[MS-SDPEXT]:
Session Description Protocol (SDP) Version 2.0 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 Open Specification Promise or the Community Promise. 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.

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release (beta) version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release (beta) versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial version
04/25/2008	0.2		Revised and edited technical content
06/27/2008	1.0		Revised and edited technical content
08/15/2008	1.01		Revised and edited technical content
12/12/2008	2.0		Revised and edited technical content
02/13/2009	2.01		Revised and edited technical content
03/13/2009	2.02		Revised and edited 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.

Date	Revision History	Revision Class	Comments
03/18/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	4.0	Major	Significantly changed the technical content.
04/11/2012	4.0	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction 8
1.1 Glossary 8
1.2 References 9
1.2.1 Normative References 9
1.2.2 Informative References 11
1.3 Protocol Overview (Synopsis) 11
1.4 Relationship to Other Protocols 14
1.5 Prerequisites/Preconditions 14
1.6 Applicability Statement 14
1.7 Versioning and Capability Negotiation 14
1.8 Vendor-Extensible Fields 15
1.9 Standards Assignments 15

2 Messages 16
2.1 Transport 16
2.2 Message Syntax 16

3 Protocol Details 17
3.1 Details 17
3.1.1 Abstract Data Model 17
3.1.2 Timers 17
3.1.3 Initialization 17
3.1.4 Higher-Layer Triggered Events 17
3.1.5 Message Processing Events and Sequencing Rules 17
3.1.5.1 Supported Values and Parameters for the a=crypto Attribute 17
3.1.5.2 Specifying and Negotiating SSRTP 18
3.1.5.2.1 Processing and Negotiating SSRTP 19
3.1.5.2.2 Renegotiation of Encryption 21
3.1.5.3 Representing new Payload Types 21
3.1.5.4 Interpreting the Preference of Formats in the Format List 22
3.1.5.5 Format for Dual-Tone Multi-Frequency(DTMF) in SDP 22
3.1.5.6 Restriction on the Name of the RTP Payload for Redundant Audio Data 23
3.1.5.7 Restriction on the Name and sampling rate for comfort noise 23
3.1.5.8 Negotiating SRTP or SSRTP Optionally 23
3.1.5.9 Connection-Oriented Media Address Support 24
3.1.5.10 Limited support for setup and connection Attributes 24
3.1.5.10.1 Limited support for the a=setup Attribute 25
3.1.5.10.2 Limited support for the a=connection Attribute 25
3.1.5.11 Text Telephony Support 25
3.1.5.12 Early Media Support 26
3.1.5.12.1 Restriction to Receiving an SDP Answer in Provisional Response 26
3.1.5.12.2 Receiving an SDP Answer in Provisional Response and Starting Media Streams 26
3.1.5.12.3 SDP Answer in Provisional and Final Responses 26
3.1.5.12.4 ICE Processing When an SDP Answer is Received in the Provisional Response 27
3.1.5.13 Extensions for reliable provisional response processing and related offer/answer models 27
3.1.5.14 No Support for Renegotiation of SRTP or SSRTP Encryption Parameters 28
3.1.5.15 Labeling a Media Description with an a=label Attribute 28

3.1.5.16 Address types in the c= line 28
3.1.5.17 No Support for Optional Parameters in the a=rtcp Attribute 28
3.1.5.18 Application sharing media stream/type m=applicationsharing 29
3.1.5.18.1 a=x-applicationsharing-session-id attribute 29
3.1.5.18.2 a=x-applicationsharing-role attribute 29
3.1.5.18.3 a=x-applicationsharing-media-type attribute 29
3.1.5.18.4 a=mid attribute 30
3.1.5.19 Interpretation of o= line in the SDP 30
3.1.5.20 Deviations from ICE-06 30
3.1.5.20.1 General Outline of the ICE Methodology 30
3.1.5.20.2 ICE RE-INVITE Initiator 31
3.1.5.20.3 No Update of Candidates Between INVITE and ICE RE-INVITE 31
3.1.5.20.4 Extending the Transport to Connection-Oriented (TCP) 31
3.1.5.20.5 No IPv6 transport addresses 31
3.1.5.21 Deviation from ICE V19 31
3.1.5.21.1 Support for IPv6 transport addresses 31
3.1.5.21.1.1 a=x-candidate-ipv6 attribute 32
3.1.5.21.2 LITE implementation 32
3.1.5.21.3 Ice-options attributes 32
3.1.5.21.4 Ice-mismatch attributes 33
3.1.5.21.5 ice-ufrag and ice-pwd attributes 33
3.1.5.22 Deviation from ICE-TCP-07 33
3.1.5.22.1 Default Candidate 33
3.1.5.22.2 Local Candidate 33
3.1.5.23 Extensions for call hold and retrieve 33
3.1.5.23.1 Invoking hold 33
3.1.5.23.2 Clearing hold (retrieve) 34
3.1.5.24 Extension for video receive capabilities a=x-caps 34
3.1.5.25 Extensions to optimize the media path to a gateway 35
3.1.5.25.1 a=x-bypassid attribute 35
3.1.5.25.2 a=x-bypass attribute 36
3.1.5.25.3 a=x-mediasettings attribute 36
3.1.5.26 Extensions for diagnostic info in SDP messages 36
3.1.5.27 Extensions for Music-on-Hold 38
3.1.5.27.1 a=feature attribute 38
3.1.5.27.2 User agent behavior for a=feature attribute 38
3.1.5.28 Extensions for media bandwidth 38
3.1.5.28.1 a=x-mediabw attribute 39
3.1.5.28.2 User agent behavior for a=x-mediabw attribute 39
3.1.5.29 Extensions for declaring device capabilities 40
3.1.5.29.1 a=x-devicecaps attribute 40
3.1.5.29.2 User agent behavior for a=x-devicecaps attribute 40
3.1.5.30 Extensions for RTCP-based feedback messages 41
3.1.5.30.1 a=rtcp-rsize attribute 41
3.1.5.30.2 a=rtcp-fb attribute 41
3.1.5.30.3 User agent behavior for a=rtcp-rsize and a=rtcp-fb attributes 42
3.1.5.31 Extensions for Synchronization Source (SSRC) range allocation 42
3.1.5.31.1 a=x-ssrc-range attribute 43
3.1.5.31.2 User agent behavior for a=x-ssrc-range attribute 43
3.1.5.32 Extensions for Media Source ID (MSI) assignment 44
3.1.5.32.1 a=x-source-streamid attribute 44
3.1.5.32.2 User agent behavior for a=x-source-streamid attribute 44
3.1.5.33 Extensions for media source labeling 45

3.1.5.33.1 a=x-source attribute 45
3.1.5.33.2 User agent behavior of a=x-source attribute 45
3.1.5.34 Extensions for multiplexed media channels 45
3.1.5.34.1 Indicating multiplexed media channels in an SDP message 45
3.1.5.34.2 User agent behavior for negotiating multiplexed media channels 46
3.1.5.35 Extensions for multi-channel main-video modality negotiation 46
3.1.5.35.1 Requirements to negotiate a multi-channel main-video modality 47
3.1.6 Timer Events 47
3.1.7 Other Local Events 47

4 Protocol Examples 48
4.1 Generic Examples 48
4.1.1 Client Makes an Offer using ICE as described in IETFDRAFT-ICENAT-06 48
4.1.2 Client Receives Response with SSRTP to ICENAT-06 Offer 49
4.1.3 Client Makes an Offer using ICE as described in IETFDRAFT-ICENAT-19 50
4.1.4 Client Receives Response with SSRTP to ICENAT-19 Offer 52
4.2 Encryption Using SRTP Examples that Demonstrate Extensions 53
4.3 Offer/Answer Exchange for Various SRTP Encryption Scenarios 54
4.3.1 Offerer With SRTP or Client Scale-SRTP Encryption Optionally and Answerer With SRTP or Client Scale-SRTP Encryption Optionally 54
4.3.1.1 Offer 54
4.3.1.2 Answer 54
4.3.1.3 Noteworthy points 55
4.3.2 Offerer With SRTP or Client Scale-SRTP Optionally and Answerer With SRTP or Server SSRTP Encryption Optionally 55
4.3.2.1 Offer 55
4.3.2.2 Answer 55
4.3.2.3 Noteworthy points 55
4.3.3 Offerer With SRTP or Client Scale-SRTP Encryption Optionally and Answerer With SRTP Encryption Optionally 56
4.3.3.1 Offer 56
4.3.3.2 Answer 56
4.3.3.3 Noteworthy points 56
4.3.4 Offerer With SRTP or Client Scale-SRTP Encryption Optionally and Answerer Cannot Support SRTP or SSRTP Encryption 56
4.3.4.1 Offer 56
4.3.4.2 Answer 56
4.3.4.3 Noteworthy points: 57
4.3.5 Offerer With SRTP or Client Scale-SRTP Encryption Compulsorily and Answerer With SRTP Encryption Optionally 57
4.3.5.1 Offer 57
4.3.5.2 Answer 57
4.3.5.3 Noteworthy points 57
4.4 Restriction to the name and sampling rate for wide band comfort noise 57
4.5 Offer/Answer Exchange for application sharing 58
4.5.1 Offer 58
4.5.2 Answer 58
4.5.3 Noteworthy points 59
4.6 Offer/Answer Exchange with optimized media path to a gateway 59
4.6.1 Incoming call from gateway to client 59
4.6.2 Outbound call from client to gateway 62
4.7 Extensions for music-on-hold 64
4.7.1 Offer specifying music-on-hold 64

4.7.2 Offer removing music-on-hold 64
4.8 Offer/Answer Exchange for multi-channel main-video modality 64
4.8.1 Offer from client 65
4.8.2 Answer from MCU 67

5 Security 71
5.1 Security Considerations for Implementers 71
5.2 Index of Security Parameters 71

6 Appendix A: Product Behavior 72

7 Change Tracking 77

8 Index 78

1 Introduction

This document specifies a proprietary extension to the Session Description Protocol (SDP) to support audio/video and application sharing calls.

SDP is used to negotiate and establish session characteristics during call setup.

Unless explicitly specified, this protocol follows the offer/answer model to represent session characteristics using an SDP to establish a session.

This protocol is used to negotiate audio/video and application sharing call setup and adding video (or audio) to an existing audio (or video) only call.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

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

Augmented Backus-Naur Form (ABNF)
Internet Protocol version 4 (IPv4)
Internet Protocol version 6 (IPv6)
network address translation (NAT)
Remote Desktop Protocol (RDP)
server
Transmission Control Protocol (TCP)
UPDATE
user agent
User Datagram Protocol (UDP)

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

200 OK
audio video profile (AVP)
base64 encoding
Client Scale Secure Real-Time Transport Protocol (Client Scale-SRTP)
Common Intermediate Format (CIF)
conference
Content-Type header
contributing source (CSRC)
dialog
dual-tone multi-frequency (DTMF)
endpoint
forward error correction (FEC)
Host Candidate
Interactive Connectivity Establishment (ICE)
INVITE
Media Source ID (MSI)
ms-diagnostics-public header
Multipoint Control Unit (MCU)
Multipurpose Internet Mail Extensions (MIME)
participant

Real-Time Transport Protocol (RTP)
Relayed Candidate
Scale Secure Real-Time Transport Protocol (SSRTP)
SDP answer
SDP offer
Secure Real-Time Transport Protocol (SRTP)
Server Reflexive Candidate
Server Scale Secure Real-Time Transport Protocol (Server SSRTP)
session
Session Description Protocol (SDP)
Session Initiation Protocol (SIP)
SIP protocol client
SIP request
SIP response
stream
Synchronization Source (SSRC)
telecommunications device for the deaf (TDD)
Transport Layer Security (TLS)

The following terms are specific to this document:

multiple points of presence (MPOP): A condition in which a single user signs in from multiple devices. A user who has multiple points of presence can be contacted through any of these devices.

secure audio video profile (SAVP): A protocol that extends the audio-video profile specification to include the Secure Real-Time Transport Protocol, as described in [RFC3711].

