address Translation Traversal Aid Extensions 41
3.4.1 Abstract Data Model 42
3.4.2 Timers 42

3.4.3 Initialization 42
3.4.4 Higher-Layer Triggered Events 42
3.4.4.1 User Agent Operation 42
3.4.5 Message Processing Events and Sequencing Rules 43
3.4.5.1 SIP Server (Proxy, Registrar) Operation 43
3.4.6 Timer Events 44
3.4.7 Other Local Events 44
3.5 Extensions for Reliable and Consistent Message Routing Within Redundant Server Network 44
3.5.1 Abstract Data Model 45
3.5.2 Timers 45
3.5.2.1 SIP Proxy Operation 45
3.5.3 Initialization 45
3.5.4 Higher-Layer Triggered Events 46
3.5.5 Message Processing Events and Sequencing Rules 46
3.5.5.1 SIP Proxy Operation 46
3.5.6 Timer Events 46
3.5.7 Other Local Events 46
3.6 Extensions for Dialog State Recovery in Case of Outages in SIP and other Network Elements on the Dialog Path 47
3.6.1 Abstract Data Model 47
3.6.1.1 SIP Proxy Operation 47
3.6.1.2 User Agent Operation 47
3.6.2 Timers 48
3.6.2.1 User Agent Operation 48
3.6.3 Initialization 48
3.6.3.1 User Agent Operation 48
3.6.4 Higher-Layer Triggered Events 48
3.6.4.1 User Agent Operation 48
3.6.5 Message Processing Events and Sequencing Rules 48
3.6.5.1 SIP Proxy Operation 48
3.6.5.2 SIP Registrar Operation 49
3.6.5.3 User Agent Operation 49
3.6.5.3.1 Processing 430 (Flow Failed) Responses 49
3.6.5.3.2 Processing Registration Refresh Responses 50
3.6.5.3.3 Processing Mid- Dialog Refresh Requests 50
3.6.5.3.4 Dialog Recovery Procedure 50
3.6.6 Timer Events 51
3.6.6.1 User Agent Operation 51
3.6.7 Other Local Events 51
3.7 Phone Number Resolution Extensions 51
3.7.1 Abstract Data Model 51
3.7.1.1 User Agent Operation 52
3.7.1.2 SIP Proxy Operation 52
3.7.2 Timers 52
3.7.3 Initialization 52
3.7.3.1 User Agent Operation 52
3.7.4 Higher-Layer Triggered Events 52
3.7.4.1 User Agent Operation 52
3.7.5 Message Processing Events and Sequencing Rules 53
3.7.5.1 SIP Proxy Operation 53
3.7.6 Timer Events 53
3.7.7 Other Local Events 53

3.8 Extensions for Call Processing and Routing Based on Routing Script Preamble and Call Designation Parameters 53
3.8.1 Abstract Data Model 54
3.8.2 Timers 54
3.8.2.1 Registered Endpoints Timer 54
3.8.2.2 Call Forwarding Timer 54
3.8.2.3 Primary User Timer 54
3.8.2.4 Secondary Target Timer 55
3.8.3 Initialization 55
3.8.4 Higher-Layer Triggered Events 55
3.8.5 Message Processing Events and Sequencing Rules 55
3.8.5.1 Call Processing and Routing Elements 55
3.8.5.1.1 Routing Element Name and Version 55
3.8.5.1.2 Routing Element Flags 56
3.8.5.1.3 Routing Element Wait Times 57
3.8.5.1.4 Routing Element Lists 57
3.8.5.2 Incoming INVITE Processing 57
3.8.5.2.1 Ms-Sensitivity Header 58
3.8.5.2.2 Rules for Handling the INVITE 58
3.8.5.2.2.1 Ringing Primary Targets 59
3.8.5.2.2.2 Delegate Ringing 59
3.8.5.2.2.3 Team Ringing 60
3.8.5.2.2.4 Ringing Private Line 60
3.8.5.3 Handling 303 Response 60
3.8.5.4 Handling 605 Response 60
3.8.5.5 Handling 415 Response 60
3.8.5.6 Handling 2XX Responses 61
3.8.5.7 Other Responses 61
3.8.5.8 Generating 199 Response 61
3.8.5.9 1XX Responses Generated 61
3.8.5.10 History-Info Header Field Processing 61
3.8.6 Timer Events 63
3.8.6.1 Registered Endpoint Timer Expiry 63
3.8.6.2 Call Forwarding Timer Expiry 64
3.8.6.3 Primary User Timer Expiry 64
3.8.6.4 Secondary Target Timer Expiry 64
3.8.7 Other Local Events 64
3.9 Extensions for Federation and Public IM Connectivity 64
3.9.1 Abstract Data Model 64
3.9.1.1 ms-source-type parameter 64
3.9.1.2 ms-ep-fqdn parameter 65
3.9.1.3 ms-source-verified-user parameter 65
3.9.1.4 ms-source-network parameter 65
3.9.2 Timers 65
3.9.3 Initialization 66
3.9.4 Higher-Layer Triggered Events 66
3.9.5 Message Processing Events and Sequencing Rules 66
3.9.5.1 Server Behavior 66
3.9.5.2 Client Behavior 66
3.9.6 Timer Events 66
3.9.7 Other Local Events 66
3.10 Extensions for Remote Users 66
3.10.1 Abstract Data Model 66

3.10.2 Timers 67
3.10.3 Initialization 67
3.10.4 Higher-Layer Triggered Events 67
3.10.5 Message Processing Events and Sequencing Rules 67
3.10.5.1 Server Behavior 67
3.10.5.2 Client Behavior 67
3.10.6 Timer Events 67
3.10.7 Other Local Events 67
3.11 Extensions for Logging and Monitoring 67
3.11.1 Abstract Data Model 68
3.11.2 Timers 68
3.11.3 Initialization 68
3.11.4 Higher-Layer Triggered Events 68
3.11.4.1 Client Behavior 68
3.11.5 Message Processing Events and Sequencing Rules 68
3.11.5.1 Client Behavior 68
3.11.5.2 Proxy Behavior 69
3.11.6 Timer Events 69
3.11.7 Other Local Events 69
3.12 Extensions for Call Context 69
3.12.1 Abstract Data Model 69
3.12.2 Timers 69
3.12.3 Initialization 69
3.12.4 Higher-Layer Triggered Events 69
3.12.5 Message Processing Events and Sequencing Rules 70
3.12.5.1 Client Behavior 70
3.12.5.2 Server Behavior 71
3.12.6 Timer Events 71
3.12.7 Other Local Events 71
3.13 Safe Call Transfer Extension 71
3.13.1 Abstract Data Model 71
3.13.2 Timers 72
3.13.3 Initialization 72
3.13.4 Higher-Layer Triggered Events 72
3.13.5 Message Processing Events and Sequencing Rules 72
3.13.6 Timer Events 72
3.13.7 Other Local Events 72
3.14 Extensions for ICE SDP Interworking and Multipart MIME Support 72
3.14.1 Abstract Data Model 72
3.14.2 Timers 72
3.14.3 Initialization 73
3.14.4 Higher-Layer Triggered Events 73
3.14.4.1 Outgoing INVITE 73
3.14.5 Message Processing Events and Sequencing Rules 73
3.14.5.1 Processing INVITE 73
3.14.5.2 Processing 415Response 74
3.14.6 Timer Events 74
3.14.7 Other Local Events 74
3.15 Extensions for Agent Anonymity 74
3.15.1 Abstract Data Model 74
3.15.1.1 Ms-Call-Info Header 74
3.15.1.2 P-Agent-On-Behalf-Of Header 74
3.15.2 Timers 74

3.15.3 Initialization 74
3.15.4 Higher-Layer Triggered Events 75
3.15.5 Message Processing Events and Sequencing Rules 75
3.15.5.1 Server Behavior 75
3.15.6 Timer Events 75
3.15.7 Other Local Events 75
3.16 E911 Message Processing 75
3.16.1 Abstract Data Model 75
3.16.2 Timers 75
3.16.3 Initialization 75
3.16.4 Higher-Layer Triggered Events 75
3.16.5 Message Processing Events and Sequencing Rules 76
3.16.5.1 Client Behavior 76
3.16.5.2 Server Behavior 76
3.16.6 Timer Events 76
3.16.7 Other Local Events 76

4 Protocol Examples 77
4.1 EPID Mechanism 77
4.2 SIP.INSTANCE Mechanism Example 77
4.3 GRUU Mechanism 77
4.4 Firewall and Network Address Translation Traversal Aid Extensions 78
4.5 Reliable and Consistent Message Routing Within Redundant Server Network 79
4.6 Dialog State Recovery 79
4.7 Routing Preamble 81
4.7.1 Blocking Preamble 81
4.7.2 Simultaneous Ring 81
4.7.3 Call Forward 81
4.7.4 Team Ring 82
4.8 History-Info Example 82
4.9 Extension for Federation and Public IM Connectivity 83
4.10 Extension for Remote Users 83
4.11 Extension for Call Context 84
4.12 Multipart MIME 85
4.12.1 Two- level Multipart MIME Example 85
4.12.2 Three- level Multipart MIME Example 86
4.13 Agent Anonymity 89
4.14 E911 INVITE 90

5 Security 92
5.1 Security Considerations for Implementers 92
5.2 Index of Security Parameters 92

6 Appendix A: Full Routing Script Preamble Format 93

7 Appendix B: Full Location Profile Format 96

8 Appendix C: Full Call Context Format 97

9 Appendix D: E911 PIDF Extension Format 98

10 Appendix E: Product Behavior 99

11 Change Tracking 104

12 Index 105

1 Introduction

This document specifies proprietary software application extensions for implementing call routing functionality to the Session Initiation Protocol (SIP). SIP is used by applications to establish, modify, and terminate multimedia sessions or calls.

The extensions discussed in this protocol are used by SIP clients, proxies, and servers.

1.1 Glossary

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

Active Directory
Augmented Backus-Naur Form (ABNF)
authentication
Coordinated Universal Time (UTC)
domain
fully qualified domain name (FQDN)
globally unique identifier (GUID)
Hash-based Message Authentication Code (HMAC)
network address translation (NAT)
security association (SA)
server
SHA-1 hash
Transmission Control Protocol (TCP)
universally unique identifier (UUID)
user agent
UTC (Coordinated Universal Time)

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

200 OK
address-of-record
call
callee
caller
conference
content type
delegate
dialog
endpoint
endpoint identifier (EPID)
federated user
federation
Globally Routable User Agent URI (GRUU)
hash
header field
in-band provisioning
Interactive Connectivity Establishment (ICE)
INVITE
location profile
MIME (Multipurpose Internet Mail Extensions)
NOTIFY
participant
Presence Information Data Format (PIDF)

proxy
public IM connectivity
public switched telephone network (PSTN)
REGISTER
Request-URI
SERVICE
Session Description Protocol (SDP)
Session Initiation Protocol (SIP)
SHA-1
SIP element
SIP message
SIP protocol client
SIP registrar
SIP request
SIP response
SIP transaction
SUBSCRIBE
token
transaction
tuple
URI (Uniform Resource Identifier)
URL (Uniform Resource Locator)
URN (Uniform Resource Name)
user agent client (UAC)
user agent server (UAS)
Web service
XML document
XML schema
XSD

The following terms are specific to this document:

external user: Any user who is located outside the enterprise network boundary, including remote users, federated users, and public instant messaging (IM) users.

federated partner: An enterprise that is trusted for federation (2).

location profile description: An XML document that contains the name of a location profile and a set of translation rules that are associated with that profile.

Media Access Control (MAC) address: A hardware address that uniquely identifies each node of a network.

optimized dialing: A client-side optimization that occurs when users start dialing a phone number. The protocol client compares the collected digit sequence with the translation rules in the location profile and, when a match is detected, applies the rule and sends an INVITE request to the protocol server.

private line: A feature that can be enabled for a voice account and provides an additional, unpublished phone number for a user. A user can choose to disclose the phone number for a private line.

public IM provider: A provider of a public instant messaging (IM) service.

public IM user: An external user who belongs to a public instant messaging (IM) provider.

remote user: A user who has a persistent identity within an enterprise and is connected from outside the enterprise network boundary.

translation rule: A tuple that consists of a regular expression that matches a subset of local numbers and a replacement pattern for it.

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.

[E164] ITU-T, "The International Public Telecommunication Numbering Plan", Recommendation E.164, February 2005, http://www.itu.int/rec/T-REC-E.164/e

Note There is a charge to download the specification.

[FIPS180] Federal Information Processing Standards Publication, "Secure Hash Standard", FIPS PUB 180-1, April 1995, http://www.itl.nist.gov/fipspubs/fip180-1.htm

[FIPS198a] National Institute of Standards and Technology, "The Keyed-Hash Message Authentication Code (HMAC)", FIPS PUB 198, March 2002, http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf

[IETFDRAFT-ICENAT-06] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", draft-ietf-mmusic-ice-06, October 2005, http://tools.ietf.org/html/draft-ietf-mmusic-ice-06

[IETFDRAFT-ICENAT-19] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", draft-ietf-mmusic-ice-19, October 2007, http://tools.ietf.org/html/draft-ietf-mmusic-ice-19

[IETFDRAFT-MCICSIP-11] Jennings, C., Ed. and Mahy, R., Ed., "Managing Client Initiated Connections in the Session Initiation Protocol (SIP)", draft-ietf-sip-outbound-11, November 2007, http://tools.ietf.org/id/draft-ietf-sip-outbound-11.txt

[IETFDRAFT-OUGRUAUSIP-10] Rosenberg, J., "Obtaining and Using Globally Routable User Agent (UA) URIs (GRUU) in the Session Initiation Protocol (SIP)", draft-ietf-sip-gruu-10, July 2006, http://tools.ietf.org/id/draft-ietf-sip-gruu-10.txt

[IETFDRAFT-RCDPR-303-01] Ramanathan, R., Parameswar, S., and Vakil, M., "Response Code for Dynamic Proxy Redirect", draft-rajesh-sipping-303-01, February 2007, http://tools.ietf.org/id/draft-rajesh-sipping-303-01.txt

[IETFDRAFT-RCITD-199-01] Holmberg, C., "Response Code for Indication of Terminated Dialog", draft-ietf-sip-199-01.txt, August 2008, http://tools.ietf.org/id/draft-ietf-sip-199-01.txt

[IETFDRAFT-SF-605-01] Ramanathan, R., Vakil, M., and Parameswar, S., "Serial Forking and 605", draft-rajesh-sipping-605-01, March 2007, http://tools.ietf.org/id/draft-rajesh-sipping-605-01.txt

[IETFDRAFT-SIPSOAP-00] Deason, N., "SIP and SOAP", draft-deason-sip-soap-00, June 30 2000, http://www.softarmor.com/wgdb/docs/draft-deason-sip-soap-00.txt

[MC-RegEx] Microsoft Corporation, "Regular Expression Language Elements", http://msdn.microsoft.com/en-us/library/az24scfc(VS.80).aspx

[MS-CONFBAS] Microsoft Corporation, "Centralized Conference Control Protocol: Basic Architecture and Signaling Specification", June 2008.

[MS-E911WS] Microsoft Corporation, "Web Service for E911 Support Protocol Specification", March 2010.

[MS-OCER] Microsoft Corporation, "Client Error Reporting Protocol Specification", June 2008.

[MS-PRES] Microsoft Corporation, "Presence Protocol Specification", June 2008.

[MS-SDPEXT] Microsoft Corporation, "Session Description Protocol (SDP) Version 2.0 Protocol Extensions", June 2008.

[MS-SIPREGE] Microsoft Corporation, "Session Initiation Protocol (SIP) Registration Extensions", June 2008.

[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

[RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997, http://www.ietf.org/rfc/rfc2141.txt

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E., "SIP: Session Initiation Protocol", RFC 3261, June 2002, http://www.ietf.org/rfc/rfc3261.txt

[RFC3265] Roach, A. B., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002, http://www.ietf.org/rfc/rfc3265.txt

[RFC3326] Schulzrinne H., et al., "The Reason Header Field for the Session Initiation Protocol (SIP)", RFC 3326, December 2002, http://www.ietf.org/rfc/rfc3326.txt

[RFC3327] Willis, D., et al., "Session Initiation Protocol (SIP) Extension Header Field for Registering Non-Adjacent Contacts", RFC 3327, December 2002, http://www.ietf.org/rfc/rfc3327.txt

[RFC3548] Josefsson, S., Ed., "The Base16, Base32, and Base64 Data Encodings", RFC 3548, July 2003, http://www.ietf.org/rfc/rfc3548.txt

[RFC3863] Sugano, H., Fujimoto, S., Klyne, G., et al., "Presence Information Data Format (PIDF)", RFC 3863, August 2004, http://www.ietf.org/rfc/rfc3863.txt

[RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, December 2004, http://www.ietf.org/rfc/rfc3966.txt

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

[RFC4028] Donovan, S., Rosenberg, J., "Session Timers in the Session Initiation Protocol (SIP)", RFC 4028, April 2005, http://www.ietf.org/rfc/rfc4028.txt

[RFC4119] Peterson, J., "A Presence-based GEOPRIV Location Object Format", December 2005, http://www.ietf.org/rfc/rfc4119.txt

[RFC4122] Leach, P., Mealling, M., and Salz, R., "A Universally Unique Identifier (UUID) URN Namespace", RFC 4122, July 2005, http://www.ietf.org/rfc/rfc4122.txt

[RFC4235] Rosenberg, J. and Schulzrinne, H., "An INVITE-Initiated Dialog Event Package for the Session Initiation Protocol (SIP)", November 2005, ftp://ftp.rfc-editor.org/in-notes/rfc4235.txt

[RFC4244] Barnes, M., "An Extension to the Session Initiation Protocol (SIP) for Request History Information", RFC 4244, November 2005, http://www.ietf.org/rfc/rfc4244.txt

[RFC5139] Thomson, M. and Winterbottom, J., "Revised Civic Location Format for Presence Information Data Format Location Object (PIDF-LO)", February 2008, http://www.rfc-editor.org/rfc/rfc5139.txt

1.2.2 Informative References

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

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary", June 2008.

[XML10] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Third Edition)", February 2004, http://www.w3.org/TR/REC-xml

[XMLNS] World Wide Web Consortium, "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation 8 December 2009, http://www.w3.org/TR/REC-xml-names/

[XMLSCHEMA0] Fallside, D., Ed. and Walmsley, P., Ed., "XML Schema Part 0: Primer, Second Edition", W3C Recommendation, October 2004, http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/

1.3 Protocol Overview (Synopsis)

This document discusses **Session Initiation Protocol (SIP)** extensions that are used in this protocol architecture.

Endpoint (5) identification extensions have been designed to help route **calls** within SIP topologies with more than one protocol client endpoint (5). They provide unique identities and addresses to multiple communication endpoints (5) representing the same user or service and allow the **servers (2)** and other protocol clients to identify a specific endpoint (5) that initiated communication and to route calls to a specific endpoint (5). These extensions are described in detail in section 3.1 through section 3.3.

Extensions to SIP **URI (Uniform Resource Identifier)** and **header field** syntax ensure that messages within **SIP transactions** are processed consistently and reliably delivered within SIP topologies with multiple redundant servers (2). These extensions also resolve addressing issues in network topologies where the protocol client and server (2) are separated by a firewall or a **network address translation (NAT)** device. These extensions are described in detail in section 3.4, section 3.5, and section 3.6.

The phone number resolution extensions provide a way for **SIP elements** to resolve partially specified local phone numbers to a number that allows the server (2) to route the call to a unique enterprise user or forms a unique number in a public telephone network, as defined by International Telecommunications Union Recommendation. These extensions are described in detail in section 3.7.

The routing script preamble and call designation extensions provide a way for a protocol client to describe a set of endpoints (5) to receive calls targeted at the user as well as define parameters for

routing action taken by the server (2) when processing these calls. These extensions are described in section 3.8.

The extensions for **federation (2)** and **public IM connectivity** provide a way to inform protocol clients whether the **SIP message** is from a **remote user**, **federated user** or a **public IM user**. The extensions for remote users provide a way to inform a protocol client that it is connected to the server (2) from outside the enterprise network boundary. These extensions are described in section 3.9 and section 3.10.

Section 3.11 describes an extension that provides a way to correlate multiple SIP **dialogs** for logging and monitoring purposes.

The extensions to create notes and other context information related to a given call and send them to another party during **transaction** establishment are described in section 3.12, section 3.13, and section 3.14.

The extensions to provide anonymity to a call are described in section 3.15.

1.4 Relationship to Other Protocols

This protocol defines an **XML schema** that supports various extensions specified in this protocol. For more information about XML, see [XML10], [XMLNS], and [XMLSCHEMA0].

This protocol is invoked as an extension of SIP. This protocol incorporates SIP protocols.

1.5 Prerequisites/Preconditions

This protocol assumes that both the **SIP protocol clients** and the server (2) support SIP. The prerequisites for this protocol and the SIP prerequisites are identical.

1.6 Applicability Statement

This protocol is applicable when both the SIP protocol clients and the server (2) support SIP and intend to use one or more of the enhancements offered by this protocol.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

Standard SIP extension mechanisms can be used by vendors as needed.

1.9 Standards Assignments

None.

2 Messages

The following sections specify how this protocol messages are transported and specify this protocol message syntax.

2.1 Transport

This protocol does not introduce a new transport to exchange messages and is capable of being used with any transport used by SIP.

2.2 Message Syntax

This protocol relies on the SIP message format, as specified in [RFC3261] section 7, and extends definitions of URI and header field parameters by adding new values for parameter and header field names, as well as their corresponding values. This protocol defines new message body types in addition to those defined in [RFC3261]. All of the message syntax specified in this protocol is described in both prose and an **Augmented Backus-Naur Form (ABNF)**.

2.2.1 SIP URI Parameter Extensions

This protocol defines several new URI parameter names and values. The original ABNF for **uri-parameter** in [RFC3261] section 25 is extended as follows:

```
uri-parameter = transport-param / user-param / method-param
                / ttl-param / maddr-param / lr-param
                / opaque-param
                / gruu-param
                / grid-param
                / received-param
                / ms-opaque-param
                / ms-received-cid-param
                / ms-route-sig-param
                / ms-key-info-param
                / ms-identity-param
                / ms-fe-param
                / ms-role-rs-to-param
                / ms-role-rs-from-param
                / ms-ent-dest-param
                / default-param
                / phone-context-param
                / other-param
opaque-param = "opaque=" opaque-value
opaque-value = ua-opaque-val
               / app-voicemail-opaque-val
               / app-locationprofile-opaque-val
               / app-conf-opaque-val
               / server-opaque-val
               / state-opaque-val
               / pvalue
ua-opaque-val = "user:epid:" encoded-uuid-val
app-voicemail-opaque-val = "app:voicemail"
app-locationprofile-opaque-val = "app:locationprofile:get"
app-conf-opaque-val = "app:conf:" conf-entity-val ":id:"
                      encoded-conf-id-val
server-opaque-val = "srvr:" server-type-val ":"
                    encoded-server-instance-val
```

```
state-opaque-val = "state:" pvalue
encoded-uuid-val = 1*paramchar
conf-entity-val = "focus" / "audio-video" / "chat"
                  / "meeting" / "phone-conf"
encoded-conf-id-val = 1*paramchar
server-type-val = "HomeServer" / "MediationServer" / "MRAS" / "QoSM"
encoded-server-instance-val = 1*paramchar
gruu-param = "gruu"
grid-param = "grid" ["=" pvalue]
received-param = "received=" (IPv4address / IPv6address)
ms-opaque-param = "ms-opaque=" pvalue
ms-received-cid-param = "ms-received-cid=" pvalue
ms-route-sig-param = "ms-route-sig=" pvalue
ms-key-info-param = "ms-key-info=" pvalue
ms-fe-param = "ms-fe=" pvalue
ms-role-rs-to-param = "ms-role-rs-to"
ms-role-rs-from-param = "ms-role-rs-from"
ms-ent-dest-param = "ms-ent-dest"
ms-identity-param = "ms-identity=" pvalue
default-param = "default"
phone-context-param = "phone-context=" descriptor
location-name = domainname / global-number-digits
```

State-opaque-val follows the product behavior in this endnote<1>.

paramchar, **pvalue**, **IPv4address**, and **IPv6address** are defined in [RFC3261] section 25.

domainname and **global-number-digits** are defined in [RFC3966] section 3.

2.2.1.1 SIP URI Parameter Extensions for Record-Route, Path, and Route Header Fields

The following SIP URI parameter extensions can be used in URIs inserted by SIP **proxies** into the **Record-Route** header fields of any message described in [RFC3261] section 16, or into the **Path** header field of the **REGISTER** request described in [RFC3327] section 5.

- **ms-opaque-param**
- **ms-route-sig-param**
- **ms-key-info-param**
- **ms-identity-param**
- **ms-fe-param**
- **ms-role-rs-to-param**
- **ms-role-rs-from-param**
- **ms-ent-dest-param**

These extensions can then appear in the **Route** header field. As specified in [RFC3261] section 12, the list of URIs in the **Record-Route** header fields, taken in order with all URI parameters, is stored in the dialog state. This list of URIs is also stored in the **Route** header fields of every **SIP request** in the SIP dialog. Additionally, as specified in [RFC3327] section 5, the content of the **Path** header

fields is stored by the registrar and then used by the SIP proxy that is responsible for the **domain** of the request destination to populate **Route** header fields.

2.2.1.2 SIP URI Parameter Extensions for Contact, Route Header and Request-URI Fields

The following SIP URI parameter extensions can be inserted by SIP elements into the **URI** of the **Contact** header field:

- **opaque-param**
- **gruu-param**
- **grid-param**
- **ms-fe-param**
- **ms-opaque-param**

These extensions can then appear in the **Request-URI** field because, as specified in [RFC3261] section 12, the URI in the **Contact** header field is stored in the dialog state and is included as the **Request-URI** field of each SIP request within a dialog. Also, if the **Contact** header field is used in the REGISTER request, as described in [RFC3261] section 10, the **Contact** header field can be stored by the SIP location service and then used by the SIP proxy, as described in [RFC3261] section 16, to populate the **Request-URI** field. In addition, as described in [RFC3261] section 16.4, if the SIP element sending the request is a strict router, it can place the URI from the **Contact** header field into the **Route** header field.

2.2.1.3 SIP URI Parameter Extensions for Contact, Record-Route, Path, Route Header and Request-URI Fields

The following SIP **URI** parameter extensions can be inserted by the SIP proxy into the **URI** of the **Contact**, **Record-Route**, or **Path** header fields created by the upstream SIP element:

- **received-param**
- **ms-received-cid-param**

If inserted into the **URI** of **Record-Route** or **Path** header fields, these parameter extensions can appear in the **Route** header field, as described in section 2.2.1.1. If inserted into the **URI** of the **Contact** header field, these extensions can appear in the Request-URI field, as described in section 2.2.1.2.

2.2.2 Syntax of Globally Routable User Agent URI

This protocol defines several **Globally Routable User Agent URI (GRUU)** syntax forms for the **SIP registrar** that is compliant with this protocol. These syntax forms are based on SIP URI parameter extensions described in section 2.2.1 and are intended to satisfy the requirements for the GRUU syntax that is defined in [IETFDRAFT-OUGRUAUSIP-10] section 6.

```
user-agent-gruu = "sip:" address-of-record
                  ";gruu"
                  ";opaque=" ua-opaque-val
voice-mail-gruu = "sip:" address-of-record
                  ";gruu"
                  ";opaque=" app-voicemail-opaque-val
```

```
location-profile-gruu = "sip:" address-of-record
                  ";gruu"
                  ";opaque=" app-locationprofile-opaque-val
                  (;default-param / phone-context-param)
conf-endpoint-gruu =sip:" address-of-record
                ";gruu"
                ";opaque=" app-conf-opaque-val
server-instance-gruu = "sip:" server-fqdn "@" domain-fqdn
                 ";gruu"
                 ";opaque=" server-opaque-val
address-of-record = userinfo host
server-fqdn = host
domain-fqdn = host
```

ua-opaque-val, **app-voicemail-opaque-val**, **app-conf-opaque-val**, **server-opaque-val**, and **app-locationprofile-opaque-val** are defined in section 2.2.1.

userinfo, **host**, and **token** are defined in [RFC3261] section 25.1.