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

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

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.

[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-ICETCP-07] Rosenberg, J., "TCP Candidates with Interactive Connectivity Establishment (ICE)", draft-ietf-mmusic-ice-tcp-07, July 2008, http://tools.ietf.org/html/draft-ietf-mmusic-ice-tcp-07

[IETFDRAFT-OFFANS-08] Sawada, T., Kyzivat, P., "SIP (Session Initiation Protocol) Usage of the Offer/Answer Model", April 2008, http://tools.ietf.org/html/draft-ietf-sipping-sip-offeranswer-08

[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

[MS-DTMF] Microsoft Corporation, "RTP Payload for DTMF Digits, Telephony Tones, and Telephony Signals Extensions".

[MS-ICE2] Microsoft Corporation, "Interactive Connectivity Establishment (ICE) Extensions 2.0".

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

[MS-OCPSTN] Microsoft Corporation, "Session Initiation Protocol (SIP) for PSTN Calls Extensions".

[MS-RTP] Microsoft Corporation, "Real-time Transport Protocol (RTP) Extensions".

[MS-RTPRADEX] Microsoft Corporation, "RTP Payload for Redundant Audio Data Extensions".

[MS-SIPRE] Microsoft Corporation, "Session Initiation Protocol (SIP) Routing Extensions".

[MS-SRTP] Microsoft Corporation, "Secure Real-time Transport Protocol (SRTP) Extensions".

[MS-SSRTP] Microsoft Corporation, "Scale Secure Real-time Transport Protocol (SSRTP) Extensions".

[RFC1521] Borenstein, N., and Freed, N., "MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1521, September, 1993, http://www.ietf.org/rfc/rfc1521.txt

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

[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

[RFC3262] Rosenberg, J., and Schulzrinne, H., "Reliability of Provisional Responses in the Session Initiation Protocol (SIP)", RFC 3262, June 2002, http://www.ietf.org/rfc/rfc3262.txt

[RFC3264] Rosenberg, J., and Schulzrinne, H., "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002, http://www.rfc-editor.org/rfc/rfc3264.txt

[RFC3389] Zopf, R., "Real-Time Transport Protocol (RTP) Payload for Comfort Noise (CN)", RFC 3389, September 2002, http://www.rfc-editor.org/rfc/rfc3389.txt

[RFC3551] Schulzrinne, H., and Casner, S., "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, July 2003, http://www.ietf.org/rfc/rfc3551.txt

[RFC3605] Huitema, C., "Real Time Control Protocol (RTCP) Attribute in Session Description Protocol (SDP)", RFC 3605, October 2003, http://www.ietf.org/rfc/rfc3605.txt

[RFC4145] Yon, D., and Camarillo, G., "TCP-Based Media Transport in the Session Description Protocol (SDP)", RFC 4145, September 2005, http://www.rfc-editor.org/rfc/rfc4145.txt

[RFC4235] Rosenberg, J., Schulzrinne, H., and Mahy, R., Ed., "An INVITE-Initiated Dialog Event Package for the Session Initiation Protocol (SIP)", RFC 4235, November 2005, http://www.rfc-editor.org/rfc/rfc4235.txt

[RFC4566] Handley, M., Jacobson, V., and Perkins, C., "SDP: Session Description Protocol", RFC 4566, July 2006, http://www.ietf.org/rfc/rfc4566.txt

[RFC4568] Andreasen, F., Baugher, M., and Wing, D., "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, July 2006, http://www.rfc-editor.org/rfc/rfc4568.txt

[RFC4571] Lazzaro, J., "Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport", RFC 4571, July 2006, http://www.ietf.org/rfc/rfc4571.txt

[RFC4574] Levin, O., and Camarillo, G., "The Session Description Protocol (SDP) Label Attribute", RFC 4574, August 2006, http://www.rfc-editor.org/rfc/rfc4574.txt

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and Rey, J., "Extended RTP Profile for Real-Time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006, http://www.ietf.org/rfc/rfc4585.txt

1.2.2 Informative References

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

[MS-H264PF] Microsoft Corporation, "RTP Payload Format for H.264 Video Streams Extensions".

[MS-ICE] Microsoft Corporation, "Interactive Connectivity Establishment (ICE) Extensions".

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

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.txt

1.3 Protocol Overview (Synopsis)

Initiation of a multimedia **session (3)** or conference requires the exchange of the media details, transport addresses, and other metadata between the parties involved. This exchange facilitates the negotiation of media characteristics for establishing the session (3). The media characteristics associated with a session (3) are specified in **Session Description Protocol (SDP)**. The exchange and negotiation of the session (3) properties follows the specification of the offer/answer model with the SDP. In applications, these protocols are used to negotiate and establish a multimedia session (3).

It is a common requirement that the media being exchanged in a multimedia session (3) be protected using some form of encryption. When the **Real-Time Transport Protocol (RTP)** is used to exchange media, the media can be protected using the **Secure Real-Time Transport Protocol (SRTP)**, which requires exchange of attributes related to SRTP, such as cryptographic parameters. These characteristics can also be negotiated using SDP when the cryptographic characteristics of the media stream are described by a new SDP attribute name crypto, as described in [RFC4568].

After the session (3) is established, media can flow between the participating parties. Often, other networking components, such as **network address translation (NAT)**, are present between two parties and prevent media from traversing between the two parties. In such cases, **Interactive Connectivity Establishment (ICE)** can be used to facilitate media traversal through these network components. For information about the proprietary extension to the ICE protocol, see [MS-

ICE]. ICE specifies a protocol for setting up the audio/video RTP streams in a way that allows the streams to perform NAT.

This protocol uses and extends these protocols to support multimedia sessions (3). This protocol extension consists of the following additions, enhancements, and restrictions:

- Specifies the parameter and values that are supported for the **a=crypto**, as specified in the security description for media streams.
- SRTP and **Scale Secure Real-Time Transport Protocol (SSRTP)** encryption parameters are not renegotiated once the session (3) is established.
- An SDP attribute named **a=cryptoscale** is used for the negotiation of all the cryptographic parameters associated with SSRTP.
- **RTAudio**, **G722-Stereo**, **RTVideo** and **H.264UC** codecs, in addition to **RTData**, are supported.
- A media format for video **forward error correction (FEC)**, **ULPFEC-UC**.
- SRTP encryption can be used optionally. This option allows for support of remote peers that do not support Secure-RTP.
- **Transmission Control Protocol (TCP)** media addresses in SDP are not used when ICE is not used. This is a deviation from the specification of TCP-based media transport in the SDP.
- TCP media addresses in SDP are not used in the first **Session Initiation Protocol (SIP) INVITE** method when ICE is used.
- Addresses are not used for the **rtcp** attribute**.**
- Limited support for the **setup** attribute. This is a deviation from the TCP-based media transport in the SDP.
- Limited support for the **connection** attribute. This is a deviation from the TCP-based media transport in the SDP.
- This protocol handles early media only in very specific scenarios in a constrained manner and does not support early media in any other scenarios. Section 3 has more details on these scenarios and constraints.
- Special handling of **Internet Protocol version 6 (IPv6)** addresses in SDP.
- Several deviations from the ICE specifications.
- The session version on the **o=** line is not incremented.
- The format for **dual-tone multi-frequency (DTMF)** in SDP.
- A restriction on the name of the RTP payload for redundant audio data.
- A restriction on the name and sampling rate for comfort noise.
- A media type **m=applicationsharing**, which identifies an RDP-based application sharing media stream, or session (3), over RTP. In the context of the application sharing media stream, **m=applicationsharing**, four new attributes are defined:

  - **a=x-applicationsharing-session-id:** Identifies an RDP session (3).

- **a=x-applicationsharing-role:** Determines the party sharing role.
- **a=x-applicationsharing-media-type** ("sharer" or "viewer"). Negotiates the RDP media type.
- **a=mid:** An identifier of the media described by the containing **m=** line.

- A media level attribute **a=tty**, which indicates that a **user agent** has been configured to optimize the transfer of tones used in text telephony.
- A video media level attribute **a=x-caps**, which indicates the video capabilities supported by a video receiver.
- A **Multipurpose Internet Mail Extensions (MIME)** type **application/GW-SDP** is defined. This MIME type holds the gateway SDP or SIP trunk SDP.
- A media level attribute **a=x-bypassid**, which is a declarative attribute used to indicate the location of the media endpoint or media processor associated with this SDP. It is used to optimize the media path with a gateway in the same location.
- A media level attribute **a=x-bypass**, which is a declarative attribute that signifies that the media line with which it is associated involves bypass. It is a media level attribute sent in an answer when the answerer has chosen the bypass path.
- A media level attribute **a=x-mediasettings**, which contains the stream capabilities of the sender.
- A media level attribute **a=x-ms-SDP-diagnostics**, which is used to notify recipient of additional diagnostic information for that media line.
- A media level attribute **a=feature:MoH**, associated with an m=audio media type, which is a declarative attribute to indicate the media channel is streaming music-on-hold media.
- A session level attribute **a=x-mediabw** to declare the maximum send and receive bandwidth available for a modality.
- A session level attribute **a=x-devicecaps** to declare the device capabilities on an **endpoint (5)**.
- A media level attribute **a=rtcp-fb** to declare that certain RTCP-based feedback messages can be sent and received for the media stream.
- A media level attribute **a=rtcp-rsize** to indicate that certain RTCP-based feedback messages are transmitted in a Reduced-Size format defined in [MS-RTP].
- A media level attribute **a=x-ssrc-range** to declare the range used for **Synchronization Source (SSRC)** identifiers for the send stream.
- A media level attribute **a=x-source-streamid** to declare a **Media Source ID (MSI)** for a media stream.
- A media level attribute **a=x-source** to declare the name of a media source for a media description.
- A protocol to negotiate multiplexed media streams.
- A protocol to negotiate a multi-channel main-video modality.

For details about these extensions and deviations, see Section 3.

1.4 Relationship to Other Protocols

This protocol depends on:

- SDP, as described in [RFC4566], for media negotiation.
- SIP, as described in [RFC3261], for establishing and initializing a session (3).
- SDP for media streams, as described in [RFC4568], for media encryption.
- An offer/answer model for SDP, as described in [RFC3264], to represent session (3) characteristics used with SDP.
- A methodology for network address translation (NAT) traversal for offer/answer protocols, as described in [IETFDRAFT-ICENAT-06] and [IETFDRAFT-ICENAT-19], for media to traverse NAT and firewalls.

1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

This protocol is applicable to the following:

- The negotiation of SSRTP parameters between the two communicating peers. The SSRTP encryption can be used by an application in conference scenarios when communicating with a **Multipoint Control Unit (MCU)**.
- The ability to negotiate whether the media is encrypted using SRTP/SSRTP. The ability to negotiate SRTP encryption or SSRTP encryption optionally enables the application to communicate using these encryption mechanisms when the remote peer cannot support either of these encryptions.
- The ability to support five new codecs.
- The ability to support a new video FEC payload format.
- The ability to do connection-oriented (TCP) media in selected scenarios.
- The ability to do early media in selected scenarios.
- The negotiation of video-receive capabilities between two video peers. The ability to negotiate video-receive capabilities enables a video source to know what a video receiver is capable of receiving, such as frame rate, resolution, bit rate, and the number of video streams.
- The ability to support RTCP-based feedback messages, transmitted in a Reduced-Size format.
- The ability to support negotiation of an SSRC range on a media stream.

1.7 Versioning and Capability Negotiation

No version number is defined in this protocol. Session (3) characteristics are negotiated using SDP and the offer/answer model for SDP.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

As an extension to SDP, this protocol prescribes the format of session (3) descriptions intended to support audio, video, and application sharing calls, and can use any transport protocol used to carry SDP messages.

SIP is a commonly used transport for SDP messages. In this case, session descriptions, represented as SDP messages, MUST be included in the body of SIP messages. The **Content-Type header** of such a SIP message MUST contain the type or sub-type "application/sdp" or "application/gw-sdp". For more information about the Content-Type header, see [RFC1521] section 4. The **application/gw-sdp** MIME type holds the gateway SDP or SIP trunk SDP and contains *x-bypassid* as a parameter. A client can use this parameter to decide if further parsing of the SDP is needed, thereby optimizing processing.

SIP messages can be transported over TCP or **Transport Layer Security (TLS)**. TLS SHOULD be used to protect the encryption key, as the key is passed in the SIP/SDP signaling.

2.2 Message Syntax

The messages for this protocol are SDP messages. An SDP message contains the description of a media session (3). The session (3) and media characteristics are described by a set of **<type>=<value>** lines, as specified in [RFC4566]. The extensions are defined as custom SDP attributes.

For additional syntax and protocol details, see section 3.1.5.

3 Protocol Details

3.1 Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

All the message processing events and sequencing rules for media negotiation conform to the SDP specifications in [RFC4566] and [RFC3264], with some exceptions or modifications for the custom attributes introduced in this document.

Also note that the behavior described in [RFC4566] and [RFC3264] does not follow a simple client/server model. The two parties involved in an SDP exchange are peers. Which peer creates or modifies a session (3) changes which peer is the offerer or answerer as specifieded in [RFC3264]. In addition, some SDP attributes follow an offer/answer behavior and some SDP attributes supply information to the other peer with no answer expected.

3.1.5.1 Supported Values and Parameters for the a=crypto Attribute

The **a=crypto** attribute is as specified in [RFC4568], with the exception that a single white space MUST be used. The attribute has the following format, expressed using **Augmented Backus-Naur Form (ABNF)** notation, as defined in [RFC5234].

```
a=crypto tag WSP crypto-suite WSP key-params *(WSP session-param)
```

tag field: The **tag** field is used to specify a decimal number to identify a particular cryptographic attribute in the SDP security description for media streams, as specified in [RFC4568]. In the current extension, the semantics of the **tag** field is more restricted, in that the decimal value MUST be unique across the **a=crypto** and **a=cryptoscale** attributes. **a=cryptoscale** is a new attribute defined by this protocol and is specified in more detail in section 3.1.5.2.

crypto-suite field: The **crypto-suite** field is used to specify cryptographic methods or algorithms for media encryption. The only **crypto-suite** option supported is AES_CM_128_HMAC_SHA1_80. In other words, **crypto-suite** MUST be "AES_CM_128_HMAC_SHA1_80". In [RFC4568], this is defined in the context of "RTP/SAVP" as the transport. In the current extensions, use of this field is extended to the case when the transport is "RTP/AVP" in an **SDP offer**. This deviation from [RFC4568] is required to support negotiation of SRTP optionally, as specified in section 3.1.5.8.

key-params field: The **key-params** field is used to specify the keying information. The **key-params** are further defined in [RFC4568], as follows:

```
key-params = <key-method> ":" <key-info>
```

More than one **key-params** instance per line of **a=crypto** MUST NOT be used.

The **key-method** subfield is used to specify the provisional method of the keying information. As specified in [RFC4568], the only method that MUST be used is "inline", indicating that the keying material is provided in the **key-info** field.

The **key-info** field is specified in [RFC4568]. The specification of **key-info** in [RFC4568] is specifically targeted to the "RTP/SAVP" transport. In the current extension the **key-info** field can be used for both "RTP/SAVP" and "RTP/AVP". This extension is required to support negotiation of SRTP optionally, as specified in section 3.1.5.8.

Following is the format specified in [RFC4568] for the **key-info** field.

```
"inline:" <key||salt> ["|" lifetime] ["|" MKI ":" length]
```

Following is a list of constraints and values accepted for the **key-info** field:

- "MKI" SHOULD be used. If MKI is used the MKI *length* MUST be 1 byte.
- The value for lifetime MUST be "2^31" in SDP offers and answers sent.
- The value of lifetime MUST be ignored in SDP offers and answers received, and "2^31" MUST be used instead.

session-param field: The **session-param** field MUST NOT be used.

The following is an example **a=crypto** attribute:

```
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:t20I47Tyj1NDG6H+gWNpIzAzRPfYeQg8pP+ukwoy|2^31|1:1
```

Horizontal tab (HTAB code as defined in Augmented Backus-Naur Form (ABNF)) between tokens MUST NOT be used by the application.

3.1.5.2 Specifying and Negotiating SSRTP

The new **a=cryptoscale** attribute is introduced to support SSRTP encryption of the media in an audio/video session (3). This attribute has the following format expressed in ABNF notation, as defined in [RFC5234]:

```
"a=cryptoscale:" tag WSP scale-srtp-flavor WSP crypto-suite WSP key-params *(session-param)
```

The definition of **tag**, **crypto-suite**, **key-params**, and **session-param** are the same as that specified for the **a=crypto** attribute. All the extensions to or deviations from [RFC4568] related to the **a=crypto** attribute, as specified in section 3.1.5.1, also apply to the **a=cryptoscale** attribute.

Note that the **tag** field MUST be a unique decimal value across all the **a=crypto** and **a=cryptoscale** attributes.

The new field, **scale-srtp-flavor**, is specified as follows:

```
scale-srtp-flavor="client" | "server"
```

An application supporting media encryption using **Client Scale Secure Real-Time Transport Protocol (Client Scale-SRTP)** chooses a value of "client" for the **scale-srtp-flavor** field. An application supporting media encryption using Server Scale Secure Real-Time Transport Protocol (Server SSRTP) chooses a value of "server" for the **scale-srtp-flavor** field. An application MUST use either the Client Scale-SRTP encryption or the **Server SSRTP** encryption. It MUST NOT use both at the same time.

To negotiate SSRTP successfully, one peer must identify itself (via the **scale-srtp-flavor** field) to be the SSRTP "client" and the other must identify itself as the SSRTP "server". The SSRTP client supports Client Scale-SRTP encryption, and the SSRTP server supports Server SSRTP encryption. Which peer is the SSRTP server and which peer is the SSRTP client is independent of which peer sent the SDP offer. Typically, only a Multipoint Control Unit (MCU) would operate as an SSRTP server.

The SSRTP server encrypts media it sends to the SSRTP client using SSRTP encryption. SSRTP encryption is defined in [MS-SSRTP]. The SSRTP client decrypts SSRTP-encrypted media it receives from the SSRTP server.

The SSRTP client encrypts media it sends to the SSRTP server using SRTP encryption. SRTP encryption is defined in [MS-SRTP]. The SSRTP server decrypts SRTP-encrypted media it receives from the SSRTP client.

The fields of the attribute **a=crypto** and **a=cryptoscale** are themselves not encrypted. Protection of the fields and encryption information is provided by the TLS transport over which the SIP signaling is carried.

The following is an example **a=cryptoscale** attribute given by a peer which supports Client Scale-SRTP encryption:

```
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:85Sm2QWogZ9N256qxTRhfIRxjUp9q1ISMxwbiloc|2^31|1:1
```

3.1.5.2.1 Processing and Negotiating SSRTP

The **a=cryptoscale** attribute is used to negotiate SSRTP encryption of media.

The following table specifies how an application can communicate its SRTP and SSRTP encryption preferences.

Protocol element	Description
a=crypto:<tag> <crypto-suite> <key-params> [<session-params>]	Supports SRTP encryption.
a=cryptoscale:<tag> client <crypto-suite> <key-params> [<session-params>]	Supports the client flavor of SSRTP encryption.
a=cryptoscale:<tag> server <crypto-suite> <key-params> [<session-params>]	Supports the server flavor of SSRTP encryption.

With the current extensions, an application expresses its ability to support SRTP and SSRTP by specifying the **a=crypto** and **a=cryptoscale** attributes, respectively, in an SDP message as the body of a **SIP request**.

An application MUST propose to support only one type of the SSRTP encryption, either "client" or "server", in the SDP message. The application MUST NOT add both client and server types of SSRTP to the SDP message.

An application which supports SSRTP encryption SHOULD also support SRTP encryption.

An application SHOULD respond to the SIP request with only one preferred encryption in the SDP message in the **SIP response**, out of all the proposed encryptions specified in the SDP message of the SIP request.

The following table summarizes all the possible combinations of successful SRTP or SSRTP negotiation. The behavior applies to both an initial SIP INVITE, as specified in [RFC3261], and a re-INVITE to add new modality.

SDP offer contains	SDP answer contains	Result encryption from offerer to answerer	Result encryption from answerer to offerer
SRTP	SRTP	SRTP encrypted	SRTP encrypted
Client SSRTP	Server SSRTP	SRTP encrypted	SSRTP encrypted
Server SSRTP	Client SSRTP	SSRTP encrypted	SRTP encrypted
SRTP and Client SSRTP	SRTP	SRTP encrypted	SRTP encrypted
SRTP and Client SSRTP	Server SSRTP	SRTP encrypted	SSRTP encrypted
SRTP and Server SSRTP	SRTP	SRTP encrypted	SRTP encrypted
SRTP and Server SSRTP	Client SSRTP	SSRTP encrypted	SRTP encrypted

If neither SSRTP nor SRTP encryption can be negotiated successfully for a modality, then either the media between the peers will be unencrypted; or, if at least one peer requires the media to be encrypted, the offer/answer negotiation will either fail or the modality not be established.

An application can specify multiple **a=crypto** and **a=cryptoscale** attributes in an SDP offer, but there MUST NOT be **a=crypto** or **a=cryptoscale** attributes which have **key-params** fields which are identical except for their **<key||salt>** values. For example, in an SDP offer containing the following **a=crypto** attributes:

```
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:t20I47Tyj1NDG6H+gWNpIzAzRPfYeQg8pP+ukwoy|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:85Sm2QWogZ9N256qxTRhfIRxjUp9q1ISMxwbiloc|2^31|1:1
```

there is nothing distinguishing these attributes which would cause an answerer to prefer selecting one attribute over the other. In this example SDP offer, the **a=crypto:3** attribute is unnecessary.

3.1.5.2.2 Renegotiation of Encryption

An application MUST NOT use SIP re-INVITE to re-negotiate the encryption type, SRTP or SSRTP, or any other parameter in the **a=crypto** or **a=cryptoscale** lines.

3.1.5.3 Representing new Payload Types

This protocol adds support for six new payload types:

- **RTAudio** for audio streams.
- **G722-Stereo** for audio streams.
- **RTVideo** for video streams.
- **H.264UC** for video streams.
- **RTData** for application sharing streams.
- **ULPFEC-UC** for video streams.

The media formats of these payload types are described by the parameters in the following table that SHOULD<1> be specified using the **a=rtpmap:** and **a=fmtp:** attributes for dynamic payload types, as specified in [RFC4566].

Payload	Encoding name	Clock rate	Bit rate
RTAudio	x-msrta	16000	29000
RTAudio	x-msrta	8000	11800
RTVideo	x-rtvc1	90000	Not applicable: bitrate *fmtp* parameter is significant only for audio. bitrate SHOULD not be present in an SDP offer or answer for RTVideo and MUST be ignored.
RTData	x-data	90000	Not applicable: bitrate *fmtp* parameter is significant only for audio. bitrate SHOULD not be present in an SDP offer or answer for RTData and MUST be ignored.
H.264UC<2>	X-H264UC	90000	Not applicable: bitrate *fmtp* parameter is significant only for audio. bitrate SHOULD NOT be present in an SDP offer or answer for H.264UC and MUST be ignored. However, the **a=fmtp:** attribute for H.264UC MUST specify two parameters: **packetization-mode=1** and **mst-mode=NI-TC**. These two parameters can appear in any order and are case-insensitive. Also, an **a=x-ssrc-range** media-level attribute MUST be present. The length of the range given MUST be at least 30. If either the **a=fmtp** or **a=x-ssrc-range** attribute requirement is not met, the H.264UC video codec in an SDP offer or answer MUST be ignored.