2.2.3 Record-Route Header Field Extension

This protocol defines a new **Record-Route** header field parameter and its value. The original ABNF for the **Record-Route** header field in [RFC3261] section 25 is extended as follows:

```
rr-param = rr-p-ms-rrsig
          / generic-param
rr-p-ms-rrsig = "ms-rrsig=" pvalue
```

pvalue is defined in [RFC3261] section 25.

2.2.4 Contact Header Field Extensions

This protocol defines a new **Contact** header field parameter and its value. The original ABNF for the **Contact** header field in [RFC3261] section 25 is extended as follows:

```
contact-params = c-p-q / c-p-expires
                / c-p-proxy
                / contact-extension
c-p-proxy = "proxy=" "replace"
```

In addition to the extension defined in this protocol, this protocol uses the **sip.instance** media feature tag introduced in [IETFDRAFT-MCICSIP-11] section 12.5, with syntax defined in [IETFDRAFT-MCICSIP-11] section 10, for use as the **Contact** header field parameter. The syntax for the **+sip.instance** parameter in the **Contact** header field from [IETFDRAFT-MCICSIP-11] section 10 is as follows:

```
c-p-instance   =  "+sip.instance" EQUAL
                  LDQUOT "<" instance-val ">" RDQUOT
instance-val   = *uric ; defined in [RFC3986]
```

Because this protocol requires that only **universally unique identifier (UUID) URN (Uniform Resource Name)** be used as the **+sip.instance** parameter value, the **instance-val** definition is restricted to the **UUID** URN syntax (**UUID-URN**), as defined in [RFC4122] and [RFC2141].

The URN definition from [RFC2141], as applicable to the UUID URN defined in [RFC4122] is as follows:

```
UUID-URN = "urn:" UUID-NID ":" UUID-NSS
```

The UUID namespace identifier syntax from [RFC4122] is as follows:

```
UUID-NID = "uuid"
```

The UUID namespace specific string syntax from [RFC4122] is as follows:

```
UUID-NSS               = time-low "-" time-mid "-"
                         time-high-and-version "-"
                         clock-seq-and-reserved
                         clock-seq-low "-" node
time-low               = 4hexOctet
time-mid               = 2hexOctet
time-high-and-version  = 2hexOctet
clock-seq-and-reserved = hexOctet
clock-seq-low          = hexOctet
node                   = 6hexOctet
hexOctet               = hexDigit hexDigit
hexDigit =
            "0" / "1" / "2" / "3" / "4" / "5" / "6" / "7" / "8" / "9" /
            "a" / "b" / "c" / "d" / "e" / "f" /
            "A" / "B" / "C" / "D" / "E" / "F"
```

Also, this protocol uses the **sip.rendering** media feature tag defined in [RFC4235] section 5.2. This, in conjunction with procedures described for music-on-hold specified in [MS-SDPEXT] section 3.1.5.28, can be used by SIP **user agents** to signal that the music-on-hold feature is being invoked by including it in the SIP request that initiates music-on-hold.<2>

2.2.5 Via Header Field Extensions

This protocol defines new **Via** header field parameters and their values. The original ABNF for the **Via** header field in [RFC3261] section 25 is extended as follows:

```
via-params = via-ttl / via-maddr / via-received /via-branch
             / via-branched
             / via-ms-internal-info
             / via-ms-received-port
             / via-ms-received-cid
             / via-extension
via-branched = "branched=" ("TRUE" / "FALSE")
via-ms-internal-info = "ms-internal-info=" quoted-string
via-ms-received-port = "ms-received-port=" port
via-ms-received-cid  = "ms-received-cid=" token
```

token, **quoted-string**, and **port** are defined in [RFC3261] section 25.1.

2.2.6 From and To Header Field Extensions

This protocol defines a new **From** and **To** header field parameter and its value. The original ABNF for the **From** and **To** header fields in [RFC3261] section 25 is extended as follows:

```
from-param = tag-param
             / epid-param
             / generic-param
to-param = tag-param
             / epid-param
             / generic-param
epid-param = "epid=" epid-param-value
epid-param-value = 1*16 tokenchar
tokenchar = (alphanum / "-" / "." / "!" / "%" / "*"
             / "_" / "+" / "`" / "'" / "~" )
```

alphanum is defined in [RFC3261] section 25.

2.2.7 Location Profile Syntax

This section describes the **location profiles** syntax and associated **translation rules** used by the SIP elements to resolve partially specified local phone numbers. The **XML documents** with **location profile descriptions** are delivered as **application/ms-location-profile-definition+xml** content in the body of responses to the SIP **SERVICE** requests, as described in [IETFDRAFT-SIPSOAP-00]. The complete schema is defined in section 7.

2.2.7.1 Location Profile Description Element

Each location profile description MUST include a **Name** element and one or more **Rule** elements. The **Name** element MUST be a string suitable for use as a **phonecontext** parameter in the **tel URI**, as defined in [RFC3966] section 3. As specified in [RFC3966], the content of the **tel URI** can also be used as the user portion of a SIP URI.

The location profile description can also contain the following elements:

ExternalAccessPrefix: Element that contains the prefix string that SHOULD be added when dialing external phone numbers.<3>

OptimizeDeviceDialing: Element that, if **true**, indicates to the endpoint (5) using this location profile that the endpoint (5) can do **optimized dialing**. If the value of this element is **false**, the endpoint (5) cannot optimize device dialing when using this location profile.<4>

```
  <xsd:complexType name="LocationProfileDescriptionType">
    <xsd:sequence>
      <xsd:element ref="Name" minOccurs="1" maxOccurs="1"/>
      <xsd:element name="Rule" type="RuleType" minOccurs="1" maxOccurs="unbounded"/>
<xsd:element ref="ExternalAccessPrefix" minOccurs="0" maxOccurs="1"/>
<xsd:element ref="OptimizeDeviceDialing" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
  </xsd:complexType>
```

2.2.7.2 Location Profile Rule Element

Each location profile **Rule** element MUST include **Pattern** and **Translation** elements. The **Pattern** element is a regular expression that uses the regular expression syntax defined in [MC-RegEx]. The

Translation element is a replacement pattern that uses the replacement pattern syntax defined in [MC-RegEx].

The **Rule** element can also contain the following elements:

InternalEnterpriseExtension: Element that, if **true**, indicates that the phone number obtained as a result of applying this rule corresponds to an internal enterprise number. If the value of this element is **false**, the phone number obtained as a result of applying this rule cannot be assumed to be an internal enterprise number.<5>

ApplicableForDeviceDialing: Element that, if **true**, indicates that the device can use the rule for optimized dialing. If the value of this element is **false**, the device cannot use this rule for optimized dialing.<6>

```
  <xsd:complexType name="RuleType">
    <xsd:sequence>
      <xsd:element name="Pattern"   type="xsd:string"/>
      <xsd:element name="Translation" type="xsd:string"/>
<xsd:element name="InternalEnterpriseExtension" type="xsd:boolean" minOccurs="0"/>
    <xsd:element name="ApplicableForDeviceDialing" type="xsd:boolean" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
```

2.2.8 Routing Script Preamble Syntax

This section specifies the syntax of the routing preamble published by the protocol client in the routing category. The complete schema is defined in section 6.

```
  <xs:complexType name="routing-type">
    <xs:annotation>
      <xs:documentation>The name and version attributes are both mandatory.
</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="preamble" type="tns:preamble-type" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" />
    <xs:attribute name="version" type="xs:integer" />
    <xs:attribute name="minSupportedClientVersion" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:element name="routing" type="tns:routing-type" />
```

The preamble provides the data used by the server (2) while routing audio calls sent to the protocol client. The preamble MUST contain the identification attributes specified in section 2.2.8.1, and can contain additional elements specified in sections 2.2.8.1 through 2.2.8.5.

If the value of the **version** attribute is 1, the **minSupportedClientVersion** attribute SHOULD NOT be present.<7>

The **minSupportedClientVersion** attribute, if present, SHOULD be ignored. In addition any unknown element or attribute SHOULD be ignored.<8>

2.2.8.1 Identification and Version

The **name** attribute is a string value that provides a scope for the **version** attribute.

2.2.8.2 Target Element

The **target** element specifies a target the call can be routed to. The **uri** attribute, if present, SHOULD be a valid SIP URI. At least one of the **uri** or **application** attributes MUST be present.

```
<xs:complexType name="target-type">
    <xs:annotation>
      <xs:documentation>At least one of uri or application attributes are
required.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="uri" type="xs:string" use="optional" />
    <xs:attribute name="application" type="xs:string" use="optional" />
  </xs:complexType>
```

2.2.8.3 List Element

The **list** element defines a list of **target** elements that are grouped together. Each **list** element SHOULD have a unique **name** attribute and can contain zero or more **target** elements.

```
<xs:complexType name="list-type">
    <xs:complexContent>
      <xs:extension base="tns:preamble-member-base-type">
        <xs:sequence>
          <xs:element name="target" type="tns:target-type" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

2.2.8.4 Flags Element

The **flags** element defines flags that can be used by the script installed on the server (2). Each **flags** element MUST have a **name** attribute that is unique among all **flags** elements defined in the preamble.

```
<xs:complexType name="flags-type">
    <xs:complexContent>
      <xs:extension base="tns:preamble-member-base-type">
        <xs:attribute name="value" type="xs:string" use="required" />
      </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

2.2.8.5 Wait time Element

The **wait** element defines an amount of time in seconds that is referenced by the server (2) while executing the forwarding rules defined by the protocol client. The **name** attribute MUST be unique among all **wait** elements. The **seconds** attribute value SHOULD be between 0 and 1,200 seconds.

```
<xs:complexType name="wait-type">
    <xs:complexContent>
      <xs:extension base="tns:preamble-member-base-type">
```

```
        <xs:attribute name="seconds" type="xs:nonNegativeInteger" use="required" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

2.2.9 Ms-Sensitivity Header Field Syntax

This protocol defines a header field called **Ms-Sensitivity** to indicate if a call can be directed to another person or diverted to another device representing the same person. The ABNF for this header is as follows:

```
Ms-Sensitivity = "Ms-Sensitivity" HCOLON ("normal" / "private" /
                 "normal-no-diversion" / "private-no-diversion")
```

A sensitivity of "normal" MUST be assumed if the **Ms-Sensitivity** header field is not present. If the header field contains a value other than one of those specified or appears more than once, a 400 response SHOULD be returned.

2.2.10 Ms-Forking Header Field Syntax

This protocol defines a header field called **Ms-Forking**. The **Ms-Forking** header field indicates to the endpoint (5) that sent the **INVITE** that a proxy is likely to perform either parallel or serial forking, or both.

```
Ms-Forking = "Ms-Forking" HCOLON "Active"
```

Endpoints (5) can use this information to limit when they accept early media. The **Ms-Forking** header field MUST appear only in 1XX responses.

2.2.11 Ms-Correlation-Id Header Field Syntax

This protocol defines a header field called **Ms-Correlation-Id**. The **Ms-Correlation-Id** header field is used to indicate that multiple SIP dialogs are correlated. This correlation is only used for diagnostic and monitoring purposes. It does not affect the routing behavior of the SIP proxy or endpoints (5).

```
Ms-Correlation-Id = "Ms-Correlation-Id" HCOLON  UUID
```

UUID is defined by [RFC4122].

2.2.12 Reason Header Field Extension

This protocol defines a **Reason** header field parameter. The ABNF from [RFC3326] is extended as follows:

```
Reason            =  "Reason" HCOLON reason-value *(COMMA reason-value)
reason-value      =  protocol *(SEMI reason-params)
protocol          =  "SIP" / "Q.850" / token
reason-params     =  protocol-cause / reason-text
                      / ms-acceptedby-param
                      / reason-extension
```

```
ms-acceptedby-param = "ms-acceptedby=" SIPURI
```

SIPURI is defined in [RFC3261] section 25.

2.2.13 Content-Disposition Header Field Extension

This section follows the product behavior described in endnote <9>.

This protocol defines a **Content-Disposition** header field parameter. The ABNF syntax defined in [RFC3261] is extended as follows:

```
Content-Disposition    = "Content-Disposition" HCOLON disp-type
                             *(SEMI disp-param)
disp-type               =  "render" / "session" / "icon" / "alert"
                             / disp-extension-token
disp-param              =  handling-param / ms-proxyfallback-param
                             / generic-param
ms-proxyfallback-param  =  "ms-proxy-2007fallback"
```

2.2.14 Extensions for Federation and Public IM Connectivity

This protocol defines an **ms-edge-proxy-message-trust** header field. This header field can be added by the SIP proxy to any incoming SIP request or **SIP response** from an **external user** to inform the destination protocol client whether the SIP message originates from a remote user, a **federated partner,** or a **public IM provider**. This header field MUST NOT be added by the protocol client.

The ABNF for the **ms-edge-proxy-message-trust** header field is specified as follows:

```
"ms-edge-proxy-message-trust" HCOLON sourcetype-param SEMI epfqdn-param SEMI userverify-param
SEMI sourcenetwork-param

sourcetype-param = "ms-source-type="  ("AuthorizedServer" /
 "AutoFederation" /  "DirectPartner" /
"EdgeProxyGenerated" /  "InternetUser")

epfqdn-param = "ms-ep-fqdn=" pvalue

userverify-param = "ms-source-verified-user="  ( "verified" / "unverified")

sourcenetwork-param = "ms-source-network=" ("federation" / "publiccloud")
```

HCOLON, **SEMI**, and **pvalue** are defined in [RFC3261] section 25.

Details regarding the header field parameters and their values are specified in section 3.9. Example usage for this header field is covered in section 4.9.

2.2.15 Extensions for Remote Users

This protocol defines an **ms-user-logon-data** header field. This header field can be added by the SIP proxy to any outgoing SIP request or response to remote users to inform the destination protocol client that it is connected from outside the enterprise network boundary. A protocol client MUST NOT add the **ms-user-logon-data** header field in any SIP messages sent to the server (2).

The ABNF for the **ms-user-logon-data** header field is specified as follows:

```
"ms-user-logon-data" HCOLON "RemoteUser"
```

HCOLON is defined in [RFC3261] section 25.

Details regarding the header field parameters and their values are specified in section 3.10. Example use of this header field is covered in section 4.10.

2.2.16 History-Info Header Field extensions

This section follows the product behavior described in endnote <10>.

This protocol defines a **History-Info** header field parameter. The ABNF from [RFC4244] is extended as follows:

```
History-Info           = "History-Info" HCOLON
                         hi-entry *(COMMA hi-entry)
hi-entry               = hi-targeted-to-uri *( SEMI hi-param )
hi-targeted-to-uri     = name-addr
hi-param               = hi-index / hi-ms-retarget-reason / hi-ms-line-type
                         / hi-extension
hi-index               = "index" EQUAL 1*DIGIT *(DOT 1*DIGIT)
hi-ms-retarget-reason  = "ms-retarget-reason" EQUAL
                         hi-retarget-reason-val
hi-retarget-reason-val = "forwarding" / "team-call"
                         / "delegation" / token
hi-ms-line-type        = "ms-line-type" EQUAL hi-line-type-val
hi-line-type-val       = "private" / token
hi-extension           = generic-param
```

2.2.17 P-Dialog-Recovery-Action Header Field Syntax

This section follows the product behavior described in endnote <11>.

This protocol defines a **P-Dialog-Recovery-Action** header field. This header can be added by the SIP proxy to a 430 Flow Failed response.

The ABNF for the **P-Dialog-Recovery-Action** header field is as follows:

```
P-Dialog-Recovery-Action = "P-Dialog-Recovery-Action" HCOLON
                             pdr-action *(COMMA pdr-action)
pdr-action               = "Registration-Route-Set-Update"
                           / "Dialog-Route-Set-Update"
                           / "Wait-For-Session-Update"
                           / pdr-action-extension
pdr-action-extension     = token
```

2.2.18 Option Tag extensions

This section follows the product behavior described in endnote <12>.

This protocol defines option tags for use in the **Supported** header field. The new tags extend the set of option tags defined in [RFC3261].

Ms-Dialog-Route-Set-Update: Option tag for support of the dialog route set recovery extension. Inclusion of this tag in the **Supported** header field of the request indicates that the user agent can perform dialog route set recovery, as described in section 3.6.

Ms-Safe-Transfer: Option tag for support of call transfer via SIP REFER request. Inclusion of this tag in the **Supported** header field of the request indicates that the user agent can copy parameters from the **Refer-To** header field URI of the REFER request to the INVITE request, as described in section 3.13.

2.2.19 Call Context Syntax

This section follows the product behavior described in endnote <13>.

This section describes the call context syntax that can be used by SIP elements to convey notes about the current call or the call being transferred. The call context description is delivered as **application/ms-conversation-context+xml** content in the body of a SIP INVITE request to initiate a new call.

```
  <xs:complexType name="XmlConvContextType" >
    <xs:sequence>
      <xs:element name="id" type="xs:token"  minOccurs="1" maxOccurs="1/>
      <xs:element name="from" type="tns:XmlConvContextParticipantType" minOccurs="1"
maxOccurs="1"/>
      <xs:element name="to" type="tns:XmlConvContextParticipantType" minOccurs="1"
maxOccurs="1"/>
      <xs:element name="participants" type="tns:XmlConvContextParticipantCollectionType"
minOccurs="1" maxOccurs="1" />
      <xs:element name="date" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="mode" type="xs:token" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="conversationId" type="xs:token" minOccurs="1" maxOccurs="1"/>
      <xs:element name="dataFormat" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="contextData" type="xs:string" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
```

The complete schema is defined in section 8.

The call context **content type** provides notes about the current call from a server (2) to a protocol client. The call context MUST contain the elements specified in sections 2.2.19.1 through 2.2.19.9, and can contain additional elements specified in section 2.2.19.10.

2.2.19.1 Id Element

The **id** element defines a unique identifier generated by the authoring device, either the protocol client or the server (2), of the call context data to differentiate one set of call context data from another across all call context generated by a given author. The **id** element MUST be unique among all call context data created by a given author and appear only once in the call context data.

```
<xs:element name="id" type="xs:token"  minOccurs="1" maxOccurs="1"/>
```

2.2.19.2 From Element

The **from** element describes the author of the call context data that is being conveyed. The **from** element MUST be present in the call context data and appear only once.

```
<xs:element name="from" type="tns:XmlConvContextParticipantType" minOccurs="1" maxOccurs="1/>

  <xs:complexType name="XmlConvContextParticipantType">
    <xs:sequence>
      <xs:element name="uri" type="xs:string" minOccurs="1" maxOccurs="1/>
      <xs:element name="displayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfUri" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfDisplayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
```

The **from** element MUST contain a **uri** element representing the author of the call context data, such as sip:alice@contoso.com. The **from** element can also contain the following elements:

- **displayName**
- **onBehalfUri**
- **onBehalfDisplayName**

Child element	Usage
Uri	A URI representing the author of the notes, such as sip:alice@contoso.com.
displayName	A plain-text identifier of the author of the notes, such as "Alice".
onBehalfUri	The URI of the user the call context data was authored on behalf of, if created by a third party.
onBehalfDisplayName	The plain-text identifier of the user the call context data was authored on behalf of, if created by a third party.

2.2.19.3 To Element

The **to** element describes the party the call context data was originally conveyed to by the author, who is described by the **from** element. The **to** element MUST be present in the call context data and appear only once.

```
<xs:element name="to" type="tns:XmlConvContextParticipantType" minOccurs="1" maxOccurs="1"/>

  <xs:complexType name="XmlConvContextParticipantType">
    <xs:sequence>
      <xs:element name="uri" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="displayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfUri" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfDisplayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
```

The **to** element MUST contain a **uri** element representing the user the call context data was originally conveyed to by the author of the call context data. The **to** element can also contain the following elements:

- **displayName**
- **onBehalfUri**

- **onBehalfDisplayName**

Child element	Usage
Uri	A URI representing the original recipient of the notes, such as sip:alice@contoso.com.
displayName	A plain-text identifier of the original recipient of the notes, such as "Alice".
onBehalfUri	The URI of the user the call context data was original conveyed to on behalf of, if conveyed by a third party.
onBehalfDisplayName	The plain-text identifier of the user the call context data was originally conveyed to on behalf of, if conveyed by a third party.

2.2.19.4 Participants Element

The **participant** element describes a list of one or more parties that were **participants (2)** in the call when the call context data was authored. It MUST be present in the call context data and appear only once.

```
<xs:element name="participants" type="tns:XmlConvContextParticipantCollectionType"
minOccurs="1" maxOccurs="1" />

  <xs:complexType name="XmlConvContextParticipantCollectionType">
    <xs:sequence>
      <xs:element name="participant" type="tns:XmlConvContextParticipantType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
```

The **participants** element MUST contain one or more **participant** elements.

2.2.19.5 Participant Element

The **participant** element describes a party involved with the call when the related call context data was authored. A **participant** element MUST be present for the author of the call context data and can be present for other parties in the call.

```
<xs:element name="participant" type="tns:XmlConvContextParticipantType" minOccurs="1"
maxOccurs="unbounded" />

  <xs:complexType name="XmlConvContextParticipantType">
    <xs:sequence>
      <xs:element name="uri" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="displayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfUri" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfDisplayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
```

The **participant** element MUST contain a URI representing the address of a given participant (2) to the call. The **participant** element can also contain the following elements:

- **displayName**

- **onBehalfUri**
- **onBehalfDisplayName**

Child element	Usage
Uri	A URI representing a participant (2) of the call related to the call context data, such as "sip:alice@contoso.com".
displayName	A plain-text identifier of the participant (2) identified by the URI, such as "Alice".
onBehalfUri	The URI of the user the participant (2) is acting on behalf of, if the participant (2) is acting in a third-party capacity.
onBehalfDisplayName	The plain-text identifier of the user the participant (2) is acting on behalf of, if the participant (2) is acting in a third-party capacity.

2.2.19.6 Date element

The **date** element provides a **UTC (Coordinated Universal Time)** timestamp that denotes when the author created the call context data. It MUST be present in the call context data and MUST appear only once.

```
<xs:element name="date" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
```

2.2.19.7 ConversationId element

The **conversationId** element provides a correlating identifier between the call context data and the related call that the data was authored for. It MUST be present in the call context data and MUST appear only once.

```
<xs:element name="conversationId" type="xs:token" minOccurs="1" maxOccurs="1"/>
```

The **conversationId** element MUST reflect a unique identifier related to the call that the call context data was authored for.

2.2.19.8 DataFormat element

The **dataFormat** element denotes the **MIME (Multipurpose Internet Mail Extensions)** type format of the **contextData** element in the call context data. It MUST be present in the call context data, and MUST appear only once in the call context data.

```
<xs:element name="dataFormat" type="xs:string" minOccurs="1" maxOccurs="1"/>
```

The **dataFormat** element MUST have a value of "text/plain".

2.2.19.9 ContextData element

The **contextData** element conveys the textual notes about the call that the author created to provide further context about the related call. It MUST be present in the call context data, and MUST appear only once.

```
<xs:element name="contextData" type="xs:string" minOccurs="1" maxOccurs="1"/>
```

The **contextData** element is a free-text element.

2.2.19.10 Mode element

The **mode** element provides an indication of a communications mode that was in use on the call at the time the call context data was authored.

```
<xs:element name="mode" type="xs:token" minOccurs="0" maxOccurs="unbounded"/>
```

The **mode** element can be present one or more times in the call context data, although each **mode** value SHOULD represent a unique modality involved in the call related to the call context data. The following **tokens** are supported:

- **audio**
- **video**
- **im**
- **applicationSharing**

Mode	Meaning
Audio	An audio modality was involved for the call relating to the call context data.
Video	A video modality was involved in the call relating to the call context data.
im	The instant messaging modality was involved in the call relating to the call context data.
applicationSharing	The application sharing modality was involved in the call related to the call context data.

2.2.20 Ms-Call-Info Header Field Syntax

This protocol defines a header field called **Ms-Call-Info**<14>**.** The **Ms-Call-Info** header field is used to communicate a call property to a client endpoint (5).

The ABNF for the **Ms-Call-Info** header field is specified as follows:

```
"Ms-Call-Info" HCOLON "Rgs.Anonymization"
```

HCOLON is specified in [RFC3261] section 25. If the header field contains a value other than the one specified, the header SHOULD be ignored.

A server (2) endpoint (5) that performs anonymization SHOULD send this header. The anonymization is provided to the recipient of the header. The identity of the originator of the request can still be shown.

2.2.21 P-Agent-On-Behalf-Of Header Field Syntax

This protocol defines a header field called **P-Agent-On-Behalf-Of**.<15> If a client endpoint (5) attempts to establish a call on behalf of, it MUST use the **P-Agent-On-Behalf-Of** header field.

The ABNF for the **P-Agent-On-Behalf-Of** header field is specified as follows:

```
"P-Agent-On-Behalf-Of" HCOLON name-addr / addr-spec
```

HCOLON, **name-addr** and **addr-spec** are specified in [RFC3261] section 25. This header SHOULD be present only in a SIP INVITE.

The server (2) endpoint (5) can use a back-to-back agent to establish the call. If the server (2) endpoint (5) cannot provide the service, it SHOULD decline the request.

2.2.22 E911 Call Syntax

This section describes the E911 call syntax that MUST be used by SIP endpoints (5) to initiate an E911 call. The SIP INVITE is marked with a **Priority: emergency** header, as specified in [RFC3261] section 20.26, and a **geolocation** header that identifies the content identifier of the call context that is delivered as an **application/pidf+xml** MIME part within the body of the request. The **pidf:presence** element is specified in **Presence Information Data Format (PIDF)**, as specified in [RFC3863], with a **GEOPRIV** location object, as specified in [RFC4119], extension for the status value embedded in it. The **location-info** element embedded in the **GEOPRIV** element MUST conform to the civic location format specified in [RFC5139]. If the address cannot be trusted to match the location of the endpoint (5) initiating the request, the method element embedded in the **GEOPRIV** element MUST have the value "Manual". The **GEOPRIV status** element embedded in the **pidf:presence** element is followed by an **msftE911PidfExtn extension** element, as described in section 9.

For an example E911 INVITE, see section 4.14.

3 Protocol Details

Endpoint Identification Extensions

This protocol provides several mechanisms for identification of SIP endpoints (5). These mechanisms produce an identifier that carries some or all of the following properties:

- **Long-lived:** Can persist across device, application, or server (2) shutdowns.
- **Distinguishes a specific instance:** Can distinguish a specific endpoint (5) among several endpoints (5) that share the same user or service or application **address-of-record** to maintain per-endpoint (5) state, such as **security association (SA)**, registration state, and presence state, in various SIP elements.
- **Routes to specific instance:** Can be used to address calls to a specific SIP endpoint (5) among several endpoints (5) that share the same user or service or application address-of-record event outside of the SIP transaction.

To maintain compliance with this protocol, the user agent MUST use one of the mechanisms described in sections 3.1, 3.2, and 3.3 to identify each SIP endpoint (5) that it represents.

3.1 EPID Mechanism

The **EPID** mechanism uses an **epid** parameter in the **From** or **To** header fields. When combined with the address-of-record in the **From** or **To** header field, it forms an identifier that carries all of the endpoint (5) identification properties, which are **long-lived**, **distinguishes specific instance**, and **routes to specific instance**, defined in section 3.

3.1.1 Abstract Data Model

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

User agents are responsible for generating **epid** parameter values in accordance with requirements in section 3.1.3.1; however, the exact mechanism is outside the scope of this protocol. To create a value for an **epid** parameter, the user agent SHOULD use a hexadecimal string no more than 16 hexadecimal characters long. A 64-bit random number or the 8-byte **Media Access Control (MAC) address** of the local network interface card can be encoded as a 16-character hexadecimal string to form a value for an **epid** parameter. This string SHOULD be stored in persistent storage for future use by the same user agent.

3.1.2 Timers

None.

3.1.3 Initialization

Except as specified in the following sections, the rules for initialization are as specified in [RFC3261].

3.1.3.1 User Agent Initialization

To use the EPID endpoint (5) identification mechanism defined in this section, a user agent MUST obtain an identifier that complies with the **epid-param-value** syntax defined in section 2.2.6 and uniquely identifies itself within all user agents that share the same address-of-record. This identifier SHOULD be persisted across power cycles of the SIP endpoint (5) that the user agent represents.

3.1.4 Higher-Layer Triggered Events

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.1.4.1 User Agent Operation

To use the EPID endpoint identification mechanism defined in this section, a user agent MUST add the **epid** parameter with a value obtained as described in section 3.1.3 to the **From** header field of every request that it generates, whether or not the request is part of a SIP transaction.

The SIP dialog state created by the user agent that is compliant with this protocol MUST include the **remote epid** parameter in addition to other elements defined in [RFC3261] section 12. For a **UAC**, **remote epid** is set to the value of the **epid** parameter in the **To** header field, if it is present, and is set to empty if it is not present. For a **UAS**, the **remote epid** parameter is set to the **epid** parameter value in the **From** header field, if it is present, and is set to empty if it is not present.

When forming a request within an existing SIP transaction that contains a non-empty **remote epid** in its state, the user agent that is compliant with this protocol MUST add the **epid** parameter with the value of **remote epid** to the **To** header field.

If the user agent that is compliant with this protocol initiates a call to a specific SIP endpoint (5), it SHOULD obtain the address-of-record and the value of the **epid** parameter for such an endpoint (5). The user agent can obtain the address-of-record and the **epid** parameter from the previous dialog with the same endpoint (5) or from the presence document described in [MS-PRES], or it can use any other mechanism. The user agent SHOULD then create a request with the desired address-of-record placed in the **Request-URI** field, place the same address-of-record in the URI of the **To** header field, and add an **epid** parameter to the **To** header field.

3.1.5 Message Processing Events and Sequencing Rules

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.1.5.1 User Agent Operation

If the **To** header field of the request received by the user agent compliant with this protocol contains an **epid** parameter and its value differs from the user agent's own **epid** parameter value obtained as described in section 3.1.3, the user agent MUST discard the request instead of processing it and generating a response.

3.1.5.2 SIP Registrar Operation

If the REGISTER request processed by the SIP registrar compliant with this protocol contains an **epid** parameter in the **From** header field, the registrar MUST obtain the value of the **epid** parameter and add it to the SIP location service record maintained by this registrar, in addition to the other required information described in [RFC3261] section 10.

3.1.5.3 SIP Proxy Operation

If a SIP proxy compliant with this protocol stores any state associated with SIP endpoints (5), it SHOULD use the value of the **epid** parameter, if one is present in the **From** or **To** header fields, combined with the address-of-record from the URI of the corresponding header, as an index into its state table. Specifically, the address-of-record and **epid** parameter from the **From** header field SHOULD be used to identify user agent client (UAC) endpoints (5), and **address-of-record** and **epid** parameters from the **To** header field SHOULD be used to identify UAS endpoints (5).

If a SIP proxy compliant with this protocol receives a request targeted at the address-of-record that belongs to the domain that this proxy is responsible for, and it is supposed to access a SIP location service to compute the request targets, as specified in [RFC3261] section 16, it MUST perform two additional steps:

1. The SIP proxy MUST examine the **To** header field of the request. If the **To** header field contains an **epid** parameter, the proxy MUST ignore any records returned by the SIP location service that do not have the same **epid** parameter value when computing request targets.

2. If the SIP proxy uses any record returned by the SIP location service as a request target, and the record contains an **epid** parameter value placed there by the SIP registrar, as described in section 3.1.5.2, it MUST add the **epid** parameter value to the **To** header field as an **epid** parameter, unless the **To** header field of the request already has an **epid** parameter. In the latter case, the value in the parameter is expected to be the same as in the SIP location service record; otherwise, the SIP proxy would have ignored the record, as discussed in step 1.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 SIP.INSTANCE Mechanism

This method is based on [IETFDRAFT-MCICSIP-11]. It employs the **+sip.instance** media feature tag as a **Contact** header field parameter. The value of the **+sip.instance** parameter in combination with the address-of-record in the **From** or **To** header fields forms an identifier that carries the following two properties defined in section 3:

- **Long-lived.**
- **Distinguishes specific instance.**

It does not carry the **routing to specific instance** property because the **Contact** header field and its parameters are associated with the source, but not the destination, of the message.

This protocol specifies that the user agent MUST use only the UUID URN identifier, as defined in [RFC4122] as its instance identifier in the **+sip.instance** media feature tag.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

User agents are responsible for generating **+sip.instance** parameter values in accordance with the requirements in section 3.2.3.1; however, the exact mechanism is outside the scope of this protocol. To create a value for the **+sip.instance** parameter, a user agent can use methods described in [IETFDRAFT-MCICSIP-11] section 4. Specifically, the user agent can use the methods of UUID URN computation based on time, unique names such as MAC address, or a random number generator, which are defined in [RFC4122].

Except as specified in the following sections, the rules for initialization are as specified in [RFC3261].

3.2.3.1 User Agent Initialization

To use the **SIP.INSTANCE** endpoint (5) identification mechanism defined in this section, a user agent MUST obtain a UUID using any of the procedures described in [RFC4122]. However, if the same user agent also uses the EPID mechanism, as described in section 3.1, it MUST compute an EPID namespace UUID using the algorithm for name-based UUID described in [RFC4122] section 4.3, with specific constants and algorithm choices applicable to the EPID namespace defined in this protocol.

To compute an EPID namespace:

1. Allocate a UUID to use as a namespace ID for all UUIDs generated from names in that namespace. For UUIDs in the EPID namespace defined in this protocol, the following UUID has been allocated:

   fcacfb03-8a73-46ef-91b1-e5ebeeaba4fe

2. Choose the **SHA-1 hash** algorithm described in [FIPS180].

3. Convert the EPID value to a canonical sequence of octets, which for the EPID namespace has been defined as ASCII encoding of the **epid** parameter value as it appears in the **From** or **To** header field of the SIP message.

4. Compute the **hash** of the namespace ID concatenated with the name.

5. Set octets zero through 3 of the **time_low** field to octets zero through 3 of the hash.

6. Set octets zero and 1 of the **time_mid** field to octets 4 and 5 of the hash.

7. Set octets zero and 1 of the **time_hi_and_version** field to octets 6 and 7 of the hash.

8. Set the four most significant bits, which are bits 12 through 15, of the **time_hi_and_version** field to the 4-bit version number, as specified in [RFC4122] section 4.1.3. For name-based UUIDs computed with the **SHA-1** function, this sequence is 0101.

9. Set the **clock_seq_hi_and_reserved** field to octet 8 of the hash.

10. Set the two most significant bits, which are bits 6 and 7, of the **clock_seq_hi_and_reserved** to zero and 1, respectively.

11. Set the **clock_seq_low** field to octet 9 of the hash.

12.Set octets zero through 5 of the node field to octets 10 through 15 of the hash.

13.Convert the resulting UUID to local byte order.

In the previous procedure, the UUID obtained SHOULD be persisted across power cycles of the SIP endpoint (5) that the user agent represents.

3.2.4 Higher-Layer Triggered Events

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.2.4.1 User Agent Operation

To use the **SIP.INSTANCE** endpoint (5) identification mechanism defined in this section, the user agent MUST add the **+sip.instance** parameter with an obtained UUID URN value, as described in section 3.2.3, to the **Contact** header field of the messages which would otherwise carry the **Contact** header field because of SIP protocol requirements. [RFC3261] requires the addition of the **Contact** header field to the dialog creating requests and responses and a REGISTER request. The **+sip.instance** parameter syntax is defined in section 2.2.4.

3.2.5 Message Processing Events and Sequencing Rules

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.2.5.1 SIP Registrar Operation

If a REGISTER request processed by a SIP registrar compliant with this protocol contains a **+sip.instance** parameter in the **Contact** header field, the registrar MUST obtain the **+sip.instance** parameter value and validate that it conforms to the UUID URN syntax described in [RFC2141] and [RFC4122]. Furthermore, if the REGISTER request also contains an **epid** parameter in the **From** header field, the registrar MUST validate that the name-based UUID, derived as described in section 3.2.3 from the **epid** parameter value, is equal to the UUID extracted from the **+sip.instance** parameter value.

If either of these validations fails, the registrar MUST reject the REGISTER request with a 400 response code. Otherwise, the registrar MUST add the UUID value that is extracted from the **+sip.instance** parameter value to the SIP location service record maintained by this registrar in addition to the other required information described in [RFC3261] section 10.

3.2.5.2 SIP Proxy Operation

If a SIP proxy compliant with this protocol stores any state associated with SIP endpoints (5), it SHOULD use the value of the UUID from the **+sip.instance** parameter in the **Contact** header field, if one is present, combined with the address-of-record from the URI of the **From** or **To** header field as an index into its state table. Specifically, the UUID from the **+sip.instance** parameter and the address-of-record from the **From** header field SHOULD be used to identify the UAC endpoint (5) in requests, and the UUID from the **+sip.instance** and address-of-record from the **To** header field SHOULD be used to identify the UAS endpoint (5) in each response.

Before the UUID from the **+sip.instance** parameter is used, the SIP proxy MUST obtain the value of the **+sip.instance** parameter and validate that it conforms to the UUID URN syntax specified in the [RFC2141] and [RFC4122]. Furthermore, if the message is a request and it also contains an **epid** parameter in the **From** header field or the message is a response and it also contains an **epid**

parameter in the **To** header field, the SIP proxy MUST validate that the name-based UUID derived as described in section 3.2.3 from the **epid** parameter value is equal to the UUID extracted from the **+sip.instance** parameter value. If validation fails, the proxy SHOULD respond with 400 response code.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 GRUU Mechanism

This method is based on [IETFDRAFT-OUGRUAUSIP-10] and uses the GRUU to provide an identifier that carries all of the properties, which are **long-lived**, **distinguishes specific instance**, and **routes to specific instance**, defined in section 3. As described in [IETFDRAFT-OUGRUAUSIP-10] section 6, only the SIP registrar authoritative for the domain can generate the GRUU for all addresses-of-record that belong to the domain and user agents MUST use either a SIP registration procedure or some other protocol or administrative mechanism to obtain a GRUU.

3.3.1 Abstract Data Model

None.

3.3.2 Timers

None.

3.3.3 Initialization

None.

3.3.3.1 User Agent Initialization

To use a GRUU-based endpoint (5) identification mechanism defined in this section, a user agent MUST obtain a GRUU from a SIP registrar using either the registration procedure defined in [MS-SIPREGE] or, if the user agent is a part of a server (2) application or a conferencing endpoint (5), it can obtain a GRUU using an administrative method outside the scope of this protocol.

3.3.4 Higher-Layer Triggered Events

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.3.4.1 User Agent Operation

To use the GRUU-based endpoint (5) identification mechanism defined in this section, a user agent MUST use the GRUU that it previously obtained, as described in section 3.3.3.1, to populate the URI in the **Contact** header field of the messages which would otherwise carry the **Contact** header field because of SIP protocol requirements. [RFC3261] requires the addition of the **Contact** header field to the dialog creating the requests. Although [RFC3261] also requires the presence of a **Contact** header field in the REGISTER request, the GRUU MUST NOT be used to populate it.

When using GRUU as a URI in the **Contact** header field, the user agent can also add a **grid** URI parameter to the **Contact** header field with a value that satisfies the syntax defined in section 2.2.1. As noted in [IETFDRAFT-OUGRUAUSIP-10] section 8.1.1, the user agent can manufacture an infinite supply of GRUUs, each of which differs by the value of the **grid** parameter. When a user agent receives a request that was sent to the GRUU, it is able to tell which GRUU was invoked by looking at the **grid** parameter.

When sending a request that contains a GRUU in the **Contact** header field, the user agent compliant with this protocol MUST forward it to a SIP registrar or proxy in the same domain as the one from which the user agent obtained the GRUU.

If the same user agent also uses the EPID mechanism, as described in section 3.1, and it uses the registration procedure defined in [MS-SIPREGE] to obtain the GRUU, it MUST insert the same **epid** parameter value into the **From** header field of every request as the one it used when performing the registration.

3.3.5 Message Processing Events and Sequencing Rules

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.3.5.1 SIP Registrar Operation

A SIP registrar compliant with this protocol can generate a GRUU by creating a SIP URI with an address-of-record in the domain that the registrar is responsible for as the user and domain portion. It then MUST add a mandatory **GRUU** parameter, and it SHOULD add an additional **opaque** parameter with a value that encodes the user agent type and an identifier of a specific endpoint (5) bound with the user agent address-of-record, as specified in [RFC3261] section 10.2.1, or an instance of the application or server (2).

When generating a GRUU for a user agent that follows the registration procedure defined in [MS-SIPREGE], the registrar can create a URI using ABNF for **user-agent-gruu** syntax, as defined in section 2.2.2. The address-of-record value in the ABNF comes from the URI in the **To** header field. The ABNF for **ua-opaque-val** syntax is defined in section 2.2.1, where **encoded-uuid-val** value is obtained by applying an encoding procedure to the binary form of the UUID obtained from the **+sip.instance** parameter of the **Contact** header field. The encoding procedure MUST produce a string that satisfies the syntax of a SIP **URI** parameter, as defined in [RFC3261] section 25. One example of an encoding procedure is defined in [RFC3548] section 4.

When generating a GRUU for an application that implements voice mail service for a user, the registrar can create a URI using ABNF for **voice-mail-gruu** syntax, as defined in section 2.2.2. The address-of-record value in the ABNF MUST belong to the user whose voice mail service is represented by the GRUU. The ABNF **appvoicemailopaqueval** syntax is defined in section 2.2.1.

When generating a GRUU for an application that implements location profile service for a user, the registrar can create a URI using ABNF for **location-profile-gruu** syntax, as defined in section 2.2.2. The address-of-record value in the ABNF MUST belong to the user whose location profile service is represented by the GRUU. The ABNF **applocationprofileopaqueval** syntax is defined in section 2.2.1.

When generating a GRUU for a multimedia **conference** endpoint (5) created by the user agent that follows the procedure for conference creation defined in [MS-CONFBAS], the registrar can create a URI using ABNF for **conf-endpoint-gruu** syntax, as defined in section 2.2.2. The address-of-record value in the ABNF MUST be associated, as specified in [RFC3261] section 10.2.1, with the user that organized the conference. The ABNF for **app-conf-opaque-val** syntax is defined in section 2.2.1, where **conf-entity-val** value describes the type of conferencing endpoint (5). Specific values in the

ABNF are described in [MS-CONFBAS]. The **encoded-conf-id-val** value can be obtained by applying the procedure defined in [RFC3548] section 4 to the binary form of conference identifier, which is defined in [MS-CONFBAS].

When generating a GRUU for a server (2) deployed within a domain for which a SIP registrar is responsible, the registrar can create a URI using ABNF for **server-instance-gruu** syntax defined in section 2.2.2. The **server-fqdn** value in the ABNF is a **fully qualified domain name (FQDN) (1)** of the server (2). The **domain-fqdn** value is the FQDN (1) of the domain for which the SIP registrar is responsible. The ABNF for **server-opaque-val** syntax is defined in section 2.2.1, where **server-type-val** value describes the type of service provided by the server (2) with the **HomeServer** string representing the SIP registrar and presence server (2), the **MRAS** string representing the media relay **authentication (2)** server (2), the **MediationServer** string representing the mediation server (2), and a **QoSM** string representing the quality of service monitoring server (2). The **encoded-server-instance-val** value can be obtained by applying the procedure defined in [RFC3548] section 4 to the binary form of the **globally unique identifier (GUID)** that is associated with the server (2) instance entry in **Active Directory**.

When a SIP registrar compliant with this protocol creates a SIP location service record for user agents that use the registration procedure defined in [MS-SIPREGE], it MUST generate a GRUU that satisfies all of the following requirements:

- When a request is sent to the GRUU, it routes to a SIP proxy with access to the SIP location service record that this registrar creates.
- The GRUU MUST include the **gruu** URI parameter.
- If the GRUU contains an **opaque** URI parameter, the URI that results from stripping out the **opaque** and **gruu** URI parameters MUST be equivalent to the address-of-record for which the SIP location service record is created.

The registrar then MUST store the GRUU with the SIP location service record that it creates as the result of the registration procedure in addition to other information described in [RFC3261] section 10. It MUST also return the GRUU to the user agent requesting it as a part of the registration procedure defined in [MS-SIPREGE]. The registrar can also use other methods of delivering GRUUs to user agents that represent server (2) application or conferencing endpoints (5) in the registrar domain.

3.3.5.2 SIP Proxy Operation

If a SIP proxy compliant with this protocol stores any state associated with SIP endpoints (5), it SHOULD use the value of the GRUU, if one is present in the **Contact** header field, as an index into its state table. Specifically, the GRUU from the **Contact** header field of SIP request messages SHOULD be used to identify UAC endpoints (5), and the GRUU from the **Contact** header field of SIP response messages SHOULD be used to identify UAS endpoints (5).

If a SIP proxy compliant with this protocol receives a request outside of the dialog, with no **Route** header fields, targeted at the URI that belongs to the domain that this proxy is responsible for, and it is supposed to access a SIP location service so that it can compute the request targets, as specified in [RFC3261] section 16, it MUST examine the target URI of the request.

For example, the **Request-URI** field is examined. If the URI contains a **gruu** parameter, and thus is a GRUU, and the URI does not refer to any GRUU known in the domain, the proxy rejects the request with a 404 response.

The proxy MUST ignore any records returned by the SIP location service that do not have the same GRUU value when computing request targets.

If the SIP proxy uses any record returned by the SIP location service as a request target, it MUST copy the **grid** parameter and its value from the original target URI, or GRUU, into the new target URI obtained from the SIP location service record. If the original target URI did not contain a **grid** parameter or the parameter value was empty, the proxy MUST insert a **grid** parameter value into the new target URI.

If a SIP proxy compliant with this protocol receives a mid-dialog request with **Route** header fields and a **Request-URI** field that belongs to the domain that this proxy is responsible for, and the proxy has access to the SIP location service in the domain, it MUST examine the URI and the **Request-URI** field. If the URI contains a **gruu** parameter, which means that it is a GRUU, and the URI does not refer to any GRUU known in the domain, the proxy MUST reject the request with a 404 response.

The proxy MUST contact the SIP location service for the domain for records where the address-of-record in the record matches the address-of-record in the URI and, from the returned set of records, select the records that have the same **GRUU** value that appears in the **Request-URI**.

If at least one record is selected:

- The SIP proxy MUST arbitrarily choose one of the selected records as a new request target. It MUST then copy the **grid** parameter and its value from the original target URI (**GRUU**) into the new target. If the original target URI did not contain the **grid** parameter or the parameter value was empty, the proxy MUST insert a **grid** parameter value into the new target URI.
- If there are no **Route** headers in the request after the proxy removes the topmost **Route** header pointing to it, as specified in [RFC3261] section 16.4, the proxy MUST copy all routing information from the selected SIP location service record to the **Route** header of the request.

If no records were selected, the proxy SHOULD reject the request with a 480 Temporarily Unavailable response.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

3.4 Firewall and Network Address Translation Traversal Aid Extensions

When a user agent forms a connection to a SIP proxy, SIP registrar, or other SIP servers (2) and that connection traverses a firewall or a NAT device, the server (2) might be unable to make a connection back to the user agent because of the firewall or NAT device. Because, during normal SIP operation, servers (2) have to send responses back to the user agent, as well as initiate and forward requests destined to the user agent, the transport layer on the SIP server (2) has to route messages to the user agent over the existing connection established from the user agent. To aid the transport layer on the SIP server (2) in routing messages over the connection from the protocol client, this protocol defines mechanisms that help save connection identification information in **Via, Contact, Record-Route,** and **Path** header fields of the incoming SIP requests. The header fields described in this protocol are designed to preserve routing information for use by the transport layer. Specifically, the following list of header fields serve this purpose:

- **Via** header fields MUST be copied from the SIP requests to responses, as specified in [RFC3261] section 8.2.6.2.

- **Contact** and **Record-Route** header fields MUST be preserved in dialog state, as specified in [RFC3261] section 12.1.1, and copied to mid-dialog requests, as specified in [RFC3261] section 12.2.1.1.
- **Contact** and **Path** header fields are saved in the SIP location service database for the user agent's domain, as specified in [RFC3327] section 5.3, and then inserted into the requests forwarded by the SIP proxies authorized for the domain, as specified in [RFC3327] section 5.4.

3.4.1 Abstract Data Model

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

[RFC3261] Section 18 specifies that the transport layer of every SIP element is responsible for managing persistent connections over the **Transmission Control Protocol (TCP)** and other connection-oriented transport protocols and then index them based on the **tuple** formed from transport address, port, and protocol of the far end of the connection. Far end is defined in [RFC3261] section 18 as the destination for connections opened by the transport layer and as a source for connections accepted by the transport layer.

If a TCP connection accepted by the transport layer traverses a NAT device, the address and port in the tuple of the far end of the connection belong to the NAT device, and not to the user agent. If the original user agent disconnects for any reason, and another user agent is allocated the same address and port, the transport layer of the SIP element cannot distinguish the new user agent from the old user agent. To avoid misidentifying the connection, the transport layer of the SIP element can maintain a counter that gets incremented with each created connection, and can make this counter a part of the tuple that indexes connections. The counter is of sufficient length that it does not wrap around before the end of the lifetime of all transactions, dialogs, and SIP location service records that were created based on the messages that had the value identifying the connection populated into their header fields.

3.4.2 Timers

None.

3.4.3 Initialization

None.

3.4.4 Higher-Layer Triggered Events

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.4.4.1 User Agent Operation

To use the firewall and NAT device traversal mechanism defined in this section, the user agent MUST add a **proxy** parameter with the value "replace" to the **Contact** header field of the messages that would otherwise carry the **Contact** header field because of SIP protocol requirements and when the URI in the **Contact** header field contains the user agent's IP address in its host portion or as the value of the **maddr** parameter. The exact syntax for the **proxy** parameter is defined in section

2.2.4, and the syntax for the SIP URI, including the host portion and the **maddr** parameter, is defined in [RFC3261] section 25.1.

3.4.5 Message Processing Events and Sequencing Rules

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261].

3.4.5.1 SIP Server (Proxy, Registrar) Operation

When a SIP proxy, SIP registrar, or any SIP server (2) compliant with this protocol receives a message that has a **Contact** header field with the **proxy** parameter, it MUST perform the following steps in addition to the processing described in the [RFC3261]:

1. If the server (2) is not the first node after the user agent, it MUST reject the message with a 400 response if the message is a request, and then discard the message if it is a response. The SIP server (2) can determine if it is the first hop by examining the **Via** header field. More than one value in this field indicates that the SIP server (2) is not the first hop.

2. If the **proxy** parameter in the **Contact** header field has any value other than "replace", the server (2) MUST reject the message with a 400 response if message is a request, and discard the message if it is a response.

3. If the URI in the **Contact** header field has a **transport** parameter and the value of this parameter is not the same as the transport protocol of the connection over which the message was received, the server (2) MUST reject the message with a 400 response if the message is a request, and discard the message if it is a response.

4. The server (2) MUST remove the **proxy** parameter and its value from the **Contact** header field.

5. If the URI in the **Contact** header field has a **maddr** parameter, the server (2) MUST replace its value with the value of the IP address of the far end of the connection on which the message was received.

6. If the URI in the **Contact** header field does not have a **maddr** parameter and the host portion of the URI is not an IP address, such as a host name, the server (2) MUST add a **maddr** parameter with the value of the IP address of the far end of the connection on which the message was received to the **Contact** header field.

7. If the URI in the **Contact** header field does not have a **maddr** parameter and the host portion of the URI is an IP address and its value is not the same as the value of the IP address of the far end of the connection on which the message was received, the server (2) MUST replace the host portion of the URI with the value of the IP address of the far end of the connection on which the message was received.

8. If the URI in the **Contact** header field does not have a port portion or if the port portion value is not the same as the value of the port of the far end of the connection on which the message was received, the server (2) MUST add the port or replace its value with the value of the port of the far end of the connection on which the message was received.

9. The server (2) MUST add a parameter with a value that uniquely identifies the connection on which the message was received among all other connections that were or could in the future be established by the server (2) with the same tuple (address, port, and transport) on the far end to the URI of the **Contact** header field. The server (2) can use the **ms-received-cid** parameter for this purpose and populate it with the value of the counter described in section 3.4.1.

10. If the server (2) is a SIP proxy, it MUST insert the **Record-Route** header field into the message, as described in [RFC3261] section 16, to remain on the path of all the subsequent messages in the dialog that is created by the message.

The syntax for a SIP URI, including host and port portions and a **maddr** parameter, is defined in [RFC3261] section 25.1.

When a SIP server (2) compliant with this protocol processes a request from another SIP element, it SHOULD save the identification information of the connection on which it received the request in the topmost **Via** header field. To do this, the server (2) SHOULD use the following **Via** header field parameter values:

- **received** parameter value, as defined in [RFC3261], to save the IP address of the far end of the connection.
- **ms-received-port** parameter value, as defined in section 2.2.5, to save the port number of the far end of the connection.
- **ms-received-cid** parameter value, defined in section 2.2.5, to save unique connection identifiers, which are values that uniquely identify the connection on which the message was received among all other connections that were or could in the future be established by the server (2) with the same tuple (address, port, and transport. The server (2) can populate **ms-received-cid** with the value of the counter described in section 3.5.1.

3.4.6 Timer Events

None.

3.4.7 Other Local Events

None.

3.5 Extensions for Reliable and Consistent Message Routing Within Redundant Server Network

Messages between user agents in a SIP element network traverse a set of one or more servers (2) or proxies that run and provide services such as network edge traversal, authentication (2), call data records, and message content archiving. It is often essential for the SIP protocol itself, as well as for the services provided by the SIP proxies, that the related messages, such as responses to the requests or all messages in the dialog, traverse the same set of proxies in a specific order. Furthermore, core functionality of the SIP proxy, such as routing, as well as potential services that it runs and provides depend on the capability to propagate contextual information between related messages. For example, the transport layer of the SIP proxy that adds the **received** parameter to the **Via** header field in the request depends on the availability of this parameter in the response to route the response.

[RFC3261] defines two basic mechanisms that ensure that the response follows the path of the request in reverse order, which are a mechanism to insert and process the **Via** header field, and that all requests in the dialog traverse the proxies that specifically chose to be on the dialog's path, which are a mechanism to insert **Record-Route** header fields, store them in the dialog route set, and populate request **Route** header fields from the dialog route set. This protocol compliments these basic mechanisms with the following additional specific functions:

- Storing references to the information that spans the lifetime of multiple SIP transactions and dialogs, such as references to data associated with the identity represented by the user agent.

- Storing information about specific services provided by the SIP proxies within the context of the dialog.
- Storing the FQDN (1) of a specific server (2) in a set of multiple redundant SIP proxies sharing the same common FQDN (1) that handles messages in the dialog.
- Ensuring that the essential context information in the **Via** or **Record-Route** header fields that the proxy inserted into the message or information in the **Via**, **Record-Route**, and **Contact** header fields inserted by other SIP elements was preserved and populated correctly without modifications into related messages by the user agents.

3.5.1 Abstract Data Model

None.

3.5.2 Timers

3.5.2.1 SIP Proxy Operation

If the SIP proxy uses an **HMAC** algorithm, as described in [FIPS198a], to protect the integrity of the **Record-Route**, **Contact**, or **Via** headers and it periodically changes the key used in the HMAC computation, as recommended by [FIPS198a], or if it uses a similar algorithm that depends on periodically updated keys, the proxy MUST start a timer per key when the key is last used to compute the HMAC before it gets changed and it MUST retain the key until the timer fires. The timer SHOULD fire no earlier than 1 hour after it is started for keys used to protect information in **Via** and **Record-Route** header fields that are copied from the request to the response. The timer SHOULD fire no earlier than 8 hours for keys used to protect information in **Contact** and **Record-Route** header field URIs that is preserved in the dialog route set and used to populate **Route** header fields in mid-dialog requests.

3.5.3 Initialization

The SIP proxy SHOULD create one or more tables to maintain the information that spans the lifetime of the dialog and then store an index to this type of table in the **Record-Route** header field that it inserts into the dialog-creating messages. Specifically, the SIP proxy SHOULD create a table of endpoints (5) that user agents communicating with the proxy represent.

Consequently, the SIP proxy SHOULD add an index to an entry in the **endpoint** table as a value of the **ms-opaque** parameter in the **Record-Route** header field **URI** which this proxy inserts into the messages, as described in [RFC3261] section 16. When the **Record-Route** header field **URI** is then stored in the dialog route set, and later copied to the **Route** header field of the mid-dialog request, the value of the **ms-opaque** parameter represents the identity of the UAS endpoint (5).<16>

Furthermore, the SIP proxy SHOULD add an index to an entry in the **endpoint** table as a value of the **ms-identity** parameter of the **Record-Route** header field **URI** which this SIP proxy inserts into the messages, as described in [RFC3261] section 16. When the **Record-Route** header field **URI** is then stored in the dialog route set and later copied to the **Route** header field of the mid-dialog request, the value of the **ms-identity** parameter can represent the identity of the UAC endpoint (5).<17>

The SIP proxy can add **ms-role-rs-to** or **ms-role-rs-from** parameters to the **Record-Route** header field **URI** so that when the **Record-Route** header field **URI** is stored in the dialog route set, and later copied to the **Route** header field of the mid-dialog request, the **ms-role-rs-to** parameter indicates that this SIP proxy is an authorized proxy for the UAS endpoint (5) domain while the **ms-**

role-rs-from parameter indicates that the SIP proxy is an authorized proxy for the domain of the UAC endpoint (5).<18>

If the SIP server (2) is a member of a set of multiple redundant proxies that appear to share the same FQDN (1) with some or all other SIP elements that communicate with them, the SIP server (2) can add its specific unique FQDN (1) as the value of the **ms-fe** parameter of the **Record-Route** or **Contact** header field **URI** so that when the **Record-Route** or **Contact** header field **URI** is stored in the dialog route set, and later copied to the Request-URI field or **Route** header field of the mid-dialog request, the **ms-fe** parameter contains the unique FQDN (1) of the server (2).

The SIP proxy can add an **ms-ent-dest** parameter to the **Record-Route** header field **URI** so that when the **Record-Route** header field **URI** is stored in the dialog route set, and later copied to the **Route** header field of the mid-dialog request, the **ms-ent-dest** parameter indicates that if the SIP proxy is an authorized proxy for the domain of the UAC endpoint (5), the UAS endpoint (5) belongs to the same domain.<19>

The SIP proxy can combine all state information that it maintains for the endpoints (5) in the dialog that spans the lifetime of the dialog, encode it using a method that produces output that satisfies the SIP **URI** parameter syntax, such as the method defined in [RFC3548] section 4, and add it as a value of an **opaque** parameter to the **Record-Route** header field **URI** that this SIP proxy inserts into messages, as described in [RFC3261] section 16.<20> When the **Record-Route** header field **URI** is then stored in the dialog route set, and later copied to the **Route** header field of the mid-dialog request, the **opaque** parameter value can be decoded and all of the information that the proxy previously stored can be made available to it.<21>

3.5.4 Higher-Layer Triggered Events

None.

3.5.5 Message Processing Events and Sequencing Rules

3.5.5.1 SIP Proxy Operation

If the SIP proxy uses an HMAC algorithm, as specified in [FIPS198a], to protect the integrity of the **Record-Route** or **Contact** header fields, and it periodically changes the key used in the HMAC computation, as recommended by the [FIPS198a], or if it uses a similar algorithm that depends on periodically updated keys, and it receives a SIP request that contains the HMAC that the SIP proxy previously inserted, and the SIP proxy no longer has the key to compute the HMAC, the SIP proxy SHOULD reject the request with a 481 Call Leg Does Not Exist response.<22> However, if the SIP proxy implements the extensions for dialog state recovery, as described in section 3.6, it SHOULD follow the procedure defined there to send a 430 Flow Failed or a 481 Call Leg Does Not Exist response.<23>

3.5.6 Timer Events

When the timer described in section 3.5.2.1 fires, the SIP proxy can destroy the key for which the timer was started. The SIP proxy SHOULD then reject all requests that contain an HMAC generated with the destroyed key with a 481 Call Leg Does Not Exist response, as described in section 3.5.5.1.<24> However, if the SIP proxy implements the extensions for dialog state recovery, as described in section 3.6, it MUST follow the procedure defined there to send a 430 Flow Failed or a 481 Call Leg Does Not Exist response.<25>

3.5.7 Other Local Events

None.

3.6 Extensions for Dialog State Recovery in Case of Outages in SIP and other Network Elements on the Dialog Path

This section follows the product behavior described in endnote <26>.

To achieve reliability of message delivery between SIP endpoints (5), typical installations deploy sets of redundant SIP proxies and other network elements, such as firewalls or NAT devices, providing an alternate path to process and route traffic between endpoints (5) in cases of unplanned or scheduled outages. However, as described in section 3.4 and section 3.5, both SIP and other network elements often maintain state information that they associate directly or indirectly, through indexing, with the SIP dialog state, and when the main SIP proxy or other network device goes out of service, the alternate, or redundant, element, which does not have the corresponding state, cannot continue processing or routing messages. This protocol defines extensions that allow SIP proxies to communicate to the endpoints (5) that the SIP dialog state carried in the mid-dialog messages no longer has necessary information. It also provides a mechanism for endpoints (5) to update, or recover, the dialog state without breaking the SIP dialog and associated media, such as audio or video, session.

A SIP endpoint (5) can register with its SIP registrar via one or more SIP proxies, as specified in [RFC3261] and [MS-SIPREGE]. If the SIP registrar gets recycled because of unplanned or scheduled outages, the binding information associated with the SIP endpoint (5) can be lost. In such a scenario, SIP message delivery to the endpoint (5) is impacted until the client re-registers and recreates the registration binding. If the SIP endpoint (5) tries to establish a new dialog with another SIP endpoint (5), mid-dialog messages are not deliverable until the SIP endpoint (5) refreshes its registration binding. This protocol defines extensions that allow SIP registrars to communicate to the endpoints (5) that the SIP registration binding is no longer valid. It also provides a mechanism for endpoints (5) to update the registration binding without breaking any other SIP dialogs and associated media sessions that it is participating in.<27>

3.6.1 Abstract Data Model

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

3.6.1.1 SIP Proxy Operation

Section 3.6.5.1 describes a way for a SIP proxy to associate the state information needed to process and route mid-dialog messages with the dialog route set. This state information can include references to transport connection identifiers, SAs, and endpoint (5) registration information, and can be used by the SIP proxy to detect that referenced information is either missing or invalid because it was created and maintained by another redundant SIP proxy.

3.6.1.2 User Agent Operation

A user agent supporting the dialog state recovery can keep states for recovery mode and can remember state for transaction retries specified in section 3.6.5.3 for dialogs where recovery is enabled.

3.6.2 Timers

3.6.2.1 User Agent Operation

If a user agent enables recovery procedures described in this section for a specific SIP dialog for which it also negotiated a session timer as described in [RFC4028], it SHOULD start a recovery refresh timer upon creation, with the interval set to at least the interval it negotiated for the session timer.

3.6.3 Initialization

3.6.3.1 User Agent Operation

A user agent compliant with this specification SHOULD enable recovery procedures for dialogs where loss of communications on SIP signaling path leads to loss of valuable state and content information, such as media state and content in an audio call, that cannot be easily recovered. User agents SHOULD NOT enable the recovery procedures for dialogs where state and content can be seamlessly restored by creation of the replacement dialog, such as the presence subscription dialog described in [MS-PRES].

3.6.4 Higher-Layer Triggered Events

3.6.4.1 User Agent Operation

If a user agent enables recovery procedures described in this section for a specific SIP dialog, it MUST include the **Ms-Dialog-Route-Set-Update** option tag in the **Supported** header field of all the requests in the dialog.

The user agent SHOULD negotiate a mechanism to periodically refresh the dialog with recovery procedures enabled. For INVITE based dialogs, the user agent SHOULD use the session timer mechanism described in [RFC4028]. For **SUBSCRIBE** based dialogs, the user agent SHOULD use the subscription refreshes described in [RFC3265]. Regardless of the specific refresh mechanism chosen by the user agent, all dialog refresh requests MUST be target refresh requests specified in [RFC3261].

3.6.5 Message Processing Events and Sequencing Rules

3.6.5.1 SIP Proxy Operation

When a SIP proxy receives a mid-dialog request and it extracts references to the state information, such as transport connection identifier, security association, or endpoint (5) registration information, that it previously encoded into the dialog route set, as described in section 3.6.1.1, the SIP proxy SHOULD check if the corresponding state information is available and valid for request processing and routing. If the information is no longer available or cannot be used to process and route the mid-dialog request, the proxy MUST perform the following steps:

1. Check if the **Ms-Dialog-Route-Set-Update** option tag is present in the **Supported** header field of the request. If the **Ms-Dialog-Route-Set-Update** option tag is NOT present, the SIP proxy SHOULD reject the request with a 481 Call Leg Does Not Exist response and stop further processing.

2. If the **Ms-Dialog-Route-Set-Update** option tag is present, the SIP proxy MUST reject the request with a 430 Flow Failed response and add a **P-Dialog-Recovery-Action** header field. The value of the **P-Dialog-Recovery-Action** header field indicates the actions that either the source

or destination endpoint (5) of the currently processed mid-dialog request needs to take to make processing or routing possible for subsequent requests in the dialog. The value of the **P-Dialog-Recovery-Action** header field MUST be set as follows:

- **Dialog-Route-Set-Update:** The proxy can recover if the source endpoint (5) of the mid-dialog request performs a dialog recovery procedure, as described in section 3.6.5.3.4.
- **Registration-Route-Set-Update**, **Dialog-Route-Set-Update:** The proxy determines that it can recover if the source endpoint (5) of the current request first refreshes its registration, as described in [RFC3261] section 10.2.4, and then performs a dialog recovery procedure, as described in section 3.6.5.3.4.
- **Wait-For-Session-Update:** The proxy determines that it can recover if the destination endpoint (5) of the current request in the dialog either refreshes its registration or sends the target refresh request in the dialog.

3.6.5.2 SIP Registrar Operation

When a SIP registrar receives a dialog creating request from a SIP endpoint (5), it MUST<28> check if the **Contact** header specifies the GRUU of the endpoint (5), as specified in section 3.3.5.1. If it does, it MUST check whether the SIP endpoint (5) registration is valid and the **Routable** flag is set to "TRUE", as specified in [MS-SIPREGE] section 3.1.2.1. If the binding is absent or the **Routable** flag is set to "FALSE", it SHOULD reject the request with a 430 Flow Failed response and add a **P-Dialog-Recovery-Action** header field. The value of the **P-Dialog-Recovery-Action** indicates the actions that the source endpoint (5) of the currently processed dialog creating the request needs to take to make processing or routing possible for requests originating from, or destined to, that endpoint (5). The value of the **P-Dialog-Recovery-Action** header field MUST be set to "Registration-Route-Set-Update, Dialog-Route-Set-Update".

3.6.5.3 User Agent Operation

The following sections document message processing events and sequencing rules for user agent operations for the dialog state recovery extensions.

3.6.5.3.1 Processing 430 (Flow Failed) Responses

When a user agent receives a 430 Flow Failed response for a mid-dialog request and the response contains a **P-Dialog-Recovery-Action** header field, the user agent MUST examine the value of this field to decide if it needs to perform dialog recovery procedures. Based on the value, the user agent takes the following actions:

- If the **P-Dialog-Recovery-Action** header field contains a **P-Dialog-Recovery-Action** tag, the user agent MUST indicate the failure to the upper layer and then perform registration refresh, as described in [RFC3261] section 10.2.4, on the endpoint (5) that received the 430 Flow Failed response. If registration is successfully refreshed, the user agent MUST execute dialog recovery procedures, as described in section 3.6.5.3.4, on all dialogs associated with the registered endpoint (5) that have dialog recovery enabled. The user agent SHOULD also terminate and re-create all dialogs associated with the registered endpoint (5) that did not have dialog recovery enabled.
- If the **P-Dialog-Recovery-Action** header field contains a single **Dialog-Route-Set-Update** tag, the user agent MUST perform a dialog recovery procedure described in section 3.6.5.3.4. If the refresh request for the dialog recovery procedure results in a successful response, the user agent MUST re-send the request that resulted in the 430 Flow Failed response with the route set and **Request-URI** field populated from the updated route set and remote target fields in the

dialog state. If the refresh request for the dialog recovery procedure does not result in a successful response, the user agent MUST indicate the failure of the original request to the upper layer.

- If as the result of performing dialog recovery procedures, the same request is re-sent two or more times and it again receives a 430 Flow Failed response, the user agent SHOULD stop retrying the same request and report the failure to the user. If the **P-Dialog-Recovery-Action** header field contains a single **Wait-For-Session-Update** tag and the user agent has negotiated a session timer, as described in [RFC4028] on the dialog, it SHOULD start or reset the recovery refresh timer with the interval set to at least the interval it negotiated for the session timer.

When a user agent receives a 430 Flow Failed response for a dialog creating request and the response contains a **P-Dialog-Recovery-Action** header field, the user agent MUST examine the value of this field to decide if it needs to perform dialog recovery procedures<29>. Based on the value, the user agent takes the following actions:

- If the **P-Dialog-Recovery-Action** header field contains a **P-Dialog-Recovery-Action** tag, the user agent MUST indicate the failure to the upper layer and then perform registration refresh, as described in [RFC3261] section 10.2.4, on the endpoint (5) that received the 430 Flow Failed response. If the registration is successfully refreshed, the user agent MUST execute dialog recovery procedures, as described in section 3.6.5.3.4, on all dialogs associated with the registered endpoint (5) that have dialog recovery enabled. The user agent SHOULD also terminate and recreate all dialogs associated with registered endpoints (5) that did not have dialog recovery enabled. Finally, it SHOULD re-send the dialog creating request that originally received the 430 response.