Payload	Encoding name	Clock rate	Bit rate
G722-Stereo<3>	G722	8000/2 **Notes:** The actual clock rate of the codec is 16000, but MUST be listed in SDP as 8000. The "/2" is not part of the clock rate but indicates that the codec has 2 channels.	128000
ULPFEC-UC<4>	x-ulpfecuc	90000	Not Applicable. bitrate *fmtp* parameter is significant only for audio. bitrate SHOULD not be present in an SDP offer or answer for ULPFEC-UC and MUST be ignored.

As an example, the following SDP message fragment shows the **RTAudio** payload type and **G722-Stereo** payload type of an audio stream:

```
m=audio 37632 RTP/AVP 117 114 ...
…
a=rtpmap:117 G722/8000/2
a=fmtp:117 bitrate=128000
…
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
…
```

Negotiation of these payload types are similar to the negotiation of other payload types, as specified in [RFC3264]. Any dynamic payload type can be chosen for these payloads following the RTP profile for audio and video conferences with minimum control specification in [RFC3551]<5>. Specifying these parameters in the **a=rtpmap:** attribute in a media description section of an SDP message indicates the preference of these codecs for that payload type.

Applications that do not support these codecs MUST NOT advertise these codecs in an SDP message. In the case of RTP, if a particular codec was referenced with a specific payload type number specified in the **a=rtpmap:** attribute in the offer, that same payload type number MUST be used for that codec in the answer.

If a user agent supports the **H.264UC** video codec, it SHOULD also support the **ULPFEC-UC** video FEC media format. For more information on H.264UC and ULPFEC-UC, refer to [MS-H264PF].

3.1.5.4 Interpreting the Preference of Formats in the Format List

In this protocol the media formats specified in an **m=** line for a particular media stream SHOULD be listed in order of preference. For the media formats it can support, the answering user agent SHOULD<6> use the same relative preference ordering as in the offer.

3.1.5.5 Format for Dual-Tone Multi-Frequency(DTMF) in SDP

The RTP payload type number used to specify DTMF, as specified in [MS-DTMF], in the **m=** line of the SDP message SHOULD be "101"<7>.

3.1.5.6 Restriction on the Name of the RTP Payload for Redundant Audio Data

The name of the payload used for redundant audio data, as specified in [MS-RTPRADEX], MUST be "RED" and is case-sensitive.

3.1.5.7 Restriction on the Name and sampling rate for comfort noise

The name of the payload used for comfort noise SHOULD<8> be "CN" and the sampling rate SHOULD<9> be 8,000 or 16,000. For more information, see [RFC3389].

3.1.5.8 Negotiating SRTP or SSRTP Optionally

To require SRTP encryption for a media stream, an application can use the SRTP, as specified in [RFC4568], to specify the **secure audio video profile (SAVP)** in an **m=** line of an SDP message, as part of the SRTP negotiation. This is shown in the following example.

```
m=audio 50004 RTP/SAVP 8 97 101
```

This description, however, does not allow for the possibility to negotiate SRTP encryption optionally, in that the support of the SRTP encryption is desired but not required.

The mechanism described here to negotiate SRTP optionally also applies to SSRTP encryption.

To support SRTP or SSRTP encryption optionally, this protocol deviates from the specification in [RFC4568]; in a SIP INVITE request, an application MUST use **audio video profile (AVP)** in the **m=** line of the SDP offer, together with the **a=crypto** or **a=cryptoscale** attribute to negotiate media encryption using SRTP or SSRTP. The application SHOULD bypass the negotiation of SRTP or SSRTP encryption by not specifying any **a=crypto** and **a=cryptoscale** attributes. To acknowledge the ability to support the SRTP or SSRTP encryption, the remote peer MUST respond to the SIP request in a SIP **200 OK** response with an SDP message specifying "SAVP" in the **m=** line and the **a=crypto** or **a=cryptoscale** attribute, respectively for SRTP or SSRTP, as part of the media description. All subsequent SIP re-INVITE requests MUST continue to have "SAVP". If the remote peer cannot support SRTP or SSRTP encryption, the remote peer MUST specify "AVP" in the **m=** line of the **SDP answer** and MUST NOT specify any **a=crypto** and **a=cryptoscale** attributes.

The following are examples of negotiating encryption.

The following example is a peer that sends an SDP offer in a SIP request to specify that it can support either SRTP or SSRTP encryption, but the support is not mandatory.

```
m=audio 50004 RTP/AVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

If the peer is capable of supporting, and does support, SRTP encryption, the following example is a response to the previous request with an SDP message.

```
m=audio 50014 RTP/SAVP 8 97 101
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:v0ncVM8eKP2bkOINeRaqcFeqjXwGMXo0sRalidZc|2^31|1:1
```

If the peer is not capable of supporting, or does not support, SRTP encryption, the following example is a response to the previous request with an SDP message.

```
m=audio 50104 RTP/AVP 8 97 101
```

The following example is a peer that sends an SDP offer in a SIP request to mandate either SRTP or SSRTP encryption support.

```
m=audio 50004 RTP/SAVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

3.1.5.9 Connection-Oriented Media Address Support

In the SDP message, as specified in [RFC4566], the **m=** line is used to specify the transport for the given media type. The supported transport can be either **User Datagram Protocol (UDP)** or TCP. TCP is a connection-oriented transport by which the media is exchanged. UDP is not connection-oriented. The following is an SDP message fragment showing an **m=** line specifying that TCP be used as the media transport, as defined in [RFC4571].

```
m=audio 50004 TCP/RTP/AVP 8 97 101
```

However, the connection-oriented transport SHOULD NOT be used when ICE, as defined in [IETFDRAFT-ICENAT-06] or [IETFDRAFT-ICENAT-19], is not enabled on the offering application and the answering application. This applies to any offer or answer received from an application that does not support ICE.

If the offer or answer is received from an application that supports ICE, according to the ICE specifications in [IETFDRAFT-ICENAT-06] and [IETFDRAFT-ICENAT-19], the port and the transport specified in the **m=** line are referred to as the active address or the address that will be tried first in the ICE methodology to establish connection.

Application sharing SHOULD<10> use [MS-ICE2] over TCP.

The ICE scenario is applicable if both peers support ICE, which can be established by examining the SDP offer and SDP answer.

For audio or video media, or other media types, a connection-oriented, or TCP, transport for the active addresses in the first SDP offer or SDP answer MUST NOT be used. A subsequent SDP offer/answer exchanged in a re-INVITE might have a connection-oriented transport for the active address, depending on the operation of ICE.

For audio or video media, or other media types, any connection-oriented transport specified in the **m=** line in the first SDP offer or SDP answer SHOULD<11> be rejected, as specified in [RFC3264] section 6.

3.1.5.10 Limited support for setup and connection Attributes

TCP-based media transport in the SDP message, as specified in [RFC4145], adds two new attributes to the SDP message. These are the **a=setup** and **a=connection** attributes. These attributes are used to establish and maintain TCP connections for the media exchange. However, the support for these attributes is limited in this protocol. These limitations are discussed in this section.

3.1.5.10.1 Limited support for the a=setup Attribute

TCP-based media transport in the SDP, as specified in [RFC4145], states that the **a=setup** attribute can have the following roles for the purpose of establishing a TCP connection:

- "active"
- "passive"
- "actpass"
- "holdconn"

When used in the context of this protocol, the **a=setup** attribute MUST have one of the following two values:

- "active"
- "passive"

The behavior of the roles "active" and "passive" are the same as specified in [RFC4145] with the following exception.

If the initial offer has a value of **a=setup:active**, the answer also has a value of **a=setup:active**, but the offerer role is still considered to be active, because it is the endpoint (5) that is initiating the outgoing connection. Subsequent offers and answers contain the correct values of "active" and "passive".

3.1.5.10.2 Limited support for the a=connection Attribute

The TCP-based media transport in the SDP, as specified in [RFC4145], states that the **a=connection** attribute can have the following values to indicate the status of a TCP connection.

"new"

"existing"

When used in the context of this protocol, the **a=connection** attribute SHOULD<12> have the "existing" value only. The **a=connection:new** attribute SHOULD only be used with the **m=applicationsharing** media type, when the initial active media address is TCP-based.

The semantics of the "existing" value is specified in [RFC4145].

3.1.5.11 Text Telephony Support

A new media level attribute **a=tty** is defined. It is included by a user agent to indicate that it has been configured to optimize the transfer of tones used in text telephony. Such tones could, for example, originate from a **telecommunications device for the deaf (TDD)** – also referred to as a TTY (teletypewriter) – that interacts with the UC device via an audio coupler. The presence of this attribute can be used by the peer user agent as an indication to enable detection for such tones, and optimize for their transfer once detected.

Note that this attribute has no offerer/answer behavior. It is used to inform the peer user agent that a TTY device can be used.

3.1.5.12 Early Media Support

Early media refers to media exchange taking place before a session (3) INVITE is accepted. This could be the initial greeting received by the user while the SIP handshake is under way. Early media support amounts to getting an SDP answer in a provisional SIP response of the 18x levels and starting media exchange after processing the SDP answer. Provisional responses are specified in detail in [RFC3261] section 13. Early media support discussed in this document is not based on any specific standards protocol. It is the subject of the following sections.

3.1.5.12.1 Restriction to Receiving an SDP Answer in Provisional Response

To support early media, all of the following conditions MUST be met when a user agent receives an SDP answer in the provisional response.

- ICE MUST be supported for early media.
- All SDP answers in the provisional responses MUST be the same.
- When the offer is forked, SDP answers not in reliable provisional responses, as specified in section 3.1.5.13, SHOULD<13> be sent by a maximum of one device. For information about how to determine if an offer is forked, see [MS-SIPRE].

3.1.5.12.2 Receiving an SDP Answer in Provisional Response and Starting Media Streams

Media streams are started after receiving an SDP answer in the provisional response, depending upon whether the SIP INVITE request was forked to **multiple points of presence (MPOP)**.

A SIP INVITE request was forked if an **ms-forking** SIP header exists in any provisional response. The **ms-forking** header is added when the call is forked to MPOP by the SIP proxy or **server (2)**. For a more detailed specification of this header, see [MS-SIPRE].

Depending on whether the SIP INVITE request was forked, media streams are started as follows:

- If an SDP answer is received in a provisional response and the SIP INVITE request was forked, the following are applicable:
  - The received streams SHOULD<14> be started, if they are not already started.
  - The send stream SHOULD<15> be started for sending DTMF only after receiving one or more RTP media packets via the corresponding receive stream.
- If an SDP answer is received in a provisional response and the SIP INVITE request was not forked, both the receive and send streams for sending DTMF only SHOULD<16> be started with the consideration that the send stream is started only after receiving one or more RTP media packets via the corresponding receive stream.

Additionally, speech for the media streams in the forward direction SHOULD NOT<17> be started until the 200 OK is received for the INVITE.

3.1.5.12.3 SDP Answer in Provisional and Final Responses

SDP answers received in the provisional response of the 18x-level and the final response of the 200-level can be different depending on whether the call is forked. Specifically, if the 18x arrived from one fork and the 200-level from another fork, the SDP answers received can be different.

If an answer was contained in an 18x-level, it SHOULD<18><19> be repeated, without any changes, in the 200 for the same fork.

If the call is not forked, the SDP answer received in the final 200 response SHOULD<20><21> be the same as the one received in the provisional 18x response.

However, a user agent which supports early media SHOULD accept an SDP answer in the final 200 response which differs, in any of the following ways, from the SDP answer in the provisional 18x response:

1. The stream direction attribute changes—for example, from **a=sendrecv** to **a=recvonly**—as long the stream direction in the final 200 response is compatible with the offer (according to the rules of [RFC3264] section 6.1).

2. The key value in an **a=crypto** or **a=cryptoscale** attribute changes.

3. A media stream, which was accepted in the SDP answer in the provisional response, is rejected in the final response. For example: the offer included both m=audio and m=video media streams, both streams were accepted in the provisional response, but the m=video stream was then rejected in the final response.

For these cases, the user agent processing the final response SHOULD make effective the differences in the SDP answer from the provisional response.

3.1.5.12.4 ICE Processing When an SDP Answer is Received in the Provisional Response

When a SIP INVITE request is NOT forked and an SDP answer is received in the provisional response, ICE processing SHOULD<22> proceed as if the SDP answer was received in the final response.

3.1.5.13 Extensions for reliable provisional response processing and related offer/answer models

[RFC3262] specifies a means for SIP entities to send reliable provisional response within an early or established **dialog**. The following sections define client behavior and considerations specific to reliable provisional response and early media.

When negotiating early offer/answer prior to the call being answered, SIP user agents SHOULD<23> use the procedures specified in [RFC3262], with the following exceptions:

- A SIP user agent MUST NOT send any SIP request containing a **Require** header with the **option** tag of "100rel".
- A SIP user agent SHOULD include **Require:100rel** in 183 responses. SIP user agents SHOULD use a reliable provisional 183 response containing an SDP answer to perform early connectivity checks or to negotiate early media.

Furthermore, SIP user agents SHOULD<24> use the procedures specified in [IETFDRAFT-OFFANS-08] when sending reliable provisional response with SDP, with the following exceptions:

- A SIP user agent MUST NOT negotiate more than one offer/answer before the call is answered.
- A SIP user agent MUST NOT include an SDP offer or SDP answer in a provisional response acknowledgement (PRACK, as defined in [RFC3262]) message or a 200 OK response to a PRACK message.

- A SIP user agent MUST use a 1XX reliable response when responding to an INVITE with early media.
- A SIP user agent MUST use a 2XX response when responding to an INVITE of an established dialog.

When dealing with forked endpoints (5) and early media, SIP user agents SHOULD<25> also process 199 response code specified in [IETFDRAFT-RCITD-199-01] to clean up early media state, if any. Information regarding when a 199 is sent is specified in [MS-SIPRE].

3.1.5.14 No Support for Renegotiation of SRTP or SSRTP Encryption Parameters

SRTP encryption parameters MUST NOT be renegotiated after the encryption is negotiated and the session (3) is established.

SSRTP encryption parameters MUST NOT be renegotiated after the encryption is negotiated and the session (3) is established.

3.1.5.15 Labeling a Media Description with an a=label Attribute

All active media descriptions, except those with **m=applicationsharing** media type, in an SDP offer or SDP answer SHOULD contain an **a=label** attribute. The syntax of the **a=label** attribute is defined in [RFC4574].

The **a=label** attribute helps identify the modality of the media stream. In particular, the attribute can distinguish the modality of a media description with type **m=video** as either main video or panoramic video.

For the audio modality, the label value MUST be "main-audio". For the main video modality (which might comprise multiple media channels), the label value MUST be "main-video". For the panoramic video modality, the label value MUST be "panoramic-video". Other label values are not recognized and MUST be ignored by the receiving user agent.

3.1.5.16 Address types in the c= line

The IP address type specified in a c= line of an SDP message MUST be either **IPv4** or IPv6.<26>

An IPv4 address SHOULD be used, if the user agent generating the SDP message has a choice of available IPv4 and IPv6 addresses.

3.1.5.17 No Support for Optional Parameters in the a=rtcp Attribute

As specified in [RFC3605], the **a=rtcp** attribute has the following format in ABNF notation, as defined in [RFC5234]:

```
rtcp-attribute =  "a=rtcp:" port  [nettype space addrtype space
                  connection-address] CRLF
```

Optional parameters are allowed in addition to the *port* parameter. However, this protocol only supports the use of the *port* parameter in the **a=rtcp** attribute and stipulates that the optional parameters after the *port* parameter MUST NOT be used.

3.1.5.18 Application sharing media stream/type m=applicationsharing

This protocol defines a new media type, **applicationsharing**, which represents an RDP-based media stream/session (3). An application sharing **m=** line identifies exactly one RDP session (3).

Application sharing media requires a lossless transport and therefore the only candidates supported are TCP-based, as specified in [RFC4145] and [IETFDRAFT-ICETCP-07].

Application sharing does not support early media.

In the context of this media type four new SDP attributes are defined.

3.1.5.18.1 a=x-applicationsharing-session-id attribute

The **session-id** attribute is used to uniquely identify an **RDP** session (3) on one end.

This attribute is optional; if missing, a viewer is going to be connected to the first available session.

Session-id has the following format in ABNF notation, as defined in [RFC5234]:

```
session-id-attribute = "a=x-applicationsharing-session-id:" *(alphanum) CRLF
```

3.1.5.18.2 a=x-applicationsharing-role attribute

The **a=x-applicationsharing-role** attribute determines the RDP sharing role of the party.

This attribute SHOULD<27> be present. The party that is sharing SHOULD<28> set the role to "sharer" and the party that is viewing SHOULD<29> set the role to "viewer." The following table lists the role for the answer based on the role in the offer.

Offer	Answer
a=x-applicationsharing-role:sharer	a=x-applicationsharing-role:viewer
a=x-applicationsharing-role:viewer	a=x-applicationsharing-role:sharer

If the SDP session (3) contains multiple application sharing **m=** lines, the (**session-id, role**) pair SHOULD<30> be unique for each active **m=** line.

The RDP **role-attribute** has the following format in ABNF notation, as defined in [RFC5234]:

```
role-attribute = "a=x-applicationsharing-role:" ( "sharer" | "viewer" ) CRLF
```

3.1.5.18.3 a=x-applicationsharing-media-type attribute

This attribute is used to negotiate the RDP media type to be used. This attribute is optional; the default is "", indicating that this is a signaling-only session (3) with no associated media stream. The value "rdp" indicates that RDP media stream can be established. The **rdp-media-type-attribute** has the following format in ABNF notation, as defined in [RFC5234]:

```
rdp-media-type-attribute = "a=x-applicationsharing-media-type:" <list-of-supported-medias>
CRLF
<list-of-supported-medias>: <rdp-flavor> *( SPACE <rdp-flavor> )
<rdp-flavor>: "rdp" | ""
```

```
SPACE: %d32
```

3.1.5.18.4 a=mid attribute

The **a=mid** attribute is used as an identifier of the media described by the **m=** line. This attribute SHOULD<31> be included.

Every time a new **m=** line media is added to the SDP message, the value of **a=mid** is incremented by 1.

The **media-identifier-attribute** has the following format in ABNF notation, as defined in [RFC5234]:

```
media-identifier-attribute="a=mid:" 1*DIGIT CRLF
```

3.1.5.19 Interpretation of o= line in the SDP

The **o=** line of an SDP message, as specified in [RFC4566], specifies the session (3) originator and session (3) identifiers that include the session ID, session version, network type, address type, and unicast address. ABNF notation, as defined in [RFC5234], of the **o=** line is as follows:

```
O=<username> <sess-id> <sess-version> <nettype> <addrtype> <unicast-address>
```

The *sess-id* parameter MUST be ignored on receive. The *sess-version* parameter MUST be a numeric value but the value MUST be ignored on receive. The *sess-version* parameter is not incremented in subsequent SDP offers or SDP answers<32>. The *nettype* parameter MUST be "IN". The *addrtype* parameter MUST be either "IP4" or "IP6"<33>. If the addrtype is "IP4" the *unicast-address* parameter MUST be the dotted-decimal representation of the IP version 4 address. If the addrtype is "IP6" the *unicast-address* parameter MUST be the textual representation of the IP version 6 address.

3.1.5.20 Deviations from ICE-06

ICE, as specified in [IETFDRAFT-ICENAT-06], is a methodology to let media traverse NAT and firewalls to reach the remote peer. The following subsections describe the deviations from the standard ICE specification.

3.1.5.20.1 General Outline of the ICE Methodology

In general, ICE works as follows:

1. A peer, or offerer, gets all its addresses at which it may be reached and provides them in an SDP offer.
2. The SDP offer is sent to the remote peer.
3. The remote peer gets all its addresses at which it may be reached and provides them in an SDP answer.
4. On receiving the SDP offer, both the offerer and the answerer begin to exchange packets to determine the optimal path for media traversal. This process of determining the optimal path is referred to as connectivity checks in the subsequent discussions.

5. After this optimal path is determined, the offerer sends a SIP re-INVITE to the remote peer, communicating the optimal address in the SDP offer. This SIP re-INVITE is referred to as an ICE re-INVITE in the subsequent sections of this document. An indicator of the ICE re-INVITE is the existence of an **a=remote-candidate** attribute for a modality. This attribute is absent in the previous SIP INVITE or SIP re-INVITE. For more details, see [IETFDRAFT-ICENAT-06].

3.1.5.20.2 ICE RE-INVITE Initiator

According to [IETFDRAFT-ICENAT-06], an ICE re-INVITE is always sent by the offerer of that media. This protocol deviates from that specification, and stipulates that the ICE re-INVITE MUST always be sent by the offerer of the call and not the offerer of the modality. This means that the caller MUST send the ICE re-INVITE.

This also means that if the local peer starts an audio call with a remote peer and then, after some time, the remote peer adds video to this call, the ICE re-INVITE for the video stream MUST be sent by the local peer, instead of by the remote peer. In contrast, the [IETFDRAFT-ICENAT-06] specification requires that the ICE re-INVITE for video is sent by the remote peer in a similar case.

3.1.5.20.3 No Update of Candidates Between INVITE and ICE RE-INVITE

According to [IETFDRAFT-ICENAT-06], the list of addresses exchanged in the original SIP INVITE can be updated anytime between the first INVITE and the ICE re-INVITE by sending a SIP **UPDATE** or SIP re-INVITE request. However, this protocol stipulates that an application MUST NOT add or remove addresses using SIP UPDATE or SIP re-INVITE until the connectivity checks have finished or until an ICE re-INVITE is exchanged successfully.

3.1.5.20.4 Extending the Transport to Connection-Oriented (TCP)

ICE, as specified in [IETFDRAFT-ICENAT-06], specifies that UDP be used as the transport and allows extensions to add other transport. This protocol adds TCP to the supported transport type in ICE. Thus, the following examples are both permitted:

```
a=candidate:ir84fUlcDqYH50bs2M/Xn/pDNE+fVfxRTbXBWG34PM8 2 1vvq9h3j8xixI3npD0X9VA UDP 0.830
10.56.65.184 63616
a=candidate:MbmhbdY6gJ1nwkKtoJWa8h9LHlpQ90uT/EiBDOvBPB4 1 76CTu2GXyKtnYlu2ZydjXA TCP 0.190
172.29.105.45 50563
```

3.1.5.20.5 No IPv6 transport addresses

The transport address given in an **a=candidate** attribute MUST be an IPv4 address.

3.1.5.21 Deviation from ICE V19

ICE, as specified in [IETFDRAFT-ICENAT-19] , is a methodology to let media traverse NAT and firewalls to reach the remote peer. Support of ICE in this protocol differs from that specified in that document. The following subsections describe deviations from the standard ICE specification.

3.1.5.21.1 Support for IPv6 transport addresses

According to [IETFDRAFT-ICENAT-19] section 5.1, an ICE **a=candidate** attribute contains two fields which transmit an IP address: **connection-address**, and the optional **rel-addr**. Although the grammar permits these fields to contain an IPv6 address, a non-IPv6-aware user agent might malfunction parsing such an **a=candidate** attribute. This section describes an SDP extension for

offering IPv6 ICE candidates in a way to avoid interop problems with non-IPv6-aware peer user agents.

A user agent SHOULD<34> be able to parse successfully an **a=candidate** attribute containing an IPv6 address in the connection-address and/or rel-addr fields.

If the receiving user agent does not support IPv6, it SHOULD ignore an **a=candidate** attribute containing an IPv6 connection-address. However, the non-IPv6-capable user agent MUST accept an **a=candidate** attribute containing an IPv4 connection-address field with an IPv6 rel-addr. The rel-addr field is not used in the ICE protocol itself, but is for informational purposes only, and so MUST be allowed by the receiving user agent.