- If as the result of performing dialog recovery procedures, the same request is re-sent two or more times and it again receives a 430 Flow Failed response, the user agent SHOULD stop retrying the same request and report the failure to the user.

3.6.5.3.2 Processing Registration Refresh Responses

When a user agent refreshes endpoint (5) registration, as described in [MS-SIPREGE], and receives a successful response containing a **Presence-State** header field with a **register-action-value** of "added" or "fixed", the user agent SHOULD execute dialog recovery procedures, as described in section 3.6.5.3.4, on all dialogs associated with the registered endpoint (5) that have dialog recovery enabled. The user agent SHOULD also terminate and recreate all dialogs associated with registered endpoints (5) that did not have dialog recovery enabled.

3.6.5.3.3 Processing Mid- Dialog Refresh Requests

When a user agent receives a session refresh request, as described in [RFC4028], on a dialog that has recovery procedures enabled, it SHOULD start or reset the recovery refresh timer with the interval set to at least the interval it negotiated for the session timer.

When a user agent receives a mid-dialog target refresh request, as described in [RFC3261], on a dialog that has recovery procedures enabled, it SHOULD extract the URIs from the **Contact** and **Record-Route** header fields in the request and update the route set and remote target field in the dialog state. If the user agent does not update the route set and remote target, subsequent outgoing requests are sent with a stale route and result in a 430 Flow Failed response.

3.6.5.3.4 Dialog Recovery Procedure

The user agent MUST execute the following steps to recover the dialog state:

1. The user agent MUST construct and send an appropriate target refresh request for the dialog. For example, the user agent sends an UPDATE request for an INVITE dialog or a SUBSCRIBE request for a SUBSCRIBE dialog. The user agent then waits for completion of the associated SIP transaction. The target refresh request MUST carry a value, as specified in section 2.2.1.2, in the **Contact** header field and no **Record-Route** header fields.

2. If the transaction initiated by the target refresh request succeeds, the user agent MUST extract the URIs from the **Contact** and **Record-Route** header fields in the response and update the route set and remote target field in the dialog state.

3. If the target refresh fails with a 430 Flow Failed response that carries a **P-Dialog-Recovery-Action** header field with a single **Wait-For-Session-Update** tag as its value, the user agent SHOULD start or reset the recovery refresh timer with the interval set to at least the interval it negotiated for the session timer.

When the dialog recovery procedure succeeds for a given dialog, the user agent SHOULD also initiate recovery procedures for other dialogs that are logically related to the recovered dialog. For example, the user agent initiates dialog recovery for the dialogs in the conference, as described in [MS-CONFBAS], when it recovers one of them.

3.6.6 Timer Events

3.6.6.1 User Agent Operation

When the recovery refresh timer defined in section 3.6.2.1 fires, the user agent MUST execute dialog recovery procedures, as described in section 3.6.5.3.4.

3.6.7 Other Local Events

None.

3.7 Phone Number Resolution Extensions

[RFC3966] defines a notion of a Local Number as a phone number that is only valid within a certain geographical area or certain part of the telephony network. As specified in [RFC3966] section 5.1.1, Local Numbers SHOULD only be used in the environment where all entities can successfully set up the call by passing this Local Number to dialing software.

This protocol provides a way to create such an environment, and employs a notion of location profile to describe it. Each location profile description carries a set of translation rules that resolve partially specified (local) numbers to identifiers which either route to unique enterprise users or form unique numbers in public telephone networks as defined by International Telecommunications Union Recommendation, contained in [E164]. A translation rule, in turn, is a tuple consisting of the regular expression that matches a subset of local numbers and a replacement pattern that provides an identifier that is no longer tied to a geographical area or part of the telephony network. This type of replacement identifier can be used for routing to a specific enterprise user or for identifying a subscriber in the public telephone network. The regular expressions and replacement patterns are based on .NET Regular Expression Language, as specified in [MC-RegEx]. In addition to defining the location profiles and translation rules that comprise them, this protocol describes a protocol that can be used by the protocol clients to obtain these profiles from the server (2).

3.7.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the

explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

3.7.1.1 User Agent Operation

A user agent compliant with this protocol SHOULD obtain the name of the default location profile to use with the partially specified phone numbers entered by the user. It SHOULD also obtain location profile descriptions with the set of translation rules to convert the partially specified local phone numbers that it receives in SIP messages from other SIP elements.

3.7.1.2 SIP Proxy Operation

A SIP proxy compliant with this protocol SHOULD maintain location profile descriptions for all local geographical areas that it serves. It SHOULD also maintain a database that maps each address-of-record in the domain for which it is responsible to a location profile description, effectively establishing a default location profile for each user.

3.7.2 Timers

None.

3.7.3 Initialization

3.7.3.1 User Agent Operation

A user agent compliant with this protocol SHOULD obtain the name of the default location profile. It SHOULD use the **in-band provisioning** protocol defined in [MS-SIPREGE].

3.7.4 Higher-Layer Triggered Events

3.7.4.1 User Agent Operation

To obtain a location profile description, the user agent MUST send a SIP SERVICE request, as specified in [IETFDRAFT-SIPSOAP-00], with the following parameters:

- The **Request-URI** field and **To** header field **URI** MUST be set to the **locationprofilegruu**, as defined in section 2.2.2, whose address-of-record matches the address-of-record that the user agent represents. If the form of the **locationprofilegruu** that contains the **default** URI parameter is used, the default location profile description for the address-of-record is returned. Otherwise, the location profile description for the profile specified in the **phone-context** parameter is returned.
- The **From** header field URI MUST be set to the address-of-record that the user agent represents.
- The **Accept** header field MUST be set to **application/ms-location-profile-definition+xml**.
- Other fields of the SERVICE request MUST be set as described in [RFC3261] and [IETFDRAFT-SIPSOAP-00], and the request MUST be sent using the rules in [RFC3261].

3.7.5 Message Processing Events and Sequencing Rules

3.7.5.1 SIP Proxy Operation

When a SIP proxy compliant with this protocol receives a SERVICE request targeted to a URI built according to **location-profile-gruu** syntax, as described in section 2.2.2, whose address-of-record matches an address-of-record in the domain for which this SIP proxy is responsible, it MUST process the request as follows:

1. Perform standard routing procedures against the **Request-URI** field, as described in [RFC3261]. One of the standard routing procedures in [RFC3261] specifies that it MUST respond with a 404 response if the address-of-record in the **Request-URI** field does not exist in the domain that the proxy is responsible for.

2. Extract the name of the location profile from the **location-profile-gruu** URI. If the **location-profile-gruu** URI contains the **default** parameter, the proxy SHOULD consult its internal database to determine the name of the location profile whose address-of-record matches the address-of-record in the **location-profile-gruu** URI. Otherwise, it MUST extract the name of the location profile from the **phone-context** URI parameter. If neither the **default** or **phone-context** parameters are present in the **location-profile-gruu** URI, the SIP proxy MUST reject the request with a 485 Ambiguous response.

3. The SIP proxy MUST then check its location profile descriptions database and attempt to locate the profile with the name extracted in Step 2. If the location profile description with the given name does not exist, the SIP proxy MUST reject the request with a 404 Not Found response. Otherwise, it MUST read the location profile description from its database and form an XML document according to the syntax described in section 2.2.7.

4. The proxy MUST form and send the response to the SERVICE request as described in [RFC3261] and [IETFDRAFT-SIPSOAP-00] and insert the following fields:

    1. The **Content-Type** header field MUST be set to **application/ms-location-profile-definition+xml**.

    2. The body of the response MUST be set to the location profile description XML document created in step 3.

3.7.6 Timer Events

None.

3.7.7 Other Local Events

None.

3.8 Extensions for Call Processing and Routing Based on Routing Script Preamble and Call Designation Parameters

This protocol specifies the Routing Script Preamble mechanism for protocol client endpoints (5) to publish rules for routing INVITEs targeted to the address-of-record of the user the user agent represents. The preamble MUST be published by the user agent into the **routing** category, as specified in [MS-PRES], and is used for all audio INVITEs except those that are exposed to policy restrictions on the server (2).

The user agent can publish preambles into multiple instances of the routing category. The different preambles MUST meet the following conditions:

- Each preamble publication MUST be in accordance with the preamble **XSD**.
- **List** elements with the same name can appear in multiple instances. The **name** attribute value of all **list** elements occurring in the same instance MUST be unique.
- The **name** attribute values of all other elements MUST be unique within that element type. For example, the preambles cannot contain two **wait** elements with the same name.

If any of the preceding conditions are not met, a server (2) that is a SIP proxy authorized for the domain of the target user's address-of-record SHOULD use a default routing script that routes only to the registered endpoints (5) of the target address-of-record.

If the server (2) finds multiple instances that are valid, it MUST generate an aggregated preamble that is then used for routing. If multiple **list** elements with the same name are found, the aggregated preamble SHOULD contain one **list** with that name containing all of the **target** elements from different instances. If the **version** attribute of the instances are different, the aggregated preamble's version MUST be the highest **version** attribute value among all instances.

The preamble published by the protocol client SHOULD match a corresponding script installed on the server (2). If no match is found, a server (2) that is a SIP proxy authorized for the domain of the target user's address-of-record SHOULD use a default routing script that routes only to the registered endpoints (5) of the target address-of-record.

If any element required by the script is not present in the preamble, the server (2) can reject the INVITE with a 480 response.

3.8.1 Abstract Data Model

None.

3.8.2 Timers

3.8.2.1 Registered Endpoints Timer

If the call is being routed to the registered endpoints (5) whose address-of-record matches the address-of-record in the **Request-URI** field, a registered endpoints timer is started. The amount of time to wait is defined by the **wait** element named **total** that is defined in the preamble. If no preamble is published, the default wait time is 20 seconds. If a preamble is published but a **wait** element named **total** is not defined, the default wait time is 15 seconds.

3.8.2.2 Call Forwarding Timer

If call forwarding is enabled, which means that the **enablecf** flag is set, and the call is routed to the target in the **forwardto** list, the call forwarding timer is started for 60 seconds.

3.8.2.3 Primary User Timer

This section follows the product behavior described in endnote <30>.

If team ringing is enabled, a primary user timer is started instead of the registered endpoints timer. The amount of time to wait is defined by the **wait** element named **user** that is defined in the preamble. If a preamble is published but a **wait** element named **user** is not defined, the default wait time is 15 seconds.

3.8.2.4 Secondary Target Timer

This section follows the product behavior described in endnote <31>.

If the call is being routed to the targets in the **team** or **delegates** list, a secondary target timer is started. The amount of time to wait is defined by the **wait** element named **team2** that is defined in the preamble.

3.8.3 Initialization

The default routing behavior for a SIP proxy authorized for the domain of the target user's address-of-record if no preamble is published by the protocol client or if the preamble name and version do not match is to ring registered endpoints (5) for 20 seconds and then forward the call to the target user's voice mail, if it is configured.

3.8.4 Higher-Layer Triggered Events

None.

3.8.5 Message Processing Events and Sequencing Rules

3.8.5.1 Call Processing and Routing Elements

User agents that are publishing can publish any preamble that is in accordance with the preamble XSD. However, the server (2) SHOULD only act on a specific list of elements, and other elements MUST be ignored. The server (2) that is a SIP proxy authorized for the domain of the target user's address-of-record SHOULD apply the routing rules based on the preamble only for INVITEs that meet one of the following criteria:

- The content type is "application/SDP" and the **Session Description Protocol (SDP)** body includes **audio**.
- The content type is "application/ms-conf-invite" and the XML body indicates that audio is available.<32>
- The content type is "multipart/MIME" and at least one part contains an SDP body that includes audio.<33>

The construction of the INVITE requests with an "application/SDP" content type is described in **[RFC3246],** the "multipart/MIME" content type is defined in **[RFC2046]**.

All other INVITEs SHOULD be routed as specified in [RFC3261]**.** The routing mechanism specified in this section is applicable only if one of the preceding three conditions is met.

An INVITE whose content type is **application/ms-conf-invite** and the XML body indicates that audio is available is called an **audio app-invite**.

3.8.5.1.1 Routing Element Name and Version

The **routing** element has **name** and **version** attributes that SHOULD be one of the supported values. The supported values for these attributes are the following:

- The **name** attribute value is **rtcdefault** and the **version** attribute is 1.
- The **name** attribute value is **rtcdefault** and the **version** attribute is 2.<34>

3.8.5.1.2 Routing Element Flags

The server (2) MUST use the **flags** element named **clientflags** to determine which features are currently enabled or disabled. Any other **flags** element or flags in **clientflags** element MUST be ignored by the server (2). The following table describes how each flag is used. The "Working hours only" column indicates if the flag can be used in conjunction with the **work_hours** flag.

Flag name	Usage	Working hours only
block	Causes all calls to the user to fail. This flag SHOULD be the only value present in a preamble intended to block inbound calls.	No
work_hours	Indicates that the routing logic SHOULD only be applied if the current time falls within the **calendarData** publication, as specified in [MS-PRES].	Not Applicable
forward_immediate	Causes calls to be forwarded to the address specified in the **forwardto** list if the **enablecf** flag is also present, or to voice mail if the **enablecf** flag is not present.	Yes
simultaneous_ring	Causes the first target listed in the **list** element named **simultaneous_ring** to be called at the same time any registered endpoints (5) are called.	Yes
enablecf	Enables call forwarding to the target in the **forwardto** list. This flag is used to toggle between activating voice mail and call forwarding.	Yes
delegate_ring<35>	Indicates that the call SHOULD be forked to the targets specified in the **delegates** list. This flag SHOULD NOT be used in combination with **team_ring**. If **team_ring** is set at the same time, **team_ring** takes precedence. This flag is applicable only if the **routing** element version is 2.	Yes
team_ring<36>	Indicates that the call SHOULD be forked to the targets specified in the **team** list. This flag is applicable only if the **routing** element version is 2.	Yes
skip_primary<37>	Indicates that the registered endpoints (5) and simultaneous ring device of the **callee** SHOULD NOT be rung unless the call is coming from or transferred by a URI in the **breakthrough** or **delegates** list. This flag is applicable only if the **routing** element version is 2. This flag is applicable only if the **delegate_ring** flag is also set.	Yes
forward_audio_app_invites<38>	Indicates that audio app-invites, as described in section 3.8.1, SHOULD be routed in the same way as all other audio invites to this user. This flag is applicable only if the **routing** element version is 2.	Yes
e911active<39>	Causes all routing rules to be suspended and calls to be forked only to registered endpoints (5). This is set by the client when the user makes an emergency call.	No

3.8.5.1.3 Routing Element Wait Times

The server (2) MUST use only the **wait** element with names defined as follows. All other **wait** elements are ignored.

Wait time name	Usage
total	Number of seconds to wait for the called party to answer. Used when routing version is 1 or when version is 2 and **team_ring** and **delegate_ring** flags are not set.
user<40>	Number of seconds to ring the user's registered endpoints (5) and simultaneous ring device before ringing the team. Applicable only if routing version is 2.
team1<41>	Reserved for future use. SHOULD be ignored.
team2<42>	Number of seconds to ring the team or **delegates**. Applicable only if routing version is 2.

3.8.5.1.4 Routing Element Lists

The server (2) MUST use only the lists specified in the following table. These lists can be empty if there is no relevant data provided by the user.

List name	Usage
forwardto	This list contains the URI that SHOULD be used when the user has selected call forwarding, which means that the **enablecf** is set under **clientflags**. Even though the **list** element syntax allows more than one item, the list SHOULD contain only one entry. If more than one entry is present, the server (2) SHOULD only use the first destination.
simultaneous_ring	This list contains the URI that defines a device that SHOULD ring at the same time as the user's registered devices. Even though the **list** element syntax allows more than one item, the list SHOULD contain only one entry. If more than one entry is present, the server (2) SHOULD only use the first destination.
team<43>	This list contains the URIs corresponding to the team members of the user. This list is applicable only if the routing version is 2.
delegates<44>	This list contains the URIs corresponding to the delegates of the user. This list is applicable only if the routing version is 2.
first_delegate<45>	Reserved for future use. SHOULD be ignored.
breakthrough<46>	List of identities that can ring the user directly even when the **skip_primary** flag is set. This is applicable only if routing version is 2.
add_voice<47>	Reserved for future use. SHOULD be ignored.

3.8.5.2 Incoming INVITE Processing

When an INVITE arrives at the SIP proxy authorized for the address-of-record in the Request-URI field, the proxy MUST process the request based on the preamble published for that address-of-record.

3.8.5.2.1 Ms-Sensitivity Header

The presence of the **Ms-Sensitivity** header field in the incoming request is used to tailor how the request is routed.

Level of sensitivity	Usage
Normal	This is the default value. All possible destinations will be selected by the server (2) subject to the routing rules as specified by the preamble.
normal-no-diversion	This has the effect of disabling voice mail and call forwarding. If the **Ms-Sensitivity** header has this value, the server (2) MUST NOT route the call to voice mail or the call forwarding target defined in the **forwardto** list or to the targets defined in the **team** list. Note that calls to the simultaneous ring target are not considered a diversion and the call MUST be forwarded to the simultaneous ring target if present.
Private	Reserved for future use. MUST be treated the same way as **Normal**.
private-no-diversion	MUST be treated the same way as **normal-no-diversion**.

3.8.5.2.2 Rules for Handling the INVITE

The SIP proxy authorized for the address-of-record in the Request-URI field SHOULD perform the following steps in order when handling the INVITE request:

1. If the **block** flag is set, a SIP error SHOULD be returned, as specified in [MS-OCER] section 10.3, and further processing of rules SHOULD be stopped.
2. If the **e911active** flag is set, the proxy SHOULD route the call only to registered endpoints (5). The registered endpoints timer SHOULD NOT be started and further processing of rules SHOULD be stopped.<48>
3. If the INVITE is an audio app-invite and the **forward_audio_app_invites** flag is not set, the proxy SHOULD route the call only to registered endpoints (5). The registered endpoints timer SHOULD NOT be started and further processing of rules SHOULD be stopped.<49>
4. If the INVITE is targeted at the **private line** of the user, the call SHOULD be processed as specified in section 3.8.5.2.2.4.
5. If the INVITE was routed to the user as a result of team or delegate ringing processing for some other user, the proxy SHOULD route the call only to registered endpoints (5) and the registered endpoints timer SHOULD NOT be started. Further processing of rules SHOULD be stopped.<50>
6. If the address-of-record in the URI of the **From** or **Referred-By** header fields, as defined in [RFC4235] section 4.1.5, is present in the **breakthrough** list, the call SHOULD be routed to the primary targets as specified in section 3.8.5.2.2.1, and further processing of rules SHOULD be stopped.<51>
7. If the **work_hours** flag is set and the current time is outside the working hours in the **calendarData** publication, as specified in [MS-PRES], the call MUST be forked to the registered endpoints (5) whose address-of-record matches the address-of-record in the **Request-URI** field, except that **Do-Not-Disturb** presence state MUST be handled as specified in step 10.
8. If the **team_ring** flag is set, team ringing SHOULD be processed as specified in section 3.8.5.2.2.3 and further processing of rules SHOULD be stopped<52>

9. If the **delegate_ring** flag is set, delegate ringing SHOULD be processed as specified in section 3.8.5.2.2.2 and further processing of rules SHOULD be stopped.<53>

10. If the user's presence state published in the presence database, as described in [MS-PRES], is "Do-Not-Disturb" and the **caller** is not in the container **300**, as specified in [MS-PRES], of the target user, the call MUST be routed to the target user's voice mail and further processing of rules SHOULD be stopped. If the call cannot be routed to voice mail because of **Ms-Sensitivity** header field value considerations described in section 3.8.5.2.1, a response indicating failure SHOULD be returned.

11. If none of the preceding conditions apply, the call MUST be routed to primary targets as specified in section 3.8.5.2.2.1.

3.8.5.2.2.1 Ringing Primary Targets

If in the processing of the INVITE based on the routing rules, the proxy decides to ring the primary targets, the following actions MUST be taken:

- If the **forward_immediate** flag is set in the protocol client flags:
  - The call SHOULD be routed to the destination in the **forwardto** list or voice mail depending on whether the **enablecf** flag is set.
  - If a **simultaneous_ring** target exists, it MUST NOT be honored if the **forward_immediate** flag is set.
  - If the call was routed to the target in the **forwardto** list, the call forwarding timer MUST be started. If the call cannot be routed because of the **Ms-Sensitivity** header field value considerations described in section 3.8.5.2.1, a response indicating failure SHOULD be returned.
- Otherwise, if the **forward_immediate** flag is not set in the protocol client flags:
  - The call MUST be forked to the registered endpoints (5) whose address-of-record matches the address-of-record in the **Request-URI** field.
  - If the **simultaneous_ring** flag is set, the INVITE MUST be routed to the target specified in the **simultaneous_ring** list. The proxy MUST then start the registered endpoints timer.

3.8.5.2.2.2 Delegate Ringing

This section follows the product behavior described in endnote <54>.

If in the processing of the INVITE based on the routing rules, the proxy decides to honor delegate ringing, the following actions MUST be taken:

- If the address-of-record in the URI of the **From** or the **Referred-By** header field is present in the **delegates** list, the INVITE MUST be routed to primary targets, as specified in section 3.8.5.2.2.1.
- If the user's presence state published in the presence database, as specified in [MS-PRES], is "Do-Not-Disturb", the routing rules and the caller are not in container **300**, as specified in [MS-PRES], of the target user, the call MUST be forked to the targets present in the **delegates** list and the secondary target timer MUST be started.
- If the user's presence state is not "Do-Not-Disturb", the call MUST be routed to all the registered endpoints (5) of the user and the primary user timer MUST be started. <55>

- If the user's presence state is not "Do-Not-Disturb", the call MUST be routed to all of the targets present in the **delegates** list. The secondary target timer MUST be started.

3.8.5.2.2.3 Team Ringing

This section follows the product behavior described in endnote <56>.

If in the processing of the INVITE based on the routing rules, the proxy decides to honor team ringing, the following actions MUST be taken:

- If the address-of-record in the URI of the **From** field or the **Referred-By-URI** field is present in the **team** list, the INVITE MUST be routed to primary targets as specified in section 3.8.5.2.2.1.
- If the user's presence state published in the presence database, as described in [MS-PRES], is "Do-Not-Disturb", the routing rules and the caller are not in container **300**, as specified in [MS-PRES], of the target user, the call MUST be forked to the targets present in the **team** list and the secondary target timer MUST be started.
- If the user's presence state is not "Do-Not-Disturb", the call MUST be routed to all the registered endpoints (5) of the user. The primary user timer MUST be started.

3.8.5.2.2.4 Ringing Private Line

This section follows the product behavior described in endnote <57>.

If the incoming INVITE is targeted at the private-line of the user, the call MUST be forked to the registered endpoints (5) whose address-of-record matches the address-of-record of the target. In addition, if the **simultaneous_ring** flag is set, the INVITE MUST be routed to the target specified in the **simultaneous_ring** list. The proxy MUST then start the registered endpoints timer.

3.8.5.3 Handling 303 Response

Any destination to which the call is forked can send a 303 Proxy Redirect response back to the server (2). [IETFDRAFT-RCDPR-303-01] specifies how this response is handled.

3.8.5.4 Handling 605 Response

Any destination to which the call is forked can send a 605 Decline All response back to the server (2). [IETFDRAFT-SF-605-01] specifies how this response is handled.

3.8.5.5 Handling 415 Response

This section follows the product behavior described in endnote <58>.

If a SIP proxy compliant with this protocol receives a 415 response from one of the targets to which the proxy forked the call, the proxy MUST handle the response as follows:

1. If the request that was sent to the target did not contain a body with a "multipart/MIME" content type, no special processing is applied and the 415 response MUST be handled as any 4XX response, as described in [RFC3261], section 16.7.
2. If **multipart/MIME** retry has been attempted for this target, the 415 response MUST be handled as any 4XX response.
3. If any **accept** header in the response indicates that the UAS supports **multipart/MIME**, no special processing is applied and the 415 response MUST be handled as any 4XX response.

4. If any part of a **multipart/MIME** body has a **Content-Disposition** header field with an **ms-proxy-2007fallback** parameter and that part has SDP content with **audio m-line**, the proxy takes the following actions:

    1. The proxy MUST re-send the INVITE to the target with only the SDP body, and

    2. The proxy MUST update its call context for that target to indicate that **multipart/MIME** retry has been attempted for this target.

The **multipart/MIME** content type is defined in **[RFC2046].**

3.8.5.6 Handling 2XX Responses

A SIP proxy compliant with this protocol SHOULD handle 2XX responses according to proxy behavior described in [RFC3261] section 16.7. In addition, the CANCEL requests sent out as a result of a 2XX response SHOULD have an **ms-acceptedby** parameter in the Reason header field. The **ms-acceptedby** parameter value SHOULD be set to the address-of-record of the destination user agent that sent the 2XX response.

3.8.5.7 Other Responses

All other responses SHOULD be treated as specified in [RFC3261].

3.8.5.8 Generating 199 Response

This section follows the product behavior described in endnote <59>.

If a proxy receives a non 2XX final response from one of the targets and the SIP proxy decides to keep or drop the final response, the proxy SHOULD generate a 199 response in accordance with [IETFDRAFT-RCITD-199-01] if:

1. A 18X response from that target had been proxied through to the caller, and

2. A 199 response was not already sent for this target.

3.8.5.9 1XX Responses Generated

Any time the SIP proxy authorized for the domain in the address-of-record of the **Request-URI** field processes an audio call as described in this protocol, a 183 response with an **Ms-Forking** header field MUST be sent back to the caller.

Any time the request was sent to one or more registered endpoints (5), a 101 response MUST be sent back to the caller.

Any time the request was forwarded to a target other than the registered endpoints (5), a 181 response MUST be sent back to the caller.

3.8.5.10 History-Info Header Field Processing

This section follows the product behavior described in endnote <60>.

When the SIP proxy authorized for the domain in the address-of-record of the Request-URI field processes the INVITE request using the published preamble, as described in section 3.8.5.2, it MUST process the **History-Info** header field in the request, if present, as follows:

1. The proxy MUST perform basic validation of the **History-Info** header field entries according to the syntax in section 2.2.16 so that it can extract the value of the **hi-index** parameter of the last entry. If validation of the **History-Info** header field fails, the proxy MUST stop further processing. The proxy can reject the request with a 400 response.

2. If validation of the **History-Info** header field succeeds, the proxy MUST store the value of the **History-Info** header field except the last entry, which is the entry targeted at the address-of-record for which the proxy processes the INVITE request, in the INVITE transaction processing context.

3. The proxy MUST also extract the value of the **hi-index** parameter from the last entry and store it in the INVITE transaction processing context.

If a **History-Info** header field is not present in the request, the proxy MUST store an empty **History-Info** header field and **hi-index** parameter value of 1 in the INVITE transaction processing context.

The proxy MUST also initialize a value of branch index to 1 in the INVITE transaction processing context.

When, as part of processing the INVITE transaction, the INVITE request is proxied or forwarded to any destination, the SIP proxy MUST copy the **History-Info** header field that it stored in the INVITE transaction processing context to the proxied or forwarded request and append one or more **History-Info** header field entries as follows:

- If the destination is a registered endpoint (5) whose address-of-record matches the address-of-record of the target of the original INVITE request or the INVITE request is forked to the destination at the same time as it is being sent to the registered endpoints (5), the proxy MUST add one **History-Info** header field entry with a **hi-targeted-to-uri** parameter set to the SIP URI of the registered endpoint (5) address-of-record, and a **hi-index** parameter set to the current value of the **hi-index** parameter in the INVITE transaction processing context.
- If the destination is a registered endpoint whose address-of-record matches the address-of-record of the target and the request was targeted at the private line of the user, the proxy SHOULD add a **hi-ms-line-type** parameter with the value "**private**"<61>.
- For other destinations, the proxy MUST add two **History-Info** header field entries:

  1. An entry with the parameters set as follows:

     - **hi-targeted-to-uri** value MUST be set to the SIP URI of the address-of-record of the target in the original INVITE request.
     - **hi-index** parameter value MUST be set to the current value of the **hi-index** parameter in the INVITE transaction processing context.
     - **hi-ms-retarget-reason** parameter value MUST be set to the value of **team-call** if the current destination was selected as the result of team ringing, or to the value of **delegation** if the current destination was selected as the result of delegate ringing, or to the value of **forwarding** in all other cases.
     - **reason** parameter MUST NOT be set if the request is being sent to a registered endpoint (5) of the target or if the INVITE request is being sent to the current destination while any previous forks to registered endpoints (5) are still active. The **reason** parameter MUST be set to the value "SIP;cause=303;text=Redirect**"** if the INVITE request is forwarded to the current destination as the result of the processing of a 303 response, as described in

section 3.8.5.3, or with the value of "SIP;cause=302;text=Moved Temporarily" if the INVITE request is forwarded to the current destination for any other reason.

2. An entry with the parameters set as follows:

   - **hi-targeted-to-uri** parameter value MUST be set to the SIP URI of the address-of-record of the destination.
   - **hi-index** parameter value MUST be set to the concatenation of a) the current value of the **hi-index** parameter in the INVITE transaction processing context , b) the "." separator, and c) the current value of the branch index in the INVITE transaction processing context.
   - The proxy MUST then increment by 1 the value of the branch index in the current INVITE transaction processing context.

When, as part of processing the INVITE transaction, the proxy generates a 181 response, it MUST add a **History-Info** header field with a single entry with the parameters set as follows:

- **hi-targeted-to-uri** parameter value MUST be set to the SIP URI of the address-of-record of the target in the original INVITE request.
- **hi-index** parameter value MUST be set to the value of 1.
- **hi-ms-retarget-reason** parameter value MUST be set to the value of **team-call** if the 181 response was generated when the original INVITE was sent to the destination as the result of team ringing, or to the value of **delegation** if the 181 response was generated when the original INVITE was sent to the destination as the result of delegate ringing, or to the value of **forwarding** in all other cases.
- **reason** parameter MUST NOT be set if the INVITE request is being sent to the current destination while any previous fork to registered endpoints (5) are still active. The **reason** parameter MUST be set to the value of "SIP;cause=303;text=Redirect**"** if the INVITE request is forwarded to the current destination as the result of the processing of a 303 response, as described in section 3.8.5.3, or with the value of "SIP;cause=302;text=Moved Temporarily" if the INVITE request is forwarded to the current destination for any other reason.

3.8.6 Timer Events

3.8.6.1 Registered Endpoint Timer Expiry

When the registered endpoint timer expires, the following actions MUST be executed by the server (2):

- If the **Ms-Sensitivity** header field value does not contain **no-diversion** and the incoming INVITE is not targeted at the private line of the user and the **enablecf** flag is set:

  1. The call MUST be forwarded to the destination defined in the **forwardto** list.
  2. A 181 response MUST be sent back to the caller indicating that the call is being forwarded.
  3. The call forwarding timer MUST be started.

- If the **Ms-Sensitivity** header field value does not contain **no-diversion** and the **enablecf** flag is not set and voice mail is configured for the callee:

  1. The call MUST be forwarded to voice mail, and

2. A 181 response MUST be sent back to the caller indicating that the call is being forwarded.

3.8.6.2 Call Forwarding Timer Expiry

When the **call forwarding** timer expires, the call MUST be forwarded to the user's voice mail if voice mail is configured for the user.

3.8.6.3 Primary User Timer Expiry

This section follows the product behavior described in endnote <62>.

When the primary user timer expires and the **team ring** flag is set, the call MUST be routed to the targets specified in the team list and the secondary target timer MUST be started. Existing transactions MUST NOT be cancelled.

When the primary user timer expires and the **delegate ring** flag is set, the call MUST be routed to the targets specified in the **delegates** list and the secondary target timer MUST be started.<63> Existing transactions MUST NOT be cancelled.

3.8.6.4 Secondary Target Timer Expiry

This section follows the product behavior described in endnote <64>.

When the secondary target timer expires, all existing transactions MUST be cancelled. If the **enablecf** flag is set, the call MUST be routed to the target specified in the **forwardto** list and the call forwarding timer MUST be started. If the **enablecf** flag is not set, the call MUST be forwarded to the user's voice mail, if one is configured.

3.8.7 Other Local Events

None.

3.9 Extensions for Federation and Public IM Connectivity

As specified in section 2.2.14, this protocol defines the **ms-edge-proxy-message-trust** header field. The following sections specify the header parameters, their values, and the message processing events for this header field.

3.9.1 Abstract Data Model

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

3.9.1.1 ms-source-type parameter

The header field can contain the **ms-source-type** parameter. This parameter represents the type of connectivity between the remote user or peer server (2) and the enterprise SIP network:

- A parameter value of **AuthorizedServer** can be used to indicate that the peer server (2) is authorized to represent a public IM provider.

- A parameter value of **AutoFederation** can be used to indicate that the **From** user's SIP domain is authorized for federation (2) and resolves through a DNS SRV record to a peer server (2) FQDN (1).
- A parameter value of **DirectPartner** can be used to indicate that the **From** user's SIP domain and the peer server (2) is authorized for direct federation (2).
- A parameter value of **EdgeProxyGenerated** can be used to indicate the SIP message was generated by a server (2) that is responsible for processing messages from SIP elements outside of the enterprise network.
- A parameter value of **InternetUser** can be used to indicate that the SIP message is received from a remote user.

3.9.1.2 ms-ep-fqdn parameter

The header field can contain the **ms-ep-fqdn** parameter. The parameter value can be used to represent the FQDN (1) of the server (2) that adds the header field.

3.9.1.3 ms-source-verified-user parameter

The header field can contain the **ms-source-verified-user** parameter. If the **ms-source-type** parameter value is equal to "InternetUser", the value of the **ms-source-verified-user** parameter MUST be set to "verified".

If the **ms-source-verified-user** parameter is added:

- A parameter value of "verified" can be used to indicate that the federated partner or public IM provider is trusted to verify the **From** user's identity and that the federated partner or public IM provider has verified the **From** user identity.
- A parameter value of "unverified" can be used to indicate that either the federated partner or public IM provider is not trusted to verify the **From** user identity or that the federated partner or public IM provider has not been able to verify the **From** user identity.

3.9.1.4 ms-source-network parameter

If the protocol client needs to be informed that the message is from a federated partner or a public IM provider, the header field MUST contain the **ms-source-network** parameter. This parameter MUST NOT be added if the **ms-source-type** parameter exists and its value is equal to "InternetUser". If the **ms-source-network** parameter is added, one of the following two items applies:

- A parameter value of "federation" MUST be used to indicate that the SIP message is from a federated user.
- A parameter value of "publiccloud" MUST be used to indicate that the SIP message is from a public IM user.

If the header field does not contain the **ms-source-network** parameter, this means that the SIP message is from a user that belongs to the same enterprise.

3.9.2 Timers

None.

3.9.3 Initialization

None.

3.9.4 Higher-Layer Triggered Events

None.

3.9.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are as specified in [RFC3261].

3.9.5.1 Server Behavior

If the server (2) forwards any message, either a request or a response, to the client that was originally received from a SIP element located outside of the enterprise network, it SHOULD insert an **ms-edge-proxy-message-trust** header field into the message. This header field provides information about source of the SIP element as determined by the server (2) that is responsible for processing messages from SIP elements outside of the enterprise network. The syntax of the **ms-edge-proxy-message-trust** header field is described in section 2.2.14.

3.9.5.2 Client Behavior

The following section specifies protocol client behavior based on parameter values contained in the **ms-edge-proxy-message-trust** header field, as follows:

- If it is identified through the SIP **NOTIFY** message that the user is a federated user or a public IM user, an indication to this effect for this user can be displayed in the contact list.
- If one or more parties in a conversation are users that do not belong to the same enterprise, an indication to this effect can be displayed in the conversation window.

3.9.6 Timer Events

None.

3.9.7 Other Local Events

None.

3.10 Extensions for Remote Users

As specified in section 2.2.15, this protocol defines the **ms-user-logon-data** header field. The following sections specify the header parameters, their values, and the message processing events for this header field.

3.10.1 Abstract Data Model

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

If this header field is present, the header field value MUST be "RemoteUser".

3.10.2 Timers

None.

3.10.3 Initialization

None.

3.10.4 Higher-Layer Triggered Events

None.

3.10.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are as specified in [RFC3261].

3.10.5.1 Server Behavior

When a server (2) forwards any message, either a request or a response, to the client that connects to it from the outside of the enterprise network, it SHOULD insert an **ms-user-logon-data** header field into the message with a value of "RemoteUser".

3.10.5.2 Client Behavior

The following section specifies protocol client behavior based on the **ms-user-logon-data** header field.

If this header field is present in the reply to a REGISTER request and has a value of "RemoteUser", the protocol client SHOULD treat the requester as an external protocol client connecting from outside of the enterprise network. Under this condition, the protocol client SHOULD do the following:

- Use a **Web service URL (Uniform Resource Locator)** that is accessible from the public Internet for distribution list expansion, address book download, and calendar services.
- Assume that it does not have direct media connectivity to the enterprise network.

3.10.6 Timer Events

None.

3.10.7 Other Local Events

None.

3.11 Extensions for Logging and Monitoring

This section follows the product behavior described in endnote <65>.

As specified in section 2.2.11, this protocol defines the **ms-correlation-id** header field. The following sections specify the header parameters, their values, and the message processing events for this header field.

3.11.1 Abstract Data Model

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

If an **ms-correlation-id** header field is present, it MUST contain a UUID, as defined in [RFC4122] Section 3. If the same value of the **ms-correlation-id** header field is included in messages for multiple SIP dialogs, those dialogs are considered to be correlated. No specific semantics are defined for which dialogs can be considered correlated; the correlation identifier is intended solely as a hint which log analysis and diagnostic tools can use to infer a relationship between two otherwise-unrelated dialogs.

For example, consider Client B that acts as a back-to-back user agent. This client receives an INVITE from Client A, and sends another INVITE to the final recipient of the message, Client C. Client B generates a new random correlation identifier, and includes the ID in the INVITE to Client C and the response to Client A. Once Client C responds, two otherwise-unrelated dialogs, D1 and D2, have been established. Server (2) processing for both dialogs is unaffected by the additional header, but a server (2) SHOULD capture and store the correlation identifier in a log. A log analysis or diagnostic tool later run on the log SHOULD use the correlation identifier to identify that dialogs D1 and D2 are related, and hence that Client A and Client C were in communication via the intermediary back-to-back user agent.

If the header is absent, or the value of the header is not used by any other dialog, the dialog is not correlated.

3.11.2 Timers

None.

3.11.3 Initialization

None.

3.11.4 Higher-Layer Triggered Events

3.11.4.1 Client Behavior

If the SIP endpoint (5) creates two dialogs that are related to each other, it SHOULD generate a UUID using a procedure compatible with [RFC4122] Section 4, and add an **Ms-Correlation-Id** header field with this value to the INVITE or REFER messages that created the dialogs.

3.11.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are as specified in [RFC3261].

3.11.5.1 Client Behavior

If the SIP endpoint (5) receives an INVITE or REFER containing an **Ms-Correlation-Id** header field, and in response it wishes to create another dialog that is related to the dialog created by that request, it SHOULD add an **Ms-Correlation-Id** header field with the same value it received to the INVITE or REFER message is uses to create the second dialog.

If the SIP endpoint (5) receives an INVITE or REFER without an **Ms-Correlation-Id** header field, and in response it wishes to create another dialog that is related to the dialog created by that request, it SHOULD generate a UUID using a procedure compatible with [RFC4122] Section 4 and add an **Ms-Correlation-Id** header field with this value both to its final response to the message received, and to the INVITE or REFER request it uses to create the second dialog.

3.11.5.2 Proxy Behavior

When a SIP proxy that logs dialog creation events processes a dialog creating request or final response to a dialog creating request that has an **Ms-Correlation-Id** header field present and the value in this field is a valid UUID, as defined in [RFC4122] section 3, it can record the value in the log. If the value is not a valid UUID, the proxy SHOULD ignore the presence of the header.

3.11.6 Timer Events

None.

3.11.7 Other Local Events

None.

3.12 Extensions for Call Context

This section follows the product behavior described in endnote <66>.

This protocol specifies the call context mechanism for protocol client and server (2) endpoints (5) to create notes related to a given call that can be sent to another party receiving the INVITE that creates a new call. There are a number of pieces of information contained within the call context content that helps the endpoint (5) to correlate and render the call context data and notes to the user. The call context data is carried within the related INVITE request as a MIME type in the message body of the request.

3.12.1 Abstract Data Model

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

User agents creating notes in relation to a call can convey those text notes using the call context data type.

3.12.2 Timers

None.

3.12.3 Initialization

None.

3.12.4 Higher-Layer Triggered Events

None.

3.12.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are as specified in [RFC3261].

3.12.5.1 Client Behavior

The following section specifies client behavior based on the **ms-conversation-context+xml** content type. The following apply:

- Can only use the SIP INVITE request to convey call context data.
- Can only include a single call context MIME body in the request.
- MUST set the content type to **application/ ms-conversation-context+xml** for the MIME body conveying call context data.
- The **id** element for each call context body MUST be unique among all call context data created by the server (2), and MUST appear only once in the call context data.
- The **from** element MUST be present in the call context data and appear only once.
- The **uri** child element MUST be present within the **from** element.
- The **displayName**, **onBehalfUri**, and **onBehalfDisplayName** child elements can appear in the **from** element and SHOULD be present if the data is available at the server (2) for that call, but MUST NOT appear more than once each.
- The **to** element MUST be present in the call context data and appear only once.
- The **uri** child element MUST be present within the **to** element.
- The **displayName**, **onBehalfUri**, and **onBehalfDisplayName** child elements can appear in the **to** element and SHOULD be present if the data is available at the server (2) for that call, but MUST NOT appear more than once each.
- The **participants** element MUST be present in the call context data and appear only once and MUST contain one or more **participant** elements.
- A **participant** element MUST be present for the author of the call context data.
- Other **participant** elements can be present for each party involved with the call.
- The **uri** child element MUST be present within the **participant** element.
- The **displayName**, **onBehalfUri**, and **onBehalfDisplayName** child elements can appear in the **participant** element, SHOULD be present if the data is available at the server (2) for that call, but MUST NOT appear more than once each.
- The **date** element MUST be in UTC format, MUST be present in the call context data and MUST appear only once.
- The **conversationId** element MUST be present in the call context data, MUST appear only once, and MUST be unique among all call context data created by the server (2).
- The **dataFormat** element MUST be present in the call context data, MUST appear only once, and MUST have a value of "text/plain".

- The **contextData** element MUST be present in the call context data.
- The **mode** element can be present one or more times in the call context data, each time with a unique value, and SHOULD consist of one of the following values:
  - audio
  - video
  - im
  - applicationSharing

3.12.5.2 Server Behavior

The following section specifies protocol server (2) behavior based on the **ms-conversation-context+xml** content type. The following apply:

- Can ignore call context data that does not comply with the **application/ ms-conversation-context+xml** XSD or is conveyed through other SIP messages other than the INVITE request to initiate a new dialog.
- Can ignore call context data with a **dataFormat** element value other than "text/plain".
- Can ignore call context data with a **mode** element that has a value other than one of the following:
  - audio
  - video
  - im
  - applicationSharing

3.12.6 Timer Events

None.

3.12.7 Other Local Events

None.

3.13 Safe Call Transfer Extension

This section follows the product behavior described in endnote <67>.

The safe call transfer extension tailors the routing behavior while transferring calls using the REFER request. Using this extension, a user agent transferring calls can request that the transferee disable call forwarding and voice mail for the triggered INVITE request.

3.13.1 Abstract Data Model

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

adhere to this model as long as their external behavior is consistent with that described in this document.

The header field can contain the **ms-ep-fqdn** parameter. The parameter value can be used to represent the FQDN (1) of the server (2) that adds the header field.

3.13.2 Timers

None.

3.13.3 Initialization

None.

3.13.4 Higher-Layer Triggered Events

If the user agent supports the safe call transfer procedure described in this section, it MUST advertise this by placing the **ms-safe-transfer** option tag in the **Supported** header of both the INVITE request and the **200 OK** response to the INVITE request.

3.13.5 Message Processing Events and Sequencing Rules

When the user agent receives a REFER request in the INVITE dialog in which it previously advertised support for safe call transfer, as described in section 3.13.4, the user agent MUST examine the **Refer-To** header field of the REFER request. If the **Ms-Sensitivity** header field is present in the **headers** parameter of the URI in the **Refer-To** header field, the user agent MUST extract the **Ms-Sensitivity** header field and its value and add it to the INVITE request that it generates as the result of processing the REFER request.

3.13.6 Timer Events

None.

3.13.7 Other Local Events

None.

3.14 Extensions for ICE SDP Interworking and Multipart MIME Support

This section follows the product behavior described in endnote <68>.

User agents use multi-part MIME to convey multiple SDP parts and call context data in an INVITE request during session initialization. This document describes a method of using multi-part MIME to enable interoperability with SIP elements for which it cannot be determined in advance whether they support [IETFDRAFT-ICENAT-06] or [IETFDRAFT-ICENAT-19] or both.

3.14.1 Abstract Data Model

None.

3.14.2 Timers

None.

3.14.3 Initialization

None.

3.14.4 Higher-Layer Triggered Events

3.14.4.1 Outgoing INVITE

This section follows the product behavior described in endnote <69>.

When a user agent initiates a SIP dialog using an INVITE containing SDP, as defined in [MS-SDPEXT], it MUST use one of the following MIME structures to construct the INVITE request body.

```
3-level deep multipart
   L1: Multipart/mixed
      L2:Multipart/alternative
         L3:SDP ICEv6  (with ms-proxy-2007fallback parameter)
            L3:SDP ICEv19
      L2:Call context
If there is no call context, the following structure is used.
2-level deep multipart
   L1:Multipart/alternative
      L2:SDP ICEv6  (with ms-proxy-2007fallback parameter)
      L2:SDP ICEv19
```

SDP ICEv6 and SDP ICEv19 are specified in [IETFDRAFT-ICENAT-06] and [IETFDRAFT-ICENAT-19] respectively. Call context is described in this section.

L1 refers to the first level in the SIP message body, L2 refers to the second level, and L3 refers to the third level.

The **ms-proxy-2007fallback** parameter in the **Content-Disposition** header field is used as a hint to the proxy server (2) to retry the INVITE with only a single body part when a 415 response is received indicating that the remote user agent does not accept multi-part. The syntax of the **ms-proxy-2007fallback** parameter is described in section 2.2.13, and the applicable proxy server (2) processing of the 415 response is described in section 3.8.5.5.

For 2-level deep multi-part, the SDP MUST be ICEv6, ICEv19 or it does not contain any **Interactive Connectivity Establishment (ICE)**.

If ICEv6 SDP is carried in the multi-part MIME, it MUST be placed in the last part.

3.14.5 Message Processing Events and Sequencing Rules

3.14.5.1 Processing INVITE

When an incoming INVITE is received that contains multi-part MIME structures described in section 3.14.4.1, the user agent MUST pick SDP ICEv19 as the offer if the UAS supports [IETFDRAFT-ICENAT-19], as specified in [MS-SDPEXT].

Alternatively, if the UAS does not support [IETFDRAFT-ICENAT-19], as specified in [MS-SDPEXT], but supports [IETFDRAFT-ICENAT-06], as specified in [MS-SDPEXT], the UAS MUST pick SDP ICEv6 as the offer.<70>

3.14.5.2 Processing 415Response

When an INVITE with the body described in section 3.14.4.1 is rejected with a 415 response, the user agent SHOULD retry the INVITE without multi-part MIME. The body SHOULD contain only SDP ICEv6 without the **ms-proxy-2007fallback** parameter in the **Content-Disposition** header field.

3.14.6 Timer Events

None.

3.14.7 Other Local Events

None.

3.15 Extensions for Agent Anonymity

As specified in section 2.2.20 and section 2.2.21, this protocol defines the **Ms-Call-Info** and **P-Agent-On-Behalf-Of** header fields. The following sections specify the headers and the message processing events for these header fields when anonymization is performed.<71>

3.15.1 Abstract Data Model

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

3.15.1.1 Ms-Call-Info Header

The **Ms-Call-Info** header conveys information about calls. The server (2) endpoint (5) SHOULD set the value of the **Ms-Call-Info** header to "rgs.anonymization". Client endpoints (5) SHOULD ignore any other value.

A server (2) endpoint (5) SHOULD add the **Ms-Call-Info** header to outgoing SIP INVITE and SIP responses to communicate the fact that the call is anonymized. The server (2) endpoint (5) SHOULD provide anonymity. For example, this can be achieved by using a signaling back-to-back agent.

3.15.1.2 P-Agent-On-Behalf-Of Header

When a client endpoint (5) makes a call on behalf of an identity, it MUST use the **P-Agent-On-Behalf-Of** header.

The server (2) endpoint (5) SHOULD validate that the user has the appropriate permission.

3.15.2 Timers

None.

3.15.3 Initialization

None.

3.15.4 Higher-Layer Triggered Events

None.

3.15.5 Message Processing Events and Sequencing Rules

3.15.5.1 Server Behavior

The server (2) endpoint (5) SHOULD send an **Ms-Call-Info** header set to "rgs.anonymization" if it provides anonymity, such as through a back-to-back agent.

Responses to new dialogs established by a user endpoint (5) SHOULD contain an **Ms-Call-Info** header set to "rgs.anonymization" if the server (2) endpoint (5) provides anonymity, such as through a back-to-back agent.

If the server (2) endpoint (5) receives an INVITE with a **P-Agent-On-Behalf-Of** header, it SHOULD validate that the requestor, which is identified by the **P-Asserted-Identity** header, has permission to make on-behalf-of requests. If the **P-Asserted-Identity** header is not present or the requestor does not have the required permission, the request SHOULD be declined with a 403 response.

If the request is valid, the server (2) endpoint (5) SHOULD proceed with the establishment of the call and, if the call is made anonymously, SHOULD add an **Ms-Call-Info** header set to "rgs.anonymization" in its response to the client endpoint (5).

3.15.6 Timer Events

None.

3.15.7 Other Local Events

None.

3.16 E911 Message Processing

This section describes the processing of the E911 INVITE<72>, as defined in section 2.2.22.

3.16.1 Abstract Data Model

None.

3.16.2 Timers

None.

3.16.3 Initialization

None.

3.16.4 Higher-Layer Triggered Events

None.

3.16.5 Message Processing Events and Sequencing Rules

Except as specified in the following section, the rules for message processing are as specified in [RFC3261].

3.16.5.1 Client Behavior

The client retrieves the **locationPolicy** in-band provisioning group, as specified in [MS-SIPREGE] section 2.2.2.5.7. The location policy indicates whether Enhanced Emergency Services are enabled for the endpoint (5), the **Emergency Dial String**, **Emergency Dial Mask**, **NotificationUri**, **ConferenceUri**, **ConferenceMode**, and **LocationPolicyTagID**. The client obtains its location by either making a request to the location information service, as specified in [MS-E911WS], or by capturing the location based on user input. The client composes the INVITE specified in 2.2.22. The client publishes a time-bound routing category instance of the preamble containing the **e911active** flag, as specified in section 3.8.5.1.2, to disable all call forwarding rules, as specified in [MS-SIPREGE]. The client sends the previously composed E911 INVITE to the server (2).

3.16.5.2 Server Behavior

The server (2) identifies an emergency call when it detects the **Priority: emergency** header in the INVITE. The server (2) retrieves the location policy based on the **LocationPolicyTagID** sent within the **PIDF-LO** embedded as a MIME part inside the message body of the INVITE. The server (2) ignores the **geolocation** header and picks the last MIME part that has a **PIDF-LO** embedded in it. Upon receiving the emergency call, in addition to routing the call to E911 Service providers or **public switched telephone network (PSTN)**, the server (2) MUST send an IM message on behalf of the client endpoint (5) making the E911 call to each target in the **NotificationUri** specified in the location policy. The IM INVITE request MUST be constructed as follows:

1. The request MUST contain a **Priority** header with the value "emergency".

2. The request MUST contain a **Call-Info** header with the SIP URI of the user making the emergency call. The **Call-Info** header MUST have a **purpose** parameter with the value "ms-emergency-notification". The ABNF for the **Call-Info** header is defined in [RFC3261].

3. The body of the message MUST be plain text containing all the descendants of the **civicAddress** and **method** elements in the **PIDF-LO** as name-value pairs. The **civicAddress** and **method** element schema are defined in [RFC4119].

3.16.6 Timer Events

None.

3.16.7 Other Local Events

None.

4 Protocol Examples

4.1 EPID Mechanism

The following REGISTER request demonstrates use of the **epid** parameter in the **From** header field.