If the user agent is sending an SDP offer or SDP answer and has ICE candidates with an IPv6 connection-address, then there is a concern whether the peer user agent will parse the SDP message properly. If the user agent does not know whether its peer is capable of parsing an IPv6 connection-address, it SHOULD use a new **a=x-candidate-ipv6** attribute (defined in the next section) to transmit the ICE candidate.

On the other hand, if the user agent does know that its peer is IPv6-capable then it SHOULD use the standard **a=candidate** attribute to transmit an ICE candidate. A user agent can discover that its peer is IPv6-capable if a previous SDP offer or SDP answer received from the peer included an ICE candidate containing an IPv6 addresses (in either an **a=candidate** attribute or **a=x-candidate-ipv6** attribute).

3.1.5.21.1.1 a=x-candidate-ipv6 attribute

The syntax of this attribute is identical, other than the attribute's name (**x-candidate-ipv6**), to the **a=candidate** attribute defined in [IETFDRAFT-ICENAT-19] section 5.1, with one additional requirement: the connection-address field MUST be an IPv6 address. If the connection-address type is not IPv6, the **a=x-candidate-ipv6** attribute MUST be rejected as syntactically incorrect.

Like **a=candidate**, the **a=x-candidate-ipv6** attribute is a media-level SDP attribute only.

The ICE candidate given in an **a=x-candidate-ipv6** attribute (consisting of foundation, transport, connection-address, port and candidate-type token values) MUST NOT duplicate an ICE candidate given in an **a=candidate** attribute.

3.1.5.21.2 LITE implementation

According to [IETFDRAFT-ICENAT-19], there are two implementations of ICE – FULL implementation and LITE implementation. This protocol does not support the LITE implementation. This means that this protocol SHOULD<35> gather the **Relayed Candidates** and **Server Reflexive Candidates** and perform connectivity checks as specified in [MS-ICE2].

3.1.5.21.3 Ice-options attributes

According to [IETFDRAFT-ICENAT-19], an offer or an answer is allowed to use the **ice-options** attribute to identify the ICE extensions supported by that agent. If an agent supports an extension, it includes the token that represents that extension in the **ice-options** attributes.

This protocol does not support the **ice-options** attribute. It SHOULD NOT<36> generate an SDP message with this attribute and SHOULD ignore this attribute if it is present in the SDP message received.

3.1.5.21.4 Ice-mismatch attributes

According to [IETFDRAFT-ICENAT-19], this attribute, when present in an answer, indicates that the agent that sends the offer contains a default destination for a media component that did not have a corresponding candidate attribute.

This protocol does not support this attribute. It SHOULD NOT<37> generate an answer with this attribute. If received as an answer to an offer, this protocol SHOULD ignore this attribute.

For offers that are generated by this protocol, the default destination for a media component SHOULD<38> have a corresponding candidate attribute.

3.1.5.21.5 ice-ufrag and ice-pwd attributes

According to [IETFDRAFT-ICENAT-19], the **ice-ufrag** attribute can be 4 to 256 bytes long and the **ice-pwd** attribute can be 22 to 256 bytes long, and they are in plaintext. This protocol determines if **base64 encoding** is used in the offer by checking their lengths. Therefore, so that the answering agent does not treat plaintext as an encoded string, this protocol SHOULD NOT<39> use an **ice-ufrag** attribute of 6 bytes and an **ice-pwd** attribute of 32 bytes long in an offer.

3.1.5.22 Deviation from ICE-TCP-07

ICE, as specified in [IETFDRAFT-ICENAT-06] and [IETFDRAFT-ICENAT-19], defines ways for media traffic to traverse NAT and a firewall. These specifications provide a general framework for describing candidates, which only use the UDP transport protocol.

[IETFDRAFT-ICETCP-07] extends the ICE protocol to include TCP transport protocol.

The deviations from these specifications are described in the subsections that follow.

3.1.5.22.1 Default Candidate

For audio and video calls, the default candidate SHOULD NOT<40> be TCP. For application sharing calls, the default candidate SHOULD<41> be TCP.

3.1.5.22.2 Local Candidate

For audio and video media type, this protocol does not gather passive local **Host Candidates** for TCP. Therefore the SDP message SHOULD NOT have any passive TCP local host candidates.

For application sharing media type, the local candidates SHOULD<42> be TCP.

3.1.5.23 Extensions for call hold and retrieve

The following specifies client behavior for the offer and answer negotiated for hold and un-hold operations when in an audio or video call.

3.1.5.23.1 Invoking hold

A protocol client invoking hold is required to do the following for all audio and video media streams in the resulting offer:

- The client SHOULD<43> change the direction of all streams to "inactive".
- The client SHOULD<44> include **sip.rendering**, as specified in [MS-SIPRE], with a value of "no".

3.1.5.23.2 Clearing hold (retrieve)

In-order to clear the hold, or retrieve the call, the protocol client is required to do the following for all audio and video media streams in the resulting offer:

- The client SHOULD<45> change the direction of all streams to "sendrecv".
- The client SHOULD<46> exclude **sip.rendering**, as specified in [MS-SIPRE].

3.1.5.24 Extension for video receive capabilities a=x-caps

This protocol defines a video media level attribute **a=x-caps**, which represents what a video receiver is capable of receiving. A video capability **a=** line defines video capabilities for each of the video codec the video receiver is capable of receiving.<47>

This attribute is optional; if missing, a video sender SHOULD set the video receive capabilities of the remote peer as **Common Intermediate Format (CIF)** at 15 fps and Video Graphics Array (VGA) at 15 fps. Quarter CIF (QCIF, which represents a video resolution of 176 width by 144 height), CIF and VGA MUST be advertised in the list of Video capabilities for a media stream that has an **a=label:main-video** SDP attribute. A video capability of VGA 13 fps MUST be treated as VGA 15 fps. High-definition (HD), which represents a video resolution of 1280 width by 720 height, is an additional video capability that can be advertised by a video receiver. This media attribute has the following format in ABNF notation, as defined in [RFC5234]:

Video-Capabilities-media-type-attribute:

```
"a=x-caps:" <video-payload-type > SPACE <list-of-video-capabilities> CRLF
```

video-payload-type: The RTP payload type number for video, such as 121 for x-rtvc1 and 34 for H.263.

list-of-video-capabilities:

```
<video-capability>";"<video-capability>
```

video-capability:

```
<Capability-ID>":"<Width-of-video-frame>":"<Height-of-video-frame>":"<frames-per-
second>":"<maximum-bitrate-bits-per-second>":"<additional-attributes>
```

Capability-ID (integer): A unique random integer among the listed capability ID for that **m=** line. The value is between 1 and 2147483647 in the entire **video-capabilities-media-type-attribute**.

Width-of-video-frame (integer): The width is one of these values:

- 176 for QCIF
- 352 for CIF
- 640 for VGA

- 1280 for HD

Height-of-the-video-frame (integer): The height is one of these values:

- 144 for QCIF
- 288 for CIF
- 480 for VGA
- 720 for HD

frames-per-second (float): Value SHOULD be less than or equal to 30.0 fps. Any values beyond 30.0 SHOULD be treated as 30.0. It specifies the maximum frame rate the receiver is capable of receiving.

maximum-bitrate-in bits-per-second (integer): Ignored and reserved for future use.

Any additional attribute SHOULD be ignored and reserved for future use.

A protocol peer, upon receiving the video capabilities SHOULD<48> do the following:

- If there is a "," in a **video-capability** attribute, anything from "," till the end of **video-capability** (";" or CRLF) SHOULD be ignored.
- If there is a syntax error in **a=x-caps**, the whole **a=x-caps** SHOULD be ignored and video receive capabilities of the remote peer SHOULD be set as CIF at 15 fps or VGA at 15 fps.

The following is an example **a=x-caps** attribute:

```
a=x-caps:121
263:1280:720:30.0:1500000:1;4359:640:480:30.0:600000:1;8455:352:288:15.0:250000:1;12551:176:1
44:15.0:180000:1
```

The **a=x-caps** attribute is not supported for the H.264UC or ULPFEC-UC video media formats. An **a=x-caps** attribute SHOULD NOT be included for H.264UC or ULPFEC-UC. An **a=x-caps** attribute for H.264UC or ULPFEC-UC, if present in a received SDP message, MUST be ignored.

3.1.5.25 Extensions to optimize the media path to a gateway

This section describes the extensions used by the client to optimize the media path to a gateway in the same location. These extensions SHOULD be used by **SIP protocol clients** that support **ms-bypass**, as specified in [MS-OCPSTN]. <49>

3.1.5.25.1 a=x-bypassid attribute

The **a=x-bypassid** attribute is a declarative attribute used to indicate the location of the media endpoint (5) associated with an SDP offer. It is a media level attribute that MUST be sent in an offer by the client to establish a baseline for the possibility of doing media bypass.

When the SIP protocol client receives a multipart/alternative MIME body in the offer, it first looks for a part of type **application/GW-SDP**. If one is found and the **x-bypassid** values match, that part is chosen.

3.1.5.25.2 a=x-bypass attribute

The **a=x-bypass** attribute is a declarative attribute that signifies that the media line with which it is associated involves bypass. It is a media level attribute that MUST be sent when the answerer has chosen the bypass path.

3.1.5.25.3 a=x-mediasettings attribute

The **a=x-mediasettings** attribute SHOULD be added by a user agent to signify the following stream capabilities:

- **holdrtcpunsupported**: SHOULD be added by a user agent to signify that RTCP is not supported when the call is on hold. If present in the negotiated SDP, the client MUST NOT expect RTCP when the call is on hold.
- **rtcpunsupported**: SHOULD be added by a user agent to signify that RTCP is not supported. If present in the negotiated SDP, the client MUST NOT expect RTCP for the call.
- **signalboostunsupported**: SHOULD be added by a user agent to signify that its media stream is not amplified. If present in the negotiated SDP, the client SHOULD apply amplification on the incoming media stream.

The grammar for this attribute is defined as follows.

```
a=x-mediasettings:(holdrtcpunsupported/rtcpunsupported/signalboostunsupported)*(SPACE
holdrtcpunsupported/rtcpunsupported/signalboostunsupported)
```

3.1.5.26 Extensions for diagnostic info in SDP messages

This protocol defines a new media level attribute **a=x-ms-SDP-diagnostics**<50>. An SDP endpoint (5) SHOULD add this attribute if it requires the receiving endpoint (5) to display a notification regarding the status of the SDP session (3).

The format for the **a=x-ms-SDP-diagnostics** in ABNF, as defined in [RFC5234], is as follows. It is similar to that of the **ms-diagnostics-public header** defined in [MS-OCER].

The parameters *EQUAL*, *HCOLON*, *SEMI*, *generic-param*, and *quoted-string* are as defined in [RFC3261] section 25.1.

```
a EQUAL x-ms-SDP-diagnostics HCOLON  ErrorId  *(.SubErrorId) SEMI reason-param * (SEMI
generic-param)
ErrorId = unsigned-integer

SubErrorId = unsigned-integer

reason-param = "reason=" reason-value

reason-value = quoted-string
```

ErrorId (unsigned-integer): Required. Value MUST be within unsigned 32-bit integer range. Represents a specific error condition, and SHOULD be used by the SIP protocol client to determine error handling behavior.

SubErrorId (unsigned-integer): Optional. If present, its value MUST be within the unsigned 32-bit integer range. **SubErrorId** can be used to differentiate related scenarios that result in the same **ErrorId**, and can be used by the SIP client to determine error handling behavior.

reason-value: Optional. The reason SHOULD indicate an explanation of the error. A SIP client SHOULD NOT use this parameter value to determine error handling behavior. This parameter value can be used for SIP server (2) troubleshooting purposes.

***(SEMI generic-param):** Optional. Can be used to define custom attribute-value pairs, to convey additional troubleshooting information to the SIP client.