```
REGISTER sip:contoso.com SIP/2.0From: <sip:alice@contoso.com>;tag=33975904fc;epid=01010101
To: <sip:alice@contoso.com>
Call-ID: 21c7d6e384c249afac26e3f3016140a6
CSeq: 88 REGISTER
```

Note that other SIP headers in the SIP request are not included.

4.2 SIP.INSTANCE Mechanism Example

This example first shows the generation of the **+sip.instance** parameter value for a user agent that uses both **epid** and **+sip.instance** parameters to identify its endpoint (5), as described in section 3.2.3.1.

Given an **epid** parameter value of 01010101, it is first converted to a canonical sequence of octets:

```
0x30 0x31 0x30 0x31 0x30 0x31 0x30 0x31
```

Next, the hash of the name-space identifier concatenated with the canonical representation of the **epid** value is computed:

```
sha1 (0x03 0xfb 0xac 0xfc 0x73 0x8a 0xef 0x46 0x91 0xb1 0xe5 0xeb 0xee 0xab 0xa4 0xfe 0x30
0x31 0x30 0x31 0x30 0x31 0x30 0x31) = 0xA8 0x82 0x16 0x4B 0x68 0xF9 0x01 0xE7 0x03 0xFC 0x7C
0x67 0x41 0xDC 0x66 0x97 0xB8 0xA1 0xA9 0x3E
```

Finally, the previous hash is used to obtain the following UUID:

```
4b1682a8-f968-5701-83fc-7c6741dc6697
```

The following REGISTER request demonstrates the use of the **+sip.instance** parameter in the **Contact** header field and the **epid** parameter in the **From** header field.

```
REGISTER sip:contoso.com SIP/2.0
From: <sip:alice@contoso.com>;tag=33975904fc;epid=01010101
To: <sip:alice@contoso.com>
Call-ID: 21c7d6e384c249afac26e3f3016140a6
CSeq: 88 REGISTER
Contact: <sip:192.0.2.1:27221; transport=tls; msopaque=29c344caf9>; methods="INVITE, MESSAGE,
 INFO, OPTIONS, BYE, CANCEL, NOTIFY, ACK, REFER, BENOTIFY"; proxy=replace; +sip.instance="<ur
n:uuid:4b1682a8-f968-5701-83fc-7c6741dc6697>"
```

Note that other SIP headers in the SIP request are not included.

4.3 GRUU Mechanism

The following examples demonstrate various GRUU syntax:

A **GRUU** for the user agent that follows the registration procedure defined in [MS-SIPREGE] is as follows:

```
sip:alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA
```

A **GRUU** for an application that implements the voice mail service for the user is as follows:

```
sip:alice@contoso.com;gruu;opaque=app:voicemail
```

GRUUs for multimedia conference endpoints (5) are as follows:

```
sip:alice@contoso.com;gruu;opaque=app:conf:focus:id:36022956C3FC3243B8121CD611363ED0
sip:alice@contoso.com;gruu;opaque=app:conf:chat:id:36022956C3FC3243B8121CD611363ED0
sip:alice@contoso.com;gruu;opaque=app:conf:audiovideo:id:36022956C3FC3243B8121CD611363ED0
```

GRUUs for servers (2) are as follows:

```
sip:homeserver.contoso.com@contoso.com;gruu;opaque=srvr:HomeServer:dL8cwxBrTuG8eC4-Q_GNGAAA
sip:mediationserver.contoso.com@contoso.com;gruu;opaque=srvr:MediationServer:_tRfGncbQyun3v75
Q1qr9QAA
sip:mrasserver.contoso.com@contoso.com;gruu;opaque=srvr:MRAS:OKPDbAVxIEKtPh2g624vPAAA
sip:qosmserver.contoso.com@contoso.com;gruu;opaque=srvr:QoSM:WftfTuTVQCSAB0ZJi-j7qAAA
```

4.4 Firewall and Network Address Translation Traversal Aid Extensions

The following example demonstrates how the original REGISTER request was modified by the SIP proxy to preserve transport layer information necessary for NAT traversal.

The original REGISTER request is as follows:

```
REGISTER sip:contoso.com SIP/2.0
From: <sip:alice@contoso.com>;tag=33975904fc;epid=01010101
To: <sip:alice@contoso.com>
Call-ID: 21c7d6e384c249afac26e3f3016140a6
CSeq: 88 REGISTER
Via: SIP/2.0/TLS 192.0.2.1:27221
Contact: <sip:192.0.2.1:27221; transport=tls; msopaque=29c344caf9>; methods="INVITE, MESSAGE,
 INFO, OPTIONS, BYE, CANCEL, NOTIFY, ACK, REFER, BENOTIFY"; proxy=replace; +sip.instance="<ur
n:uuid:4b1682a8-f968-5701-83fc-7c6741dc6697>"
```

The REGISTER request after proxy processing is as follows:

```
REGISTER sip:contoso.com SIP/2.0
From: <sip:alice@contoso.com>;tag=33975904fc;epid=01010101
To: <sip:alice@contoso.com>
Call-ID: 21c7d6e384c249afac26e3f3016140a6
CSeq: 88 REGISTER
Via: SIP/2.0/TLS 192.0.2.1:27221; received=192.168.0.2; ms-received-port=1201; ms-received-
cid=3540900
Contact: <sip:192.168.0.2:1201; transport=tls; msopaque=29c344caf9; msreceivedcid=3540900>; m
ethods="INVITE, MESSAGE, INFO, OPTIONS, BYE, CANCEL, NOTIFY, ACK, REFER, BENOTIFY"; +sip.inst
ance="<urn:uuid:4b1682a8-f968-5701-83fc-7c6741dc6697>"
```

4.5 Reliable and Consistent Message Routing Within Redundant Server Network

The following example demonstrates SIP proxies placing various pieces of information into the **Record-Route** header fields of the dialog creating a 200 OK response message to a SUBSCRIBE request.

```
SIP/2.0 200 OK
FROM: <sip:alice@contoso.com>;tag=2187d9f392;epid=01010101
TO: <sip:bob@contoso.com>;tag=3l3qz7tx
CSEQ: 3 SUBSCRIBE
CALL-ID: f0ec9c595c1f412ca6b71318beb599bb
RECORDROUTE: <sip:server1.contoso.com:5061;transport=tls;lr;ms-key-
info=mACAAODZIzT_XXbu1V_IAQECAAADZgAAAKQAANMFUpbsXZoVmYcoLP8PT9anIkOw7BnvcFRRkZewoiMYj3B61Yac
QGTK4TMsKnJXCM86liVZHosw8jUyFf2OXMyOLLv3ZVw477ajvdErKm0E5OQybBg8o6e3g1wK9rua4xUHwyZ1T6_CkS6TQ
vpebxXJG5Y8dA40VIzMIllIjAHfRSo9XMZW1lyJnpHoa53vuD1BV1QccxH9ht5dw3sKqKAgsyBT4Bmm3abFJ6nKhZpNly
bt6EkVqBD7ArG5dyNPrUlcT8VLOPINVSGwvviWBygEVRfIGauMqIbMooXLq6PMYUAg6TIYfEIdugqRnIYgu_hnihBK6wK
jV2w;msroutesig=ga3IN7MltlsglDvxIE_bYt5lVbZ3E>
RECORDROUTE: <sip:server2.contoso.com:5061;transport=tls;msrolersfrom;lr;msroutesig=ec1Fe_32f
glb4iILWFJb5iKqeNeps7y6vY9zXAAA>
CONTACT: <sip:alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA>
```

4.6 Dialog State Recovery

This section follows the product behavior described in endnote <73>.

The following example shows messages exchanged between the user agent and the proxy server (2) when the proxy detects dialog state loss and communicates this to the user agent, which subsequently recovers the dialog.

The user agent sends a mid-dialog request with the route set from the current dialog state.

```
MESSAGE sip:Alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA SIP/2.0
Route: <sip:server.contoso.com:5061;transport=tls;opaque=state:F:T:Ci.D1100:Ti.dyHFp3e3J0mXFh
CDvmsQ7QAA;lr;msroutesig=aag0AbAT3mK4Ga8lsHSyTeZnAETjcRJpFx8YnUbQAA>
From: sip:Bob@contoso.com;epid=02020202;tag=02020202
To: sip:Alice@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bK94bd
Cseq: 3 MESSAGE
Supported: Ms-Dialog-Route-Set-Update
Content-Length: 27

Alice, are you still there?
```

The proxy detects that the references to the state information stored in the route set are not valid and that the user agent supports the dialog state recovery procedure as indicated by the **Ms-Dialog-Route-Set-Update** option tag in the **Supported** header field. The proxy responds with a 430 Flow Failed response, requesting the user agent to update the dialog state information.

```
SIP/2.0 430 Flow Failed
From: sip:Bob@contoso.com;epid=02020202;tag=02020202
To: sip:Alice@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bK94bd;msreceivedcid=3540900
Cseq: 3 MESSAGE
P-Dialog-Recovery-Action: dialog-route-set-update
```

```
Content-Length: 0
```

The user agent sends the correct target refresh request without the route set to recover the dialog state.

```
INVITE sip:Alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA SIP/2.0
From: sip:Bob@contoso.com;epid=02020202;tag=02020202
To: sip:Alice@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bKa8d4
Cseq: 4 INVITE
Supported: Ms-Dialog-Route-Set-Update
Contact: <sip:Bob@contoso.com;gruu;opaque=user:epid:uVUjrngkI1wHVm3r2esBAAA>
Content-Length: 0
```

The user agent receives the 200 OK response and updates its dialog state with the new route set.

```
SIP/2.0 200 OK
RecordRoute: <sip:server.contoso:5061;transport=tls;opaque=state:F:T:Ci.D1200:Ti.dyHFp3e3J0mX
FhCDvmsQ7QAA;lr;msroutesig=aalzpOt84oODZx4KmWgmgJLf_WGfEsKwh8YnUbQAA>
From: sip:Bob@contoso.com;epid=02020202;tag=02020202
To: sip:Alice@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bKa8d4;msreceivedcid=3540900
Cseq: 4 INVITE
Contact: <sip:Alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA>
Content-Length: 0
```

The user agent then resends the request with the updated route set.

```
MESSAGE sip:Alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA SIP/2.0
Route: <sip:server.contoso:5061;transport=tls;opaque=state:F:T:Ci.D1200:Ti.dyHFp3e3J0mXFhCDvm
sQ7QAA;lr;msroutesig=aalzpOt84oODZx4KmWgmgJLf_WGfEsKwh8YnUbQAA>
From: sip:Bob@contoso.com;epid=02020202;tag=02020202
To: sip:Alice@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bK97b2
Cseq: 5 MESSAGE
Supported: Ms-Dialog-Route-Set-Update
Content-Length: 27

Alice, are you still there?
```

The request gets through and the user agent receives a successful response.

```
SIP/2.0 200 OK
From: sip:Bob@contoso.com;epid=02020202;tag=02020202
To: sip:Alice@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bK97b2;msreceivedcid=3540900
Cseq: 5 MESSAGE
Content-Length: 0
```

4.7 Routing Preamble

4.7.1 Blocking Preamble

The following is an example of a preamble that blocks the call.

```
<?xml version="1.0" encoding="utf-8"?>
<routing xmlns="http://schemas.microsoft.com/02/2006/sip/routing"
        name="rtcdefault" version="1" >
  <preamble>
<flags name="clientflags" value="block"/>
  </preamble>
</routing>
```

In the following example, because the **clientflags** contains "block", the call is blocked.

4.7.2 Simultaneous Ring

```
<?xml version="1.0" encoding="utf-8"?>
<routing xmlns="http://schemas.microsoft.com/02/2006/sip/routing"
        name="rtcdefault" version="1" >
  <preamble >
    <list name="forwardto">
      <target uri="sip:+14255550199@contoso.com;user=phone"/>
    </list>
    <list name="simultaneous_ring" >
      <target uri="sip:+14255550100@contoso.com;user=phone"/>
    </list>
    <flags name="clientflags" value="work_hours simultaneous_ring enablecf"/>
    <wait name="total" seconds="18"/>
  </preamble>
</routing>
```

In the previous example, the call is forked to all the registered endpoints (5) of the user and, because the **simultaneous_ring** flag is set, the call is also forked to the simultaneous ring device "sip:+14255550100@contoso.com;user=phone". If no success response is received within 18 seconds, which is the wait time specified in the **wait** element named **total**, all forks are cancelled. Because the **enablecf** flag is set, the call is then forked to the forwarding destination indicated in the **forwardto** list, which is "sip:+14255550199@contoso.com;user=phone".

4.7.3 Call Forward

```
<?xml version="1.0" encoding="utf-8"?>
<routing xmlns="http://schemas.microsoft.com/02/2006/sip/routing"
        name="rtcdefault" version="1" >
  <preamble >
    <list name="forwardto">
      <target uri="sip:+14255550199@contoso.com;user=phone"/>
    </list>
    <list name="simultaneous_ring" >
      <target uri="sip:+14255550100@contoso.com;user=phone"/>
    </list>
    <flags name="clientflags" value="work_hours forward_immediate simultaneous_ring
enablecf"/>
    <wait name="total" seconds="18"/>
```

```
  </preamble>
</routing>
```

In the previous example, the **forward_immediate** flag indicates that the call is forwarded immediately. Because the **enablecf** flag is also present, the call is forwarded to the destination in the **forwardto** list. If the **enablecf** flag is not present, the call is forwarded to the user's voice mail. In either case, the registered endpoints (5) and the simultaneous ring device are not rung.

4.7.4 Team Ring

This section follows the product behavior described in endnote <74>.

```
<?xml version="1.0" encoding="utf-8"?>
<routing xmlns="http://schemas.microsoft.com/02/2006/sip/routing"
        name="rtcdefault" version="2"
minSupportedClientVersion="2.0.0.0" >
  <preamble >
    <list name="team">
      <target uri="sip:Alice@contoso.com "/>
    <target uri="sip:Bob@contoso.com "/>
    </list>
    <flags name="clientflags" value="team_ring"/>
<wait name="user" seconds="10"/>
<wait name="team2" seconds="10"/>
  </preamble>
</routing>
```

In this example, the **team_ring** flag indicates that team ringing is enabled. The call is forked to all registered endpoints (5). If no success response is received within **user** seconds, which is 10 seconds in this example, the call is routed to the targets specified in the **team** list, Alice@contoso.com and Bob@contoso.com. Note that the registered endpoints (5) are not cancelled at this time. If no success response is received within 10 additional seconds, which is the **team2** wait time, all existing forks are cancelled and the call is forwarded to voice mail if the user is enabled for voice mail.

4.8 History-Info Example

This section follows the product behavior described in endnote <75>.

The following example shows the **History-Info** header field inserted by the proxy in the INVITE request forwarded to the registered endpoint (5).

```
INVITE sip:192.0.2.1:51152;transport=tls;msopaque=bab87d7e6e;msreceived-cid=244100
SIP/2.0RecordRoute: <sip:server.contoso.com:5061;transport=tls;opaque=state:F:Ci.R2>;msrrsig=
djvCtpOBl7EzJlJIPA8FZ2TtCdfcZHZduS3M4K_QAA;tag=C2FBFDDF86D85988E2FE9C475D8B20D0Via: SIP/2.0/T
LS 192.168.0.2:5061;branch=z9hG4bK.A1ABD;branched=TRUE;msinternalinfo="bvL4ijJzvRAsUh9KHAufCF
_yfKiWpHZduSTBXqAAAA"Via: SIP/2.0/TLS 192.168.0.3:1199;branch=z9hG4bK94bd;msreceivedcid=A552C
00Authentication-Info: NTLM rspauth="01000000ECFE1CAD61AAC15164000000", srand="AC62DEB8",
snum="504", opaque="DC8F829A", qop="auth", targetname="server.contoso.com", realm="SIP
Communications Service"Max-Forwards: 68Content-Length: 0From:
<sip:Alice@contoso.com>;epid=01010101To: <sip:Bob@contoso.com>;epid=02020202CSeq: 39513
INVITECall-ID: 772937b8-0e12-4639-8c79-
9d2ac32f2a56Contact: <sip:alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA>Su
pported: gruu-10History-Info: <sip:Bob@contoso.com>;index=1
```

4.9 Extension for Federation and Public IM Connectivity

The following examples show the extension header field **ms-edge-proxy-message-trust** used for federation (2) and public IM connectivity. The format for this header field is specified in section 2.2.14.

In this example, the **ms-edge-proxy-message-trust** header field indicates that the SIP NOTIFY message was received from a federated partner:

```
NOTIFY sip:192.0.2.1:18168; transport=tls; msopaque=7eacdda82d; msreceivedcid=7C9B00; grid SI
P/2.0
msedgeproxymessagetrust: mssourcetype=AutoFederation; msepfqdn=edgeserver.contoso.com; mssour
ceverifieduser=verified; mssourcenetwork=federation
```

Note that other SIP headers in the SIP request are not included.

In this example, the **ms-edge-proxy-message-trust** header field indicates that the SIP NOTIFY message was received from a public IM provider:

```
NOTIFY sip:192.0.2.1:18168; transport=tls; msopaque=7eacdda82d; msreceivedcid=7C9B00; grid SI
P/2.0
ms-edge-proxy-message-trust: ms-source-type=AuthorizedServer;ms-ep-
fqdn=edgeserver.contoso.com;ms-source-verified-user=verified;ms-source-network=publiccloud
```

Note that other SIP headers in the SIP request are not included.

In this example, the **ms-edge-proxy-message-trust** header field indicates that the SIP response was generated by a server (2) on the enterprise network edge because it could not route the outbound message:

```
SIP/2.0 504 Server time-out
msedgeproxymessagetrust: mssourcetype=EdgeProxyGenerated; msepfqdn=edgeserver.contoso.com; ms
sourceverifieduser=verified; mssourcenetwork=federation
```

Note that other SIP headers in the SIP response are not included.

4.10 Extension for Remote Users

The following examples show the extension header field **ms-user-logon-data**. The format for this header field is specified in section 2.2.15.

The following example shows a response to a REGISTER request. The **ms-user-logon-data** header field indicates that the user is a remote user.

```
SIP/2.0 200 OK
From: <sip:alice@contoso.com>;tag=1b3884236d;epid=e06accb078
To: <sip:alice@contoso.com>;tag=D4EF81E564DD858A326CC721EF4A8FAF
Call-ID: 5899a88068934f8385a0b0b5e03be045
CSeq: 3 REGISTER
ms-user-logon-data: RemoteUser
AuthenticationInfo: NTLM rspauth="01000000000000046DD35D06323180F", srand="64306136", snum="
1", opaque="0A79BAD2", qop="auth", targetname="ocsserver.contoso.com", realm="SIP Communicati
ons Service"
RecordRoute: <sip:server1.contoso.com:5061;transport=tls;lr;msreceived-cid=3AFDE300>
```

```
Contact: <sip:192.0.2.4:2904;transport=tls;msopaque=2cd64e3000;msreceivedcid=1D8AF00>;expires
=2905;+sip.instance="<urn:uuid:75ab1008bcc45544924daa177c824291>";gruu="sip:alice@contoso.com
;opaque=user:epid:CBCrdcS8RFWSTaoXfIJCkQAA;gruu"
```

4.11 Extension for Call Context

This section follows the product behavior described in endnote <76>.

The following examples show the extension content type **application/ms-conversation-context+xml**. The format for this content type is specified in section 2.2.19.

The following example shows an INVITE request containing the **application/ms-conversation-context+xml** content type in the message body of the request.

```
INVITE sip:192.0.2.3:59682;transport=tls;ms-opaque=f297889669;ms-received-cid=4EA600
SIP/2.0From: <sip:alice@contoso.com>;epid=42933B3A88;tag=f962b589a8To:
<sip:marco@contoso.com>;epid=7913c4c11dContent-Length: ...
Content-Type: multipart/mixed;boundary=0VUf5fZQGOkBjYIfaZ2yOZCi5OdMrt2A

--0VUf5fZQGOkBjYIfaZ2yOZCi5OdMrt2A
CONTENT-TYPE: multipart/alternative; boundary=4FqyUUSfl7GyNwhB0PABKoF6PTFb6Ovl
--4FqyUUSfl7GyNwhB0PABKoF6PTFb6OvlContent-Type: ...Content-ID: e22b7561-b5df-4b86-89c0-
b20702e2de83Content-Disposition: ...
...

--4FqyUUSfl7GyNwhB0PABKoF6PTFb6OvlContent-Type: ...Content-ID: 8a09b2b6-afdc-47d3-bc33-
5fda39d66463
...

--4FqyUUSfl7GyNwhB0PABKoF6PTFb6Ovl--
--0VUf5fZQGOkBjYIfaZ2yOZCi5OdMrt2AContent-ID: 5c44530a-8955-4514-8527-eaddf24b30aeContent-
Type: application/ms-conversation-context+xmlContent-Disposition: render;handling=optional
<cc:XmlConvContext xmlns:cc="http://schemas.microsoft.com/2008/03/sip/conversationContext">
<cc:id>0734aae0-a714-45d9-87bc-20ed9d432b80</cc:id>
<cc:from><cc:uri>sip:alice@contoso.com</cc:uri></cc:from>
<cc:to><cc:uri>sip:marco@contoso.com</cc:uri></cc:to>
<cc:participants>
<cc:participant>
<cc:uri>sip:alice@contoso.com</cc:uri>
<cc:displayName>Alice</cc:displayName>
</cc:participant>
<cc:participant>
<cc:uri>sip:bob@contoso.com</cc:uri>
</cc:participant>
</cc:participants>
<cc:date>2008-09-03T21:34:55.831063Z</cc:date>
<cc:mode>audio</cc:mode>
<cc:conversationId>a4f266f1a6914acb99cddef15659e38c</cc:conversationId>
<cc:dataFormat>text/plain</cc:dataFormat>
<cc:contextData>Waiting time: 00:00:18
Bob is calling, it’s his birthday today.
</cc:contextData></cc:XmlConvContext>--0VUf5fZQGOkBjYIfaZ2yOZCi5OdMrt2A--
```

4.12 Multipart MIME

4.12.1 Two- level Multipart MIME Example

All content in section 4.12 follows the product behavior described in endnote <77>.

The following example shows a two-level multi-part MIME, as described in section 3.14.

```
Content-Type: multipart/alternative;boundary="----=_NextPart_000_0059_01C91A7C.B83AD4E0"
Content-Length: 4014
------=_NextPart_000_0059_01C91A7C.B83AD4E0
Content-Type: application/sdp
Content-Transfer-Encoding: 7bit
Content-Disposition: session; handling=optional; ms-proxy-2007fallback
v=0
o=- 0 0 IN IP4 10.80.20.10
s=session
c=IN IP4 10.80.20.10
b=CT:35980
t=0 0
m=audio 50019 RTP/AVP 114 111 112 115 116 4 8 0 97 13 118 101
k=base64:9Izc9LPyPH3s1sl7XB0umY6R1B8H93Ru2knWs9pLcqxIlsPKgGq9iLaWcNNy
a=candidate:lLh4oR2NlwKLCbqk7rt7UJdJqHFEn9QeGNyYH6y8lGo 1 gKxsnl/9hhaK8j1Bc2tp4g UDP 0.830
10.80.20.10 50019
a=candidate:lLh4oR2NlwKLCbqk7rt7UJdJqHFEn9QeGNyYH6y8lGo 2 gKxsnl/9hhaK8j1Bc2tp4g UDP 0.830
10.80.20.10 50014
a=candidate:fI9holTcjzGzlUSH+fI+8hpZi/D+Y0bREpI35R6xbOY 1 V4xXN538Z4zIurS6nPYZiw TCP 0.190
131.107.1.36 52668
a=candidate:fI9holTcjzGzlUSH+fI+8hpZi/D+Y0bREpI35R6xbOY 2 V4xXN538Z4zIurS6nPYZiw TCP 0.190
131.107.1.36 52668
a=candidate:8/ugcPvoRu7X7870q7LcuZOAz8H1w1UZ1iz0JcyBfNI 1 Hv+ChtZX/SeNamyISSwstQ UDP 0.490
131.107.1.36 58325
a=candidate:8/ugcPvoRu7X7870q7LcuZOAz8H1w1UZ1iz0JcyBfNI 2 Hv+ChtZX/SeNamyISSwstQ UDP 0.490
131.107.1.36 50664
a=candidate:HSUcTjchkwG7k7cMX0tALAz4bty/uV/KvfSkV7Cc73I 1 nbUV3FDCmrixfcyP4PwwVQ TCP 0.250
10.80.20.10 50019
a=candidate:HSUcTjchkwG7k7cMX0tALAz4bty/uV/KvfSkV7Cc73I 2 nbUV3FDCmrixfcyP4PwwVQ TCP 0.250
10.80.20.10 50019
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:1KjtxsXPzJi3Llf7jhKlGv9YSEdr0sPzwx9p7wQ2|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:xgZxo13cfXDz1Vflqw2x+EB5cCdBh2Q0gsZfmE8D|2^31|1:1
a=maxptime:200
a=rtcp:50014
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:112 G7221/16000
a=fmtp:112 bitrate=24000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:116 AAL2-G726-32/8000
a=rtpmap:4 G723/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:97 RED/8000
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=rtpmap:101 telephone-event/8000
```

```
a=fmtp:101 0-16
a=encryption:required
------=_NextPart_000_0059_01C91A7C.B83AD4E0
Content-Type: application/sdp
Content-Transfer-Encoding: 7bit
Content-Disposition: session; handling=optional
v=0
o=- 0 0 IN IP4 10.80.20.10
s=session
c=IN IP4 10.80.20.10
b=CT:35980
t=0 0
m=audio 50023 RTP/AVP 114 111 112 115 116 4 8 0 97 13 118 101
k=base64:9Izc9LPyPH3s1sl7XB0umY6R1B8H93Ru2knWs9pLcqxIlsPKgGq9iLaWcNNy
a=ice-ufrag:wdB31g
a=ice-pwd:yAbXGTFPoM+Kt2+fvhUUdKkclwSChFQj
a=candidate:1 1 UDP 2130706431 10.80.20.10 50023 typ host
a=candidate:1 2 UDP 2130705918 10.80.20.10 50016 typ host
a=candidate:2 1 TCP-PASS 6556159 131.107.1.36 50370 typ relay raddr 131.107.1.36 rport 50370
a=candidate:2 2 TCP-PASS 6556158 131.107.1.36 50370 typ relay raddr 131.107.1.36 rport 50370
a=candidate:3 1 UDP 16648703 131.107.1.36 56997 typ relay raddr 131.107.1.36 rport 56997
a=candidate:3 2 UDP 16648702 131.107.1.36 56644 typ relay raddr 131.107.1.36 rport 56644
a=candidate:4 1 TCP-ACT 7076863 131.107.1.36 50370 typ relay raddr 131.107.1.36 rport 50370
a=candidate:4 2 TCP-ACT 7076350 131.107.1.36 50370 typ relay raddr 131.107.1.36 rport 50370
a=candidate:5 1 TCP-ACT 1684797951 10.80.20.10 50018 typ srflx raddr 10.80.20.10 rport 50018
a=candidate:5 2 TCP-ACT 1684797438 10.80.20.10 50018 typ srflx raddr 10.80.20.10 rport 50018
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:1KjtxsXPzJi3Llf7jhKlGv9YSEdr0sPzwx9p7wQ2|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:xgZxo13cfXDz1Vflqw2x+EB5cCdBh2Q0gsZfmE8D|2^31|1:1
a=maxptime:200
a=rtcp:50016
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:112 G7221/16000
a=fmtp:112 bitrate=24000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:116 AAL2-G726-32/8000
a=rtpmap:4 G723/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:97 RED/8000
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=encryption:required
------=_NextPart_000_0059_01C91A7C.B83AD4E0--
```

4.12.2 Three- level Multipart MIME Example

The following example shows a three-level multi-part MIME, as described in section 3.14.

```
Content-Type: multipart/mixed; boundary=HkS4RpzThV2XRK9lcuE3NJUcskesnr9w
Content-Type: multipart/alternative; boundary=sYRNyS9rx1iUksZ4fH8roFi2MbQU6dbo
```

```
--sYRNyS9rx1iUksZ4fH8roFi2MbQU6dbo
Content-Type: application/sdp
Content-ID: ccbe8227-c734-4d4a-b1ce-0ed219097ff4
Content-Disposition: session;handling=optional;ms-proxy-2007fallback
v=0
o=- 0 0 IN IP4 172.29.105.158
s=session
c=IN IP4 172.29.105.158
b=CT:1000
t=0 0
m=audio 23160 RTP/AVP 8 0 4 116 3 115 112 111 114 13 118 97 101
c=IN IP4 172.29.105.158
a=rtcp:29398
a=candidate:mDUVW7BtzxI1duehZtgEB9+HmyHI2DNgAY1V0UrdYIo 1 tKxTKKdnyDIj5nLnGLIXpw UDP 0.900
172.29.105.158 23160
a=candidate:mDUVW7BtzxI1duehZtgEB9+HmyHI2DNgAY1V0UrdYIo 2 tKxTKKdnyDIj5nLnGLIXpw UDP 0.900
172.29.105.158 29398
a=candidate:6pJIvJXR/PECSSKwaR+ygUx9hRd360XbnImL36GTD6M 1 eaPFs6Wp3vVT+WMStx5WDg TCP 0.150
172.29.105.171 51143
a=candidate:6pJIvJXR/PECSSKwaR+ygUx9hRd360XbnImL36GTD6M 2 eaPFs6Wp3vVT+WMStx5WDg TCP 0.150
172.29.105.171 51143
a=candidate:HuZ/qrwBjoj/TpiTR07CLJpJ1JpKVzjHu+EYh5G8uTg 1 ut9XFV7u5hWESZuqESPHLQ UDP 0.450
172.29.105.171 53824
a=candidate:HuZ/qrwBjoj/TpiTR07CLJpJ1JpKVzjHu+EYh5G8uTg 2 ut9XFV7u5hWESZuqESPHLQ UDP 0.450
172.29.105.171 52048
a=candidate:1/UjDo+KnYxwlJvWgELKP93RoXKk+vOKxfjCHpmh9nk 1 73jZjOF9LVx/jQTKT/bySA TCP 0.250
172.29.105.158 3512
a=candidate:1/UjDo+KnYxwlJvWgELKP93RoXKk+vOKxfjCHpmh9nk 2 73jZjOF9LVx/jQTKT/bySA TCP 0.250
172.29.105.158 3512
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:/h4AObPXOlrc7LkgLj03byQ7PVvuzfmwx3NJXn1+|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:OR/d0mnfMTRGa6IFw0JN5CeR6ZwMTWTWoz54IiOM|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:ha8qW6njHa9nEDqV78Iy1aDfDQb3dsXidivURp0+|2^31
a=label:main-audio
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:4 G723/8000
a=rtpmap:116 AAL2-G726-32/8000
a=rtpmap:3 GSM/8000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:112 G7221/16000
a=fmtp:112 bitrate=24000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=rtpmap:97 RED/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
--sYRNyS9rx1iUksZ4fH8roFi2MbQU6dbo
Content-Type: application/sdp
Content-ID: 38fcdc48-dc5e-48a0-9681-532010d92196
v=0
o=- 0 0 IN IP4 172.29.105.158
s=session
c=IN IP4 172.29.105.158
b=CT:1000
```

```
t=0 0
m=audio 25170 RTP/AVP 8 0 4 116 3 115 112 111 114 13 118 97 101
c=IN IP4 172.29.105.158
a=rtcp:14396
a=ice-ufrag:2UclRQ
a=ice-pwd:So72NmoVpGdlUE7zWhKQKsP+zteJmfSc
a=candidate:1 1 UDP 2130706431 172.29.105.158 25170 typ host
a=candidate:1 2 UDP 2130705918 172.29.105.158 14396 typ host
a=candidate:2 1 tcp-pass 6555135 172.29.105.171 56700 typ relay raddr 172.29.105.171 rport
56700
a=candidate:2 2 tcp-pass 6555134 172.29.105.171 56700 typ relay raddr 172.29.105.171 rport
56700
a=candidate:3 1 UDP 16647679 172.29.105.171 53833 typ relay raddr 172.29.105.171 rport 53833
a=candidate:3 2 UDP 16647678 172.29.105.171 57341 typ relay raddr 172.29.105.171 rport 57341
a=candidate:4 1 tcp-act 7076863 172.29.105.171 56700 typ relay raddr 172.29.105.171 rport
56700
a=candidate:4 2 tcp-act 7076350 172.29.105.171 56700 typ relay raddr 172.29.105.171 rport
56700
a=candidate:5 1 tcp-act 1684797951 172.29.105.158 26980 typ srflx raddr 172.29.105.158 rport
26980
a=candidate:5 2 tcp-act 1684797438 172.29.105.158 26980 typ srflx raddr 172.29.105.158 rport
26980
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:/h4AObPXOlrc7LkgLj03byQ7PVvuzfmwx3NJXn1+|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:OR/d0mnfMTRGa6IFw0JN5CeR6ZwMTWTWoz54IiOM|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:ha8qW6njHa9nEDqV78Iy1aDfDQb3dsXidivURp0+|2^31
a=label:main-audio
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:4 G723/8000
a=rtpmap:116 AAL2-G726-32/8000
a=rtpmap:3 GSM/8000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:112 G7221/16000
a=fmtp:112 bitrate=24000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=rtpmap:97 RED/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
--sYRNyS9rx1iUksZ4fH8roFi2MbQU6dbo--
--HkS4RpzThV2XRK9lcuE3NJUcskesnr9w
Content-ID: 2b700e68-70cd-4de9-b8e6-78625ca48b3f
CONTENT-TYPE: application/ms-conversation-context+xml
Content-Disposition: render;handling=optional
<cc:XmlConvContext xmlns:cc="http://schemas.microsoft.com/2008/03/sip/conversationContext">
  <cc:id>fb578ae6-577c-4f9f-8510-d74c29c71e2e</cc:id>
  <cc:from>
    <cc:uri>sip:help_desk@fabrikam.com</cc:uri>
  </cc:from>
  <cc:to>
    <cc:uri>sip:Agent9@fabrikam.com</cc:uri>
  </cc:to>
  <cc:participants>
```

```
    <cc:participant>
      <cc:uri>sip:danp@fabrikam.com</cc:uri>
      <cc:displayName>Dan Park</cc:displayName>
    </cc:participant>
    <cc:participant>
      <cc:uri>sip:help_desk@fabrikam.com</cc:uri>
    </cc:participant>
    <cc:participant>
      <cc:uri>sip:Agent9@fabrikam.com</cc:uri>
    </cc:participant>
  </cc:participants>
  <cc:date>2008-09-11T21:07:33.6378654Z</cc:date>
  <cc:mode>audio</cc:mode>
  <cc:conversationId>61020efc64bb4f2f87f631c99bb65b7e</cc:conversationId>
  <cc:dataFormat>text/plain</cc:dataFormat>
  <cc:contextData>Waiting time: 00:00:05
IVR information:
Question: Press or say one for Benefits press or say two for Human Resources
Answer: 1
</cc:contextData>
</cc:XmlConvContext>
--HkS4RpzThV2XRK9lcuE3NJUcskesnr9w--
```

4.13 Agent Anonymity

This section follows the product behavior described in endnote <78>.

The following example shows the INVITE a server (2) endpoint (5) sends to establish an anonymous call, excluding common required headers and the SDP part.

```
INVITE sip:Alice@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA SIP/2.0
From: sip:Bob@contoso.com;epid=02020202;tag=02020202
To: sip:Alice@contoso.com;
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bKa8d4
Cseq: 4 INVITE
Ms-Call-Info: Rgs.Anonymization
Contact:<sip:server1@contoso.com;gruu;opaque=srvr:HomeServer:VWIdpJWTA1eatgf05sHGswAA>;automa
ta;actor="attendant";text;audio;video;image
```

In this example, the server (2) endpoint (5) is impersonating Bob. The contact remains the server (2) endpoint (5) GRUU.

The following example show the 200 OK response a server (2) endpoint (5) sends to establish an anonymous call initiated by a user endpoint (5), excluding common required headers and the SDP part.

```
SIP/2.0 200 OK
From: sip:Alice@contoso.com;epid=02020202;tag=02020202
To: sip:Helpdesk@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bKa8d4
Cseq: 4 INVITE
Ms-Call-Info: Rgs.Anonymization
```

```
Contact:<sip:server1@contoso.com;gruu;opaque=srvr:HomeServer:VWIdpJWTA1eatgf05sHGswAA>;automa
ta;actor="attendant";text;audio;video;image
```

The following example show the request a client endpoint (5) can send to request a call on behalf of the Helpdesk and the response from the server (2) endpoint (5), using anonymity and excluding common required headers and the SDP part.

```
INVITE sip:Helpdesk@contoso.com;gruu;opaque=user:epid:qIIWS2j5AVeD_HxnQdxmlwAA SIP/2.0
From: sip:Alice@contoso.com;epid=02020202;tag=02020202
To: sip:Bob@contoso.com;
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bKa8d4
Cseq: 4 INVITE
P-Agent-On-Behalf-Of: sip:Helpdesk@contoso.com

SIP/2.0 200 OK
From: sip:Alice@contoso.com;epid=02020202;tag=02020202
To: sip:Bob@contoso.com;epid=01010101;tag=01010101
Call-Id: f0ec9c595c1f412ca6b71318beb599bb
Via: SIP/2.0/TLS 192.0.2.1:27221;branch=z9hG4bKa8d4
Cseq: 4 INVITE
Ms-Call-Info: Rgs.Anonymization
Contact:<sip:server1@contoso.com;gruu;opaque=srvr:HomeServer:VWIdpJWTA1eatgf05sHGswAA>;automa
ta;actor="attendant";text;audio;video;image
```

4.14 E911 INVITE

This section follows the product behavior described in endnote <79>.

The following example shows an E911 INVITE that the client endpoint (5) can send to establish an E911 call. This example excludes common required headers.

```
INVITE sip:911;phone-context=Redmond@192.168.1.12;user=phone SIP/2.0
From: "voip_911_user1"<sip:voip_911_user1@contoso.com>;epid=1D19090AED;tag=d04d65d924
To: <sip:911;phone-context=Redmond@192.168.1.12;user=phone>
CSeq: 8 INVITE
Call-ID: e6828be1-1cdd-4fb0-bdda-cda7faf46df4
VIA: SIP/2.0/TLS 192.168.0.244:57918;branch=z9hG4bK528b7ad7
CONTACT:
<sip:voip_911_user1@contoso.com;opaque=user:epid:R4bCDaUj51a06PUbkraS0QAA;gruu>;text;audio;vi
deo;image
PRIORITY: emergency
CONTENT-TYPE: multipart/mixed; boundary= ------=_NextPart_000_4A6D_01CAB3D6.7519F890
geolocation: <cid:voip_911_user1@contoso.com>;inserted-by="sip:voip_911_user1@contoso .com"
Message-Body:
------=_NextPart_000_4A6D_01CAB3D6.7519F890
Content-Type: application/sdp ; charset=utf-8
v=0
o=- 0 0 IN IP4 Client
s=session
c=IN IP4 Client
t=0 0
m=audio 30684 RTP/AVP 114 111 112 115 116 4 3 8 0 106 97
c=IN IP4 172.29.105.23
a=rtcp:60423
a=label:Audio
```

```
a=rtpmap:3 GSM/8000/1
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=ptime:20

------=_NextPart_000_4A6D_01CAB3D6.7519F890
Content-Type: application/pidf+xml
Content-ID: <voip_911_user1@contoso.com>
<?xml version="1.0" encoding="utf-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
xmlns:gp="urn:ietf:params:xml:ns:pidf:geopriv10"
xmlns:bp="urn:ietf:params:xml:ns:pidf:geopriv10:basicPolicy"
xmlns:ca="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr"
xmlns:ms="urn:schema:Rtc.LIS.msftE911PidfExtn.2008"
entity="sip:voip_911_user1@contoso.com"><tuple id="0"><status><gp:geopriv><gp:location-
info><ca:civicAddress><ca:country>US</ca:country><ca:A1>WA</ca:A1><ca:A3>Redmond</ca:A3><ca:R
D>163rd</ca:RD><ca:STS>Ave</ca:STS><ca:POD>NE</ca:POD><ca:HNO>3910</ca:HNO><ca:LOC>40/4451</c
a:LOC><ca:NAM>Contoso Corporation
</ca:NAM><ca:PC>98052</ca:PC></ca:civicAddress></gp:location-info><gp:usage-
rules><bp:retransmission-allowed>true</bp:retransmission-allowed></gp:usage-
rules></gp:geopriv><ms:msftE911PidfExtn><ms:ConferenceUri>sip:+14255550199@contoso.com;user=p
hone</ms:ConferenceUri><ms:ConferenceMode>twoway</ms:ConferenceMode><LocationPolicyTagID
xmlns="urn:schema:Rtc.Lis.LocationPolicyTagID.2008">user-tagid</LocationPolicyTagID
></ms:msftE911PidfExtn></status><timestamp>1991-09-
22T13:37:31.03</timestamp></tuple></presence>
------=_NextPart_000_4A6D_01CAB3D6.7519F890--
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full Routing Script Preamble Format

Following is the full XML schema for the routing script preamble:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.microsoft.com/02/2006/sip/routing"
xmlns:tns="http://schemas.microsoft.com/02/2006/sip/routing" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <!-- The following type defination are used in the preamble-->
  <xs:complexType name="target-type">
    <xs:annotation>
      <xs:documentation>At least one of uri or application attributes are required to be
present.</xs:documentation>
    </xs:annotation>
    <xs:attribute name="uri" type="xs:string" use="optional" />
    <xs:attribute name="application" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:complexType name="timezone-date-type">
    <xs:attribute name="name" type="xs:string" use="optional" />
    <xs:attribute name="bias" type="xs:integer" use="required" />
    <xs:attribute name="year" type="xs:short" use="required" />
    <xs:attribute name="month" type="xs:short" use="required" />
    <xs:attribute name="dayofweek" type="xs:short" use="required" />
    <xs:attribute name="day" type="xs:short" use="required" />
    <xs:attribute name="hour" type="xs:short" use="required" />
    <xs:attribute name="minute" type="xs:short" use="required" />
  </xs:complexType>
  <xs:complexType name="timezone-type">
    <xs:annotation>
      <xs:documentation>This type is based of the TIME_ZONE_INFORMATION type from Win32
API.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="standard" type="tns:timezone-date-type" />
      <xs:element name="daylight" type="tns:timezone-date-type" />
    </xs:sequence>
    <xs:attribute name="bias" type="xs:integer" use="required" />
  </xs:complexType>
  <xs:complexType name="period-type">
    <xs:attribute name="dow" type="tns:days-of-week-type" use="required" />
    <xs:attribute name="start" type="tns:minutes-from-midnight-type" use="required" />
    <xs:attribute name="end" type="tns:minutes-from-midnight-type" use="required" />
  </xs:complexType>
  <xs:complexType name="period-array-type">
    <xs:sequence>
      <xs:element name="period" type="tns:period-type" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="refname-type">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Za-z0-9_]+" />
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="preamble-member-base-type">
    <xs:attribute name="name" type="tns:refname-type" use="required" />
  </xs:complexType>
  <xs:complexType name="wait-type">
    <xs:complexContent>
```

```
      <xs:extension base="tns:preamble-member-base-type">
        <xs:attribute name="seconds" type="xs:nonNegativeInteger" use="required" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="list-type">
    <xs:complexContent>
      <xs:extension base="tns:preamble-member-base-type">
        <xs:sequence>
          <xs:element name="target" type="tns:target-type" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="time-type">
    <xs:complexContent>
      <xs:extension base="tns:preamble-member-base-type">
        <xs:sequence>
          <xs:element name="timezone" type="tns:timezone-type" minOccurs="0" maxOccurs="1" />
        </xs:sequence>
        <xs:attribute name="range" type="xs:string" use="required" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="time-period-type">
    <xs:complexContent>
      <xs:extension base="tns:preamble-member-base-type">
        <xs:sequence>
          <xs:element name="timezone" type="tns:timezone-type" minOccurs="0" maxOccurs="1" />
          <xs:element name="periodarray" type="tns:period-array-type" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="flags-type">
    <xs:complexContent>
      <xs:extension base="tns:preamble-member-base-type">
        <xs:attribute name="value" type="xs:string" use="required" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="preamble-type">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="flags" type="tns:flags-type" />
        <xs:element name="time" type="tns:time-type" />
        <xs:element name="timeperiod" type="tns:time-period-type" />
        <xs:element name="wait" type="tns:wait-type" />
        <xs:element name="list" type="tns:list-type" />
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="minutes-from-midnight-type">
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="1440" />
    </xs:restriction>
  </xs:simpleType>
```

```
  <xs:simpleType name="day-of-week-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="sun" />
      <xs:enumeration value="mon" />
      <xs:enumeration value="tue" />
      <xs:enumeration value="wed" />
      <xs:enumeration value="thu" />
      <xs:enumeration value="fri" />
      <xs:enumeration value="sat" />
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="days-of-week-type">
    <xs:list itemType="tns:day-of-week-type" />
  </xs:simpleType>
  <!-- The following type definations are used in the script-->
  <xs:simpleType name="criteria-type">
    <xs:restriction base="xs:string">
      <xs:pattern value="!{0,1}dnd" />
      <xs:pattern value="!{0,1}umenabled" />
      <xs:pattern value="!{0,1}class:(primary|secondary)" />
      <xs:pattern value="!{0,1}registered" />
      <xs:pattern value="!{0,1}time:[A-Za-z0-9_]+" />
      <xs:pattern value="!{0,1}flags:[A-Za-z0-9_]+\(.*\)" />
      <xs:pattern value="!{0,1}member:[A-Za-z0-9_]+" />
      <xs:pattern value="!{0,1}workinghours" />
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="reference-type">
    <xs:attribute name="name" type="tns:refname-type" use="required" />
  </xs:complexType>

  <!-- Root document defintion -->
  <xs:complexType name="routing-type">
    <xs:annotation>
      <xs:documentation>The name and version attributes are both mandatory.
</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="preamble" type="tns:preamble-type" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" />
    <xs:attribute name="version" type="xs:integer" />
    <xs:attribute name="minSupportedClientVersion" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:element name="routing" type="tns:routing-type" />
</xs:schema>
```

7 Appendix B: Full Location Profile Format

Following is the full XML schema for the full location profile:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.microsoft.com/2007/03/LocationProfileDescription"
targetNamespace="http://schemas.microsoft.com/2007/03/LocationProfileDescription">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
      Service Request for Location Profile Schema
      Microsoft Unified Communications Group
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="LocationProfileDescription" type="LocationProfileDescriptionType"/>

  <xsd:element name="Name" type="xsd:string"/>
  <xsd:element name="ExternalAccessPrefix" type="xsd:string"/>
  <xsd:element name="OptimizeDeviceDialing" type="xsd:boolean"/>
  <xsd:complexType name="RuleType">
    <xsd:sequence>
      <xsd:element name="Pattern"   type="xsd:string"/>
      <xsd:element name="Translation" type="xsd:string"/>
    <xsd:element name="InternalEnterpriseExtension" type="xsd:boolean" minOccurs="0"/>
    <xsd:element name="ApplicableForDeviceDialing" type="xsd:boolean" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="LocationProfileDescriptionType">
    <xsd:sequence>
      <xsd:element ref="Name" minOccurs="1" maxOccurs="1"/>
      <xsd:element name="Rule" type="RuleType" minOccurs="1" maxOccurs="unbounded"/>
    <xsd:element ref="ExternalAccessPrefix" minOccurs="0" maxOccurs="0"/>
    <xsd:element ref="OptimizeDeviceDialing" minOccurs="0" maxOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```

8 Appendix C: Full Call Context Format

Following is the schema for call context data.