The following table lists the allowed values. More values might be added in the future. If an **ErrorId** not listed here is received by an SDP endpoint (5), it SHOULD be ignored.

ErrorId	Reason string	Explanation
53000	Insufficient Bandwidth Available	Bandwidth policy checks on server (2) failed for this particular m= line.
53001	Candidates Restricted	Bandwidth policy checks required some of the original a=candidate lines to be removed for bandwidth limitation reasons.

Following example SDP answer contains an **a=x-ms-sdp-diagnostics** attribute in the media description for **m=video**:

```
v=0
o=- 168 0 IN IP4 172.18.0.106
s=session
c=IN IP4 172.18.0.106
b=CT:1000
t=0 0
m=audio 51038 RTP/SAVP 9 111 0 8 97 101 13 118
c=IN IP4 172.18.0.106
a=rtcp:51039
a=ice-ufrag:VUa7
a=ice-pwd:uSvOqE8rrlf2065N/AymKLpL
a=candidate:1 1 UDP 2130706431 172.18.0.106 51038 typ host
a=candidate:1 2 UDP 2130705918 172.18.0.106 51039 typ host
a=candidate:3 1 tcp-act 1684798719 172.18.0.106 50112 typ srflx raddr 172.18.0.106 rport
50112
a=candidate:3 2 tcp-act 1684798206 172.18.0.106 50112 typ srflx raddr 172.18.0.106 rport
50112
a=label:main-audio
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:8q4vdHtbV3uIGM7z+jgLTxltWIhd9vedIMXiO4MB|2^31|1:1
a=rtpmap:9 g722/8000
a=fmtp:9 bitrate=64000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 RED/8000
a=fmtp:97 red/8000
a=rtpmap:101 telephone-event/8000
```

```
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=ptime:20
m=video 0 RTP/AVP 121
a=x-ms-sdp-diagnostics:53000; reason="Insufficient Bandwidth Available"
a=label:main-video
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;VGA=15;PANO=15
```

3.1.5.27 Extensions for Music-on-Hold

This section specifies SDP extensions that a user agent can use in SDP offers to indicate that music-on-hold is being streamed in a given media session (3).

3.1.5.27.1 a=feature attribute

The **a=feature** attribute is a declarative media-level attribute that specifies additional features for its associated media line. Its syntax is as follows:

```
a EQUAL feature HCOLON 1*alphanum
```

The parameters *EQUAL*, *HCOLON*, and *alphanum* are as defined in [RFC3261] section 25.1. The alphanumeric string to the right of the colon indicates the particular feature being attributed to the associated media. For music-on-hold, it MUST be "MoH". Additional values might be defined in the future to signal other features besides music-on-hold.

3.1.5.27.2 User agent behavior for a=feature attribute

If a user agent wishes to use these extensions to signal that it is streaming music-on-hold, its offer MUST contain **a=sendonly** and **a=feature:MoH** attribute lines for those media. The **a=feature:MoH** line MUST NOT appear under any other media line than **m=audio**.

If all media lines in the SDP offer contain an **a=feature:MoH** attribute line, the SIP **Contact** header SHOULD include **sip.rendering**, specified in [RFC4235] section 5.2, with a value of "no".

When a user agent receives an SDP offer with **a=feature:MoH**, it can choose to render a user interface for hold or music-on-hold. When a user agent receives an SDP offer with features that it does not understand, it SHOULD ignore them.

3.1.5.28 Extensions for media bandwidth

This section specifies an SDP extension to allow a user agent to declare in an SDP offer or SDP answer what bandwidth it has available to send and receive media for a particular modality.

This extension is useful when, for example, a video modality is supported with a codec which supports a multitude of frame rate and resolution combinations. The user agent can declare the bandwidth it has available for its receive stream, which the peer can use to configure the video codec on its send stream such that its output bandwidth consumption does not exceed the receive stream bandwidth limit declared in the **a=x-mediabw** attribute.

3.1.5.28.1 a=x-mediabw attribute

The **a=x-mediabw** attribute is a declarative session-level attribute that specifies the send and receive bandwidth limits for a particular modality. These values are given from the perspective of the user agent which created the SDP message containing the attribute.

The syntax of the attribute is:

```
"a=x-mediabw:" label SPACE "send=" bandwidth ";recv=" bandwidth

label = token

bandwidth = unsigned-integer
```

The bandwidth value MUST be a decimal integer in the range 0 to 4294967295 inclusive. The bandwidth value is interpreted in kilobits-per-second (1000 Kbps). For example, a bandwidth value of 1500 represents 1,500,000 bits per second. A value of 0 means no bandwidth is available for that stream direction.

The value following the "send=" token is the send stream bandwidth limit; the value following the "recv=" token is the receive stream bandwidth limit.

The syntax of the label token is as defined in [RFC4566] section 9.

The x-mediabw attribute is a session-level only attribute. It MUST NOT be present within a media description. If present within a media description, it MUST be ignored by the receiving user agent.

There MUST NOT be more than one session-level x-mediabw attribute specifying the same label value. If there are multiple x-mediabw attributes with the same label value, the receiving user agent will pick an arbitrary attribute and ignore the others.

The x-mediabw attribute(s) of an SDP message MUST NOT be modified in a subsequent renegotiation. If the bandwidth values change in a renegotiation, the new values MUST be ignored. If a renegotiation introduces a new modality, the SDP offer SHOULD<51> add a new x-mediabw attribute for it.

3.1.5.28.2 User agent behavior for a=x-mediabw attribute

For the bandwidth limits to be effective, the label value of the x-mediabw attribute MUST match the media-level **a=label** attribute of one or more media descriptions within the SDP message, and the label value MUST be recognized by the user agent receiving the SDP message.

For the audio modality, the label value MUST be "main-audio". For the main video modality (which might comprise multiple media channels), the label value MUST be "main-video". For the panoramic video modality, the label value MUST be "panoramic-video". If the label value is not recognized by the receiving user agent, the attribute MUST be ignored.

If the label value is recognized by the receiving user agent, and one or more media streams with that label exist within the media session, the bandwidth values in the attribute SHOULD<52><53> be applied. If there is more than one active media channel with the same label value (for example, "main-video"), the bandwidth limits apply to the total bandwidth consumed by all the media channels of that modality. The receiving user agent applies the receive bandwidth limit from the x-mediabw attribute as an upper bandwidth consumption limit to its send stream(s). (The bandwidth consumed by the user agent's send stream might also be constrained by other factors, unrelated to the x-mediabw attribute.)

3.1.5.29 Extensions for declaring device capabilities

This section specifies an SDP extension to allow a user agent to declare in an SDP offer or SDP answer its device capabilities, that is, what media types it is capable of sending and/or receiving and rendering within its user interface. It is assumed that indicating support for receiving implies the user agent is capable of rendering such media within its user interface.

This extension allows a user agent to inform its peer that it is capable, for example, of sending and receiving video media, for example, because its endpoint (5) has a connected web camera device.

Declaring a device capability does not mean a user agent will negotiate a media stream for the corresponding modality, but only that it has the capability of sending/receiving/rendering a particular media type.

3.1.5.29.1 a=x-devicecaps attribute

The **a=x-devicecaps** attribute is a declarative session-level attribute that indicates the media types the user agent is capable of sending and/or receiving.

The syntax of the attribute is:

```
"a=x-devicecaps:" device-capability *[";" device-capability]

device-capability = device-type ":" capability-type *["," capability-type]

device-type = "audio" | "video" | "applicationsharing" | "data" | token

capability-type = "send" | "recv" | token
```

The syntax of token is as defined in [RFC4566] section 9. The presence of token within the grammar enables future extensibility.

There MUST NOT be duplicated capability-type token values for a device-capability. For example, **a=x-devicecaps:audio:send,send** is invalid.

There MUST be at most one device-capability declared for a given device-type. If there is more than one the receiving user agent will pick an arbitrary device-capability and ignore the others for the device-type.

The x-devicecaps attribute is a session-level only attribute. It MUST NOT be present within a media description. If present within a media description, it MUST be ignored by the receiving user agent.

The following example **a=x-devicecaps** attribute indicates the user agent has audio and video capture devices and can also render audio and video media within its user interface (for example, play audio through speakers and display video on a screen).

```
a=x-devicecaps:audio:send,recv;video:send,recv
```

3.1.5.29.2 User agent behavior for a=x-devicecaps attribute

A user agent SHOULD<54> indicate in any SDP message it sends what device capabilities it currently supports using the **a=x-devicecaps** attribute.

However, an MCU SHOULD NOT include any x-devicecaps attributes in SDP messages it sends, and SHOULD ignore the x-devicecaps attribute in any SDP messages it receives.

If the device-type is not recognized by the receiving user agent, the device-capability MUST be ignored.

If the device-type is recognized, the receiving user agent SHOULD indicate the peer's device capability within its user interface in some appropriate fashion. For example, for the video device-capability, the user agent may display a web camera icon, to suggest that a video modality can be engaged with the peer.

3.1.5.30 Extensions for RTCP-based feedback messages

This section specifies an SDP extension to allow a user agent to declare in an SDP offer or SDP answer the capability to send and receive certain RTCP-based feedback messages using a special Reduced-Size format.

3.1.5.30.1 a=rtcp-rsize attribute

The **a=rtcp-rsize** attribute is a declarative media-level attribute to indicate that all RTCP-based feedback messages (declared by the **a=rtcp-fb** media-level attribute) can be sent and received in a Reduced-Size format described specified in [MS-RTP] section 2.2.11.

The syntax of the attribute is:

```
"a=rtcp-rsize"
```

The **a=rtcp-rsize** attribute is a media-level only SDP attribute. An **a=rtcp-rsize** attribute present at the SDP session level MUST be ignored.

This attribute SHOULD NOT be included in a media description that also contains ICE **a=candidate** attributes as defined in [IETFDRAFT-ICENAT-06]. If the media description supports ICE as defined in [IETFDRAFT-ICENAT-06], the **a=rtcp-rsize** attribute MUST be ignored by the receiving user agent.

3.1.5.30.2 a=rtcp-fb attribute

The **a=rtcp-fb** attribute is a declarative media-level attribute to indicate what RTCP-based feedback messages can be sent by, and received from, the associated media stream.

This attribute extends the **a=rtcp-fb** RTCP feedback capability attribute defined in [RFC4585] section 4.2 with a custom "**x-message**" feedback type.

The syntax of the attribute is:

```
"a=rtcp-fb:" rtcp-fb-pt SPACE rtcp-fb-val

rtcp-fb-pt = "*" / fmt

fmt = integer

rtcp-fb-val = rtcp-fb-id [rtcp-fb-param]

rtcp-fb-id = "x-message" / token

rtcp-fb-param = SPACE "app" [SPACE rtcp-fb-app-param]
              / SPACE token [SPACE byte-string]

rtcp-fb-app-param = rtcp-fb-send-param [SPACE rtcp-fb-recv-param]
```

```
                    / rtcp-fb-recv-param [SPACE rtcp-fb-send-param]
rtcp-fb-send-param = "send:" rtcp-fb-sendrecv-caps

rtcp-fb-recv-param = "recv:" rtcp-fb-sendrecv-caps

rtcp-fb-sendrecv-caps = rtcp-fb-sendrecv-cap *("," rtcp-fb-sendrecv-cap)

rtcp-fb-sendrecv-cap = "dsh" / "src" / "x-pli" / token
```

The ABNF terms integer, token, byte-string are defined in [RFC4566].

An **a=rtcp-fb** attribute that conforms to this syntax but does not begin with `"* x-message app"` SHOULD be ignored.

The following briefly describes the meanings of the **rtcp-fb-sendrecv-cap** capability tokens:

- **dsh**: Dominant Speaker History Notification.
- **src**: Video Source Request.
- **x-pli**: Picture Loss Indicator.

For additional information about these capabilities, refer to [MS-RTP] section 2.2.11.