```
<?xml version="1.0"       encoding="UTF-8"?>
<xs:schema version="1.0"
targetNamespace="http://schemas.microsoft.com/2008/03/sip/conversationContext"
xmlns:tns="http://schemas.microsoft.com/2008/03/sip/conversationContext"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">

  <xs:annotation>
    <xs:documentation>Notes/Context associated with a conversation </xs:documentation>
  </xs:annotation>

  <xs:complexType name="XmlConvContextParticipantType">
    <xs:sequence>
      <xs:element name="uri" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="displayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfUri" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="onBehalfDisplayName" type="xs:string" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>


  <xs:complexType name="XmlConvContextParticipantCollectionType">
    <xs:sequence>
      <xs:element name="participant" type="tns:XmlConvContextParticipantType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>


  <xs:complexType name="XmlConvContextType" >
    <xs:sequence>
      <xs:element name="id" type="xs:token"  minOccurs="1" maxOccurs="1"/>
      <xs:element name="from" type="tns:XmlConvContextParticipantType" minOccurs="1"
maxOccurs="1"/>
      <xs:element name="to" type="tns:XmlConvContextParticipantType" minOccurs="1"
maxOccurs="1"/>
      <xs:element name="participants" type="tns:XmlConvContextParticipantCollectionType"
minOccurs="1" maxOccurs="1" />
      <xs:element name="date" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="mode" type="xs:token" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="conversationId" type="xs:token" minOccurs="1" maxOccurs="1"/>
      <xs:element name="dataFormat" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="contextData" type="xs:string" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="XmlConvContext" type="tns:XmlConvContextType" />

</xs:schema>
```

9 Appendix D: E911 PIDF Extension Format

Following is the full XML schema for the E911 PIDF extension:

```
<xs:schema xmlns:tns="urn:schema:Rtc.LIS.msftE911PidfExtn.2008"
attributeFormDefault="unqualified" elementFormDefault="qualified"
targetNamespace="urn:schema:Rtc.LIS.msftE911PidfExtn.2008"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="msftE911PidfExtn" type="tns:msftE911PidfExtn" />
  <xs:complexType name="msftE911PidfExtn">
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="ConferenceUri" type="xs:anyURI" />
      <xs:element minOccurs="1" maxOccurs="1" name="ConferenceMode"
type="tns:ConferenceModeEnum" />
      <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##other" processContents="lax"
/>
    </xs:sequence>
    <xs:anyAttribute namespace="##any" />
  </xs:complexType>
  <xs:simpleType name="ConferenceModeEnum">
    <xs:restriction base="xs:token">
      <xs:enumeration value="oneway" />
      <xs:enumeration value="twoway" />
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

The **msftE911PidfExtn** also contains an extensibility element that contains the value of the **LocationPolicyTagID** property returned in the **LocationPolicy** in-band provisioning group.

```
<LocationPolicyTagID xmlns="urn:schema:Rtc.Lis.LocationPolicyTagID.2008">location-policy-tag-
id-value</LocationPolicyTagID >
```

10 Appendix E: 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® Office Communications Server 2007
- Microsoft® Office Communications Server 2007 R2
- Microsoft® Office Communicator 2007
- Microsoft® Office Communicator 2007 R2
- Microsoft® Lync™ Server 2010
- Microsoft® Lync™ 2010

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: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<2> Section 2.2.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<3> Section 2.2.7.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<4> Section 2.2.7.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<5> Section 2.2.7.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<6> Section 2.2.7.2: Supported in Office Communications Server 2007 R2, Office Communicator 2007 R2 only.

<7> Section 2.2.8: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<8> Section 2.2.8: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<9> Section 2.2.13: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<10> Section 2.2.16: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<11> Section 2.2.17: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<12> Section 2.2.18: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<13> Section 2.2.19: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<14> Section 2.2.20: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<15> Section 2.2.21: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<16> Section 3.5.3: Supported in Office Communications Server 2007, Office Communicator 2007 only.

<17> Section 3.5.3: Supported in Office Communications Server 2007, Office Communicator 2007 only.

<18> Section 3.5.3: Supported in Office Communications Server 2007, Office Communicator 2007 only.

<19> Section 3.5.3: Supported in Office Communications Server 2007, Office Communicator 2007 only.

<20> Section 3.5.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<21> Section 3.5.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<22> Section 3.5.5.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<23> Section 3.5.5.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<24> Section 3.5.6: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<25> Section 3.5.6: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<26> Section 3.6: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<27> Section 3.6: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2: This behavior is not supported.

<28> Section 3.6.5.2: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<29> Section 3.6.5.3.1: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2: This behavior is not supported.

<30> Section 3.8.2.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<31> Section 3.8.2.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<32> Section 3.8.5.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<33> Section 3.8.5.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<34> Section 3.8.5.1.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<35> Section 3.8.5.1.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<36> Section 3.8.5.1.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<37> Section 3.8.5.1.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<38> Section 3.8.5.1.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<39> Section 3.8.5.1.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<40> Section 3.8.5.1.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<41> Section 3.8.5.1.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<42> Section 3.8.5.1.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<43> Section 3.8.5.1.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<44> Section 3.8.5.1.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<45> Section 3.8.5.1.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<46> Section 3.8.5.1.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<47> Section 3.8.5.1.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<48> Section 3.8.5.2.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<49> Section 3.8.5.2.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<50> Section 3.8.5.2.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<51> Section 3.8.5.2.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<52> Section 3.8.5.2.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<53> Section 3.8.5.2.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<54> Section 3.8.5.2.2.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<55> Section 3.8.5.2.2.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: The primary user timer is not supported.

<56> Section 3.8.5.2.2.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<57> Section 3.8.5.2.2.4: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<58> Section 3.8.5.5: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<59> Section 3.8.5.8: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<60> Section 3.8.5.10: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<61> Section 3.8.5.10: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<62> Section 3.8.6.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<63> Section 3.8.6.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<64> Section 3.8.6.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<65> Section 3.11: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<66> Section 3.12: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<67> Section 3.13: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<68> Section 3.14: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<69> Section 3.14.4.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<70> Section 3.14.5.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<71> Section 3.15: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: Extensions for Agent Anonymity. Ms-Call-Info and P-Agent-On-Behalf-Of are not supported.

<72> Section 3.16: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: E911 message processing is not supported.

<73> Section 4.6: This example does not apply to: Office Communications Server 2007, Office Communicator 2007.

<74> Section 4.7.4: Office Communications Server 2007, Office Communicator 2007. This behavior is not supported.

<75> Section 4.8: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<76> Section 4.11: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<77> Section 4.12.1: This example does not apply to: Office Communications Server 2007, Office Communicator 2007.

<78> Section 4.13: This example does not apply to: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2.

<79> Section 4.14: This example does not apply to Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, and Office Communicator 2007 R2.

11 Change Tracking

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

12 Index

A

Abstract data model
  agent anonymity extensions 74
    Ms-Call-Info header 74
    P-Agent-On-Behalf-Of Header 74
  call context extensions 69
  dialog state recovery
    SIP proxy 47
    user agent 47
  E911 75
  EPID mechanism 33
  federation extensions 64
    ms-ep-fqdn parameter 65
    ms-source-network parameter 65
    ms-source-type parameter 64
    ms-source-verified-user parameter 65
  firewall traversal aid 42
  GRUU mechanism 38
  ICE SDP interworking 72
  logging and monitoring extensions 68
  message routing with redundant server 45
  multipart MIME 72
  NAT traversal aid 42
  phone number resolution
    SIP proxy 52
    user agent 52
  public IM connectivity extensions 64
    ms-ep-fqdn parameter 65
    ms-source-network parameter 65
    ms-source-type parameter 64
    ms-source-verified-user parameter 65
  remote user extensions 66
  routing script preamble 54
  safe call transfer extension 71
  SIP.INSTANCE mechanism 35
Agent anonymity extensions
  abstract data model 74
    Ms-Call-Info header 74
    P-Agent-On-Behalf-Of header 74
  example 89
  higher-layer triggered events 75
  initialization 74
  local events 75
  message processing
    server 75
  overview 74
  sequencing rules
    server 75
  timer events 75
  timers 74
Applicability 15

C

Call context extensions
  abstract data model 69
  example 84
  higher-layer triggered events 69
  initialization 69
  local events 71
  message processing 70
    client 70
    server 71
  messages 27
    contextData element 30
    conversationId element 30
    dataFormat element 30
    date element 30
    from element 27
    Id element 27
    mode element 31
    participant element 29
    participants element 29
    to element 28
  overview 69
  schema 97
  sequencing rules 70
    client 70
    server 71
  timer events 71
  timers 69
Call Context Syntax message 27
  contextData element 30
  conversationId element 30
  dataFormat element 30
  date element 30
  from element 27
  id element 27
  mode element 31
  participant element 29
  participants element 29
  to element 28
Capability negotiation 15
Change tracking 104
Contact Header Field Extensions message 19
Content-Disposition Header Field Extension
  message 25

D

Data model - abstract
  agent anonymity extensions 74
    Ms-Call-Info header 74
    P-Agent-On-Behalf-Of header 74
  call context extensions 69
  dialog state recovery
    SIP proxy 47
    user agent 47
  E911 75
  EPID mechanism 33
  federation extensions 64
    ms-ep-fqdn parameter 65
    ms-source-network parameter 65
    ms-source-type parameter 64
    ms-source-verified-user parameter 65
  firewall traversal aid 42
  GRUU mechanism 38

ICE SDP interworking 72
logging and monitoring extensions 68
message routing with redundant server 45
multipart MIME 72
NAT traversal aid 42
phone number resolution
SIP proxy 52
user agent 52
public IM connectivity extensions 64
ms-ep-fqdn parameter 65
ms-source-network parameter 65
ms-source-type parameter 64
ms-source-verified-user parameter 65
remote user extensions 66
routing script preamble 54
safe call transfer extension 71
SIP.INSTANCE mechanism 35
Dialog state recovery
abstract data model
SIP proxy 47
user agent 47
example 79
higher-layer triggered events
user agent 48
initialization
user agent 48
local events 51
message 26
message processing
SIP proxy 48
user agent 49
overview 47
sequencing rules
SIP proxy 48
user agent 49
timer events
user agent 51
timers
user agent 48

E

E911
abstract data model 75
higher-layer triggered events 75
initialization 75
INVITE
example 90
local events 76
message processing 76
client 76
server 76
messages
call syntax 32
overview 75
PIDF Extension
schema 98
sequencing rules 76
client 76
server 76
timer events 76
timers 75
E911 Call Syntax message 32
EPID mechanism
abstract data model 33
higher-layer triggered events 34
user agent 34
initialization 33
user agent 34
local events 35
message processing 34
SIP proxy 35
SIP registrar 34
user agent 34
overview 33
sequencing rules 34
SIP proxy 35
SIP registrar 34
user agent 34
timer events 35
timers 33
EPID mechanism example 77
Examples
agent anonymity 89
call context extensions 84
dialog state recovery 79
E911 INVITE 90
EPID mechanism 77
federation extension 83
firewall traversal aid 78
GRUU mechanism 77
History-Info header field 82
message routing with redundant server 79
Multipart MIME
two-level 85
Multi-part MIME
three-level 86
NAT traversal aid 78
public IM connectivity extension 83
remote users extension 83
routing preamble
blocking preamble 81
call forward 81
simultaneous ring 81
team ring 82
SIP.INSTANCE mechanism 77
Extensions for Federation and Public IM
Connectivity message 25
Extensions for Remote Users message 25

F

Federation extension
abstract data model 64
ms-ep-fqdn parameter 65
ms-source-network parameter 65
ms-source-type parameter 64
ms-source-verified-user parameter 65
example 83
higher-layer triggered events 66
initialization 66
local events 66
message processing 66
client 66

server 66
messages 25
overview 64
sequencing rules 66
client 66
server 66
timer events 66
timers 65
Fields - vendor-extensible 15
Firewall traversal aid
abstract data model 42
example 78
higher-layer triggered events 42
user agent 42
initialization 42
local events 44
message processing 43
SIP server(proxy registrar) 43
overview 41
sequencing rules 43
SIP server(proxy registrar) 43
timer events 44
timers 42
From and To Header Field Extensions message 21

G

Glossary 10
GRUU mechanism
abstract data model 38
example 77
higher-layer triggered events 38
user agent 38
initialization
user agent 38
local events 41
message processing 39
SIP proxy 40
SIP registrar 39
overview 38
sequencing rules 39
SIP proxy 40
SIP registrar 39
timer events 41
timers 38

H

Higher-layer triggered events
agent anonymity extensions 75
call context extensions 69
dialog state recovery
user agent 48
E911 75
EPID mechanism 34
user agent 34
federation extensions 66
firewall traversal aid 42
user agent 42
GRUU mechanism 38
user agent 38
ICE SDP interworking
outgoing INVITE 73
logging and monitoring extensions
client 68
message routing with redundant server 46
multipart MIME
outgoing INVITE 73
NAT traversal aid 42
user agent 42
phone number resolution
user agent 52
public IM connectivity extensions 66
remote user extensions 67
routing script preamble 55
safe call transfer extension 72
SIP.INSTANCE mechanism 37
user agent 37
History-Info header field
example 82
extensions
messages 26
message processing 61
History-Info Header Field extensions message 26

I

ICE SDP interworking
abstract data model 72
higher-layer triggered events
outgoing INVITE 73
initialization 73
local events 74
message processing
415 response 74
INVITE 73
overview 72
sequencing rules
415 response 74
INVITE 73
timer events 74
timers 72
Implementer - security considerations 92
Index of security parameters 92
Informative references 14
Initialization
agent anonymity extensions 74
call context extensions 69
dialog state recovery
user agent 48
E911 75
EPID mechanism 33
user agent 34
federation extensions 66
firewall traversal aid 42
GRUU mechanism 38
user agent 38
ICE SDP interworking 73
logging and monitoring extensions 68
message routing with redundant server 45
multipart MIME 73
NAT traversal aid 42

phone number resolution
user agent 52
public IM connectivity extensions 66
remote user extensions 67
routing script preamble 55
safe call transfer extension 72
SIP.INSTANCE mechanism 36
user agent 36
Introduction 10

L

Local events
agent anonymity extensions 75
call context extensions 71
dialog state recovery 51
E911 76
EPID mechanism 35
federation extensions 66
firewall traversal aid 44
GRUU mechanism 41
ICE SDP interworking 74
logging and monitoring extensions 69
message routing with redundant server 46
multipart MIME 74
NAT traversal aid 44
phone number resolution 53
public IM connectivity extensions 66
remote user extensions 67
routing script preamble 64
safe call transfer extension 72
SIP.INSTANCE mechanism 38
Location profile
schema 96
Location Profile Syntax message 21
location profile description element 21
location profile rule element 21
Logging and monitoring extensions
abstract data model 68
higher-layer triggered events
client 68
initialization 68
local events 69
message processing 68
client 68
proxy 69
overview 67
sequencing rules 68
client 68
proxy 69
timer events 69
timers 68

M

Message processing
agent anonymity extensions
server 75
call context extensions 70
client 70
server 71
dialog state recovery
SIP proxy 48
user agent 49
E911 76
client 76
server 76
EPID mechanism 34
SIP proxy 35
SIP registrar 34
user agent 34
federation extensions 66
client 66
server 66
firewall traversal aid 43
SIP server(proxy registrar) 43
GRUU mechanism 39
SIP proxy 40
SIP registrar 39
ICE SDP interworking
processing 415 response 74
processing INVITE 73
logging and monitoring extensions 68
client 68
proxy 69
message routing with redundant server
SIP proxy 46
multipart MIME
processing 415 response 74
processing INVITE 73
NAT traversal aid 43
SIP server(proxy registrar) 43
phone number resolution
SIP proxy 53
public IM connectivity extensions 66
client 66
server 66
remote user extensions 67
client 67
server 67
routing script preamble
1XX responses generated 61
call processing element 55
generating 199 response 61
handling 2XX response 61
handling 303 response 60
handling 415 response 60
handling 605 response 60
History-Info header field processing 61
income INVITE 57
other responses 61
routing element 55
safe call transfer extension 72
SIP.INSTANCE mechanism 37
SIP proxy 37
SIP registrar 37
Message routing with redundant server
abstract data model 45
example 79
higher-layer triggered events 46
initialization 45

local events 46
message processing
SIP proxy 46
overview 44
sequencing rules
SIP proxy 46
timer events 46
timers
SIP proxy 45
Messages
Call Context Syntax 27
contextData element 30
conversationId element 30
dataFormat element 30
date element 30
from element 27
id element 27
mode element 31
participant element 29
participants element 29
to element 28
Contact Header Field Extensions 19
Content-Disposition Header Field Extension 25
E911 Call Syntax 32
Extensions for Federation and Public IM Connectivity 25
Extensions for Remote Users 25
From and To Header Field Extensions 21
History-Info Header Field extensions 26
Location Profile Syntax 21
location profile description element 21
location profile rule element 21
Ms-Call-Info Header Field Syntax 31
Ms-Correlation-Id Header Field Syntax 24
Ms-Forking Header Field Syntax 24
Ms-Sensitivity Header Field Syntax 24
Option Tag extensions 26
P-Agent-On-Behalf-Of Header Field Syntax 32
P-Dialog-Recovery-Action Header Field Syntax 26
Reason Header Field Extension 24
Record-Route Header Field Extension 19
Routing Script Preamble Syntax 22
flags element 23
identification 22
list element 23
target element 23
version 22
wait time element 23
SIP URI Parameter Extensions 16
Contact header field (section 2.2.1.2 18, section 2.2.1.3 18)
Path header field (section 2.2.1.1 17, section 2.2.1.3 18)
Record-Route header field (section 2.2.1.1 17, section 2.2.1.3 18)
Request-URI header field (section 2.2.1.2 18, section 2.2.1.3 18)
Route header field (section 2.2.1.1 17, section 2.2.1.2 18, section 2.2.1.3 18)
Syntax of Globally Routable User Agent URI 18
transport 16
Via Header Field Extensions 20
Ms-Call-Info header field
abstract data model 74
syntax 31
Ms-Call-Info Header Field Syntax message 31
Ms-Correlation-Id Header Field Syntax message 24
ms-ep-fqdn parameter 65
Ms-Forking Header Field Syntax message 24
Ms-Sensitivity Header Field Syntax message 24
ms-source-network parameter 65
ms-source-type parameter 64
ms-source-verified-user parameter 65
Multipart MIME
abstract data model 72
example
two-level 85
higher-layer triggered events
outgoing INVITE 73
initialization 73
local events 74
message processing
415 response 74
INVITE 73
overview 72
sequencing rules
415 response 74
INVITE 73
timer events 74
timers 72
Multi-part MIME
example
three-level 86

N

NAT traversal aid
abstract data model 42
example 78
higher-layer triggered events 42
user agent 42
initialization 42
local events 44
message processing 43
SIP server(proxy registrar) 43
overview 41
sequencing rules 43
SIP server(proxy registrar) 43
timer events 44
timers 42
Normative references 12

O

Option Tag extensions message 26
Overview (synopsis) 14

P

P-Agent-On-Behalf-Of header field
abstract data model 74

syntax 32
P-Agent-On-Behalf-Of Header Field Syntax message 32
Parameters - security index 92
P-Dialog-Recovery-Action Header Field Syntax message 26
Phone number resolution
abstract data model
SIP proxy 52
user agent 52
higher-layer triggered events
user agent 52
initialization
user agent 52
local events 53
message processing
SIP proxy 53
overview 51
sequencing rules
SIP proxy 53
timer events 53
timers 52
Preconditions 15
Prerequisites 15
Product behavior 99
Public IM connectivity extension
abstract data model 64
ms-ep-fqdn parameter 65
ms-source-network parameter 65
ms-source-type parameter 64
ms-source-verified-user parameter 65
example 83
higher-layer triggered events 66
initialization 66
local events 66
message processing 66
client 66
server 66
messages 25
overview 64
sequencing rules 66
client 66
server 66
timer events 66
timers 65

R

Reason Header Field Extension message 24
Record-Route Header Field Extension message 19
References
informative 14
normative 12
Relationship to other protocols 15
Remote users extension
abstract data model 66
example 83
higher-layer triggered events 67
initialization 67
local events 67
message processing 67
client 67
server 67
messages 25
overview 66
sequencing rules 67
client 67
server 67
timer events 67
timers 67
Routing preamble
example
blocking preamble 81
call forward 81
simultaneous ring 81
team ring 82
Routing script preamble
abstract data model 54
extensions for call processing and routing
overview 53
higher-layer triggered events 55
initialization 55
local events 64
message 22
flags element 23
identification 22
list element 23
target element 23
version 22
wait time element 23
message processing
1XX responses generated 61
call processing element 55
generating 199 response 61
handling 2XX response 61
handling 303 response 60
handling 415 response 60
handling 605 response 60
History-Info header field processing 61
incoming INVITE 57
other responses 61
routing element 55
schema 93
sequencing rules
1XX responses generated 61
call processing element 55
generating 199 response 61
handling 2XX response 61
handling 303 response 60
handling 415 response 60
handling 605 response 60
History-Info header field processing 61
incoming INVITE 57
other responses 61
routing element 55
timer events
call forwarding timer expiry 64
primary user timer expiry 64
registered endpoint timer expiry 63
secondary target timer expiry 64
timers
call forwarding 54
primary use 54

registered endpoints 54
secondary target 55
Routing Script Preamble Syntax message 22
flags element 23
identification 22
list element 23
target element 23
version 22
Routing Script Preamble Syntax message wait time element 23

S

Safe call transfer extension
abstract data model 71
higher-layer triggered events 72
initialization 72
local events 72
message processing 72
overview 71
sequencing rules 72
timer events 72
timers 72
Schemas
call context extensions 97
E911 PIDF Extension 98
location profile 96
routing script preamble 93
Security
implementer considerations 92
parameter index 92
Sequencing rules
agent anonymity extensions
server 75
call context extensions 70
client 70
server 71
dialog state recovery
SIP proxy 48
user agent 49
E911 76
client 76
server 76
EPID mechanism 34
SIP proxy 35
SIP registrar 34
user agent 34
federation extensions 66
client 66
server 66
firewall traversal aid 43
SIP server(proxy registrar) 43
GRUU mechanism 39
SIP proxy 40
SIP registrar 39
ICE SDP interworking
processing 415 response 74
processing INVITE 73
logging and monitoring extensions 68
client 68
proxy 69
message routing with redundant server
SIP proxy 46
multipart MIME
processing 415 response 74
processing INVITE 73
NAT traversal aid 43
SIP server(proxy registrar) 43
phone number resolution
SIP proxy 53
public IM connectivity extensions 66
client 66
server 66
remote user extensions 67
client 67
server 67
routing script preamble
1XX responses generated 61
call processing element 55
generating 199 response 61
handling 2XX response 61
handling 303 response 60
handling 415 response 60
handling 605 response 60
History-Info header field processing 61
incoming INVITE 57
other responses 61
routing element 55
safe call transfer extension 72
SIP.INSTANCE mechanism 37
SIP proxy 37
SIP registrar 37
SIP URI Parameter Extensions message 16
Contact header field (section 2.2.1.2 18, section 2.2.1.3 18)
Path header field (section 2.2.1.1 17, section 2.2.1.3 18)
Record-Route header field (section 2.2.1.1 17, section 2.2.1.3 18)
Request-URI header field (section 2.2.1.2 18, section 2.2.1.3 18)
Route header field (section 2.2.1.1 17, section 2.2.1.2 18, section 2.2.1.3 18)
SIP.INSTANCE mechanism
abstract data model 35
example 77
higher-layer triggered events 37
user agent 37
initialization 36
user agent 36
local events 38
message processing 37
SIP proxy 37
SIP registrar 37
overview 35
sequencing rules 37
SIP proxy 37
SIP registrar 37
timer events 38
timers 36
Standards assignments 15

Syntax of Globally Routable User Agent URI message 18

T

Timer events
agent anonymity extensions 75
call context extensions 71
dialog state recovery
user agent 51
E911 76
EPID mechanism 35
federation extensions 66
firewall traversal aid 44
GRUU mechanism 41
ICE SDP interworking 74
logging and monitoring extensions 69
message routing with redundant server 46
multipart MIME 74
NAT traversal aid 44
phone number resolution 53
public IM connectivity extensions 66
remote user extensions 67
routing script preamble
call forwarding timer expiry 64
primary user timer expiry 64
registered endpoint timer expiry 63
secondary target timer expiry 64
safe call transfer extension 72
SIP.INSTANCE mechanism 38
Timers
agent anonymity extensions 74
call context extensions 69
dialog state recovery
user agent 48
E911 75
EPID mechanism 33
federation extensions 65
firewall traversal aid 42
GRUU mechanism 38
ICE SDP interworking 72
logging and monitoring extensions 68
message routing with redundant server
SIP proxy 45
multipart MIME 72
NAT traversal aid 42
phone number resolution 52
public IM connectivity extensions 65
remote user extensions 67
routing script preamble
call forwarding 54
primary use 54
registered endpoints 54
secondary target 55
safe call transfer extension 72
SIP.INSTANCE mechanism 36
Tracking changes 104
Transport 16
Triggered events
agent anonymity extensions 75
call context extensions 69
dialog state recovery
user agent 48
E911 75
EPID mechanism 34
user agent 34
federation extensions 66
firewall traversal aid 42
user agent 42
GRUU mechanism 38
user agent 38
ICE SDP interworking
outgoing INVITE 73
logging and monitoring extensions
client 68
message routing with redundant server 46
multipart MIME
outgoing INVITE 73
NAT traversal aid 42
user agent 42
phone number resolution
user agent 52
public IM connectivity extensions 66
remote user extensions 67
routing script preamble 55
safe call transfer extension 72
SIP.INSTANCE mechanism 37
user agent 37

V

Vendor-extensible fields 15
Versioning 15
Via Header Field Extensions message 20