The following example **a=rtcp-fb** attribute declares send and receive Video Source Request and Picture Loss Indicator capabilities.

```
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
```

If an **a=rtcp-fb** attribute declares a send or receive capability that is not supported by the receiving user agent for the associated media stream, the capability MUST be ignored. For example, the **dsh** capability applies only to **m=audio** media descriptions and is ignored if declared for an **m=video** media description.

The **a=rtcp-fb** attribute is a media-level only SDP attribute. An **a=rtcp-fb** attribute present at the SDP session level MUST be ignored.

This attribute SHOULD NOT be included in a media description that also contains ICE **a=candidate** attributes as defined in [IETFDRAFT-ICENAT-06]. If the media description supports ICE as defined in [IETFDRAFT-ICENAT-06], the **a=rtcp-fb** attribute MUST be ignored by the receiving user agent.

3.1.5.30.3 User agent behavior for a=rtcp-rsize and a=rtcp-fb attributes

If a media description declares support for RTCP-based feedback messages using an **a=rtcp-fb** attribute, the media description MUST<55> also include the **a=rtcp-rsize** attribute.

3.1.5.31 Extensions for Synchronization Source (SSRC) range allocation

This section specifies an SDP extension to allow a user agent to declare in an SDP offer or SDP answer a range from which the user agent will allocate Synchronization Source (SSRC) values on a media channel send stream.

3.1.5.31.1 a=x-ssrc-range attribute

The **a=x-ssrc-range** attribute is a declarative media-level attribute which defines the range from which any SSRC values used on the send stream will be allocated. The range is inclusive.

The syntax of the attribute is:

```
"a=x-ssrc-range:" range-start "-" range-end

range-start = integer

range-end = integer
```

The ABNF term integer is as defined in [RFC4566].

The additional following constraints apply to the values of range-start and range-end:

1. range-start MUST be equal to or greater than 1;

2. range-end MUST be equal to or greater than range-start;

3. range-end MUST be less than or equal to 4294967040.

The SSRC range given by the **a=x-ssrc-range** attribute in a media description MUST NOT overlap the SSRC range defined for any other active media description located above it in the same SDP message. Otherwise, the media description SHOULD be rejected. That is, all active media channels in the SDP message with an **a=x-ssrc-range** attribute MUST have non-overlapping SSRC ranges. Two ranges `a=x-ssrc-range:A-B` and `a=x-ssrc-range:X-Y` overlap if any value ***z*** exists where X <= z <= Y and also A <= z <= B.

The **a=x-ssrc-range** attribute does not apply to the **m=applicationsharing** media type. An **m=applicationsharing** media description SHOULD NOT contain an **a=x-ssrc-range attribute**.

The **a=x-ssrc-range** attribute for an active media description MUST NOT change in a subsequent SDP offer or SDP answer.

This attribute SHOULD NOT be included in a media description which also contains ICE **a=candidate** attributes as defined in [IETFDRAFT-ICENAT-06]. If the media description supports ICE as defined in [IETFDRAFT-ICENAT-06], the **a=x-ssrc-range** attribute MUST be ignored by the receiving user agent.

3.1.5.31.2 User agent behavior for a=x-ssrc-range attribute

When allocating an **m=audio** or **m=video** media stream, a user agent SHOULD<56> allocate an appropriate SSRC range.

If a media description in a received SDP message contains an **a=x-ssrc-range** attribute, the receiving user agent SHOULD configure its receive stream to expect SSRC values in the range declared by the peer.

If the user agent is an MCU, the SSRC range allocated for a send stream of the media channel SHOULD NOT overlap the SSRC range allocated by the MCU for any other media channel within the same conference. Furthermore, the SSRC ranges allocated by an MCU SHOULD NOT overlap any Media Source ID (MSI) values it allocates within the **conference**.

The size of an SSRC range depends on the media type. A size of 1 (for example, **a=x-ssrc-range:1-1**) SHOULD be used for an **m=audio** media description. A range size of 100 (for example, **a=x-ssrc-range:101-200**) SHOULD be used for an **m=video** media description.

3.1.5.32 Extensions for Media Source ID (MSI) assignment

This section specifies an SDP extension to allow a user agent to declare in an SDP offer or SDP answer the Media Source ID (MSI). An MSI represents a **contributing source (CSRC)** and replaces the use of an SSRC value in a CSRC list. An MSI is allocated for a media stream by an MCU and given to a conference client using an **a=x-source-streamid** attribute. For additional information about MSI, refer to [MS-RTP] section 2.2.10.

3.1.5.32.1 a=x-source-streamid attribute

The **a=x-source-streamid** attribute is a declarative media-level attribute which assigns a Media Source ID (MSI) to a media stream.

The syntax of the attribute is:

```
"a=x-source-streamid:" media-source-id

media-source-id = integer
```

The ABNF term integer is as defined in [RFC4566].

The value of media-source-id MUST be greater than or equal to 1 and less than or equal to 4294967040.

The media-source-id value given by the **a=x-source-streamid** attribute in a media description MUST NOT duplicate the media-source-id value defined for any other active media description located above it in the same SDP message. That is, all active media channels in the SDP message with an **a=x-source-streamid** attribute MUST have unique media-source-id values.

The **a=x-source-streamid** attribute does not apply to the **m=applicationsharing** media type. An **m=applicationsharing** media description SHOULD NOT contain an **a=x-source-streamid** attribute.

The **a=x-source-streamid** attribute for an active media description MUST NOT change in a subsequent SDP offer or SDP answer.

This attribute SHOULD NOT be included in a media description which also contains ICE **a=candidate** attributes as defined in [IETFDRAFT-ICENAT-06]. If the media description supports ICE as defined in [IETFDRAFT-ICENAT-06], the **a=x-source-streamid** attribute MUST be ignored by the receiving user agent.

3.1.5.32.2 User agent behavior for a=x-source-streamid attribute

When allocating an **m=audio** or **m=video** media stream, an MCU user agent SHOULD<57> allocate an appropriate MSI and provide the media stream's MSI to the client using the **a=x-source-streamid** attribute. The MSI value allocated by the MCU MUST be unique within the conference.

An MCU user agent SHOULD ignore an **a=x-source-streamid** attribute in an SDP message received from a conference **participant (2)**.

3.1.5.33 Extensions for media source labeling

This section specifies an SDP extension to allow a user agent to declare in an SDP offer or SDP answer a descriptive name for a media source. If given to an MCU, this media source name can be communicated by the MCU to other participants within the conference, to be displayed within their user interface.

3.1.5.33.1 a=x-source attribute

The **a=x-source** attribute is a declarative media-level attribute that assigns a descriptive name to a media stream.

The syntax of the attribute is:

```
"a=x-source:" media-source-name

media-source-name = text
```

The ABNF term text is as defined in [RFC4566].

The media-source-name value given by the **a=x-source** attribute in a media description SHOULD NOT duplicate the media-source-name value defined for any other active media description located above it in the same SDP message. That is, all active media channels in the SDP message with an **a=x-source** attribute SHOULD have unique media-source-name values.

The **a=x-source** attribute does not apply to the **m=applicationsharing** media type. An **m=applicationsharing** media description SHOULD NOT contain an **a=x-source** attribute.

This attribute SHOULD NOT be included in a media description which also contains ICE **a=candidate** attributes as defined in [IETFDRAFT-ICENAT-06]. If the media description supports ICE as defined in [IETFDRAFT-ICENAT-06], the **a=x-source** attribute MUST be ignored by the receiving user agent.

3.1.5.33.2 User agent behavior of a=x-source attribute

A user agent SHOULD<58> include an **a=x-source** attribute in an **m=audio** or **m=video** media description if media from an attached media source device will be sent on the media stream. For example, if the media stream will be sending media from an audio input device or video camera, an **a=x-source** attribute can give a descriptive name for the media source, such as "main camera".

An MCU user agent SHOULD NOT include the **a=x-source** attribute in its media descriptions.

3.1.5.34 Extensions for multiplexed media channels

This section describes an SDP extension to negotiate multiplexed media **streams (2)**.

In terms on SDP negotiation, the multiplexed media streams all share the same set of transport addresses (as well as other attributes). This section describes how a set of media descriptions within an SDP message indicate they are to be multiplexed, and what requirements must be met<59>.

3.1.5.34.1 Indicating multiplexed media channels in an SDP message

A basic requirement is that all multiplexed media streams MUST support ICE as specified in [IETFDRAFT-ICENAT-19].

A set of media descriptions indicate their media streams are to be multiplexed by meeting the following requirements:

1. The media types, as specified by the **m=** field, MUST be equal. Multiplexing media streams of different media types is not supported.
2. The connection addresses, as specified by the **c=** field, MUST be equal.
3. The RTP port values, as specified by the **m=** field, MUST be equal.
4. The RTCP port values, as either specified by the **a=rtcp** attribute or inferred by the RTP port value, MUST be equal.
5. The transport protocols, as specified by the **m=** field, MUST be equal.
6. The ICE **a=ice-ufrag** attributes MUST be equal.
7. The ICE **a=ice-pwd** attributes MUST be equal.
8. All the media descriptions in the multiplexed set, except the first one, MUST NOT contain any **a=candidate**, **a=x-candidate-ipv6** or **a=remote-candidates** attributes. The first media description in the set MUST specify a=candidate/a=x-candidate-ipv6 attributes for ICE (and the a=remote-candidates attribute if ICE has completed).
9. A valid **a=x-ssrc-range** attribute MUST be given. The SSRC range MUST NOT overlap the SSRC range of any other media description within the SDP message.
10. An **a=label** attribute MUST be present and all the media descriptions in the multiplexed set MUST specify the same label value.

3.1.5.34.2 User agent behavior for negotiating multiplexed media channels

If a user agent wishes to include multiplexed media streams in an SDP offer, it must take care in how it forms the offer. The peer user agent might not be able to parse the multiplexed media descriptions and reject the entire offer.

To interoperate with the broadest set of peer user agents, the offering user agent SHOULD<60> construct a MIME structure containing multiple SDP content parts for the SIP INVITE request body, as described in [MS-SIPRE] section 3.15.4.1, with one of the SDP content parts omitting the multiplexed media streams. An SDP part containing the multiplexed media streams SHOULD be placed as the last part in the multi-part MIME structure.

If a set of media streams are multiplexed in an SDP offer, the corresponding media streams in the SDP answer (not including those rejected by the answerer) MUST also be multiplexed. A user agent which does support multiplexing does not have to accept all the multiplexed media streams in an SDP offer. For example, if the SDP offer includes seven multiplexed "main-video" media descriptions, but the receiving user agent supports at most five such "main-video" multiplexed streams, it SHOULD accept the first five "main-video" media descriptions in the offer and reject the remaining.

3.1.5.35 Extensions for multi-channel main-video modality negotiation

This section describes an SDP extension to negotiate a "main-video" modality consisting of multiple channels. A conference hosted by an audio/video MCU might be provisioned to support a "main-video" modality consisting of multiple channels<61>. This would enable a conference participant to simultaneously receive the video sources of multiple other participants (while also sourcing its own video into the conference).

3.1.5.35.1 Requirements to negotiate a multi-channel main-video modality

To negotiate a multi-channel "main-video" modality, the SDP offer and SDP answer MUST contain a set of **m=video** media descriptions meeting the following requirements:

1. The value of the **a=label** attributes MUST be **main-video**.
2. The media streams MUST be multiplexed according to the rules for media stream multiplexing, as described in section 3.1.5.34.1.
3. All media descriptions MUST specify an **a=rtcp-rsize** attribute.
4. All media descriptions MUST specify an **a=rtcp-fb** attribute which declares send and receive support for Picture Loss Indicator and Video Source Request capabilities. For example:**a=rtcp-fb:* x-message app send:x-pli,src recv:x-pli,src**
5. The size of the SSRC range, given by the **a=x-ssrc-range** attribute, SHOULD be 100.

In addition, the media descriptions generated by the MCU user agent MUST contain an **a=x-source-streamid** attribute.

If a conference participant user agent has a video source device (for example, web camera) associated with one of the media streams and will be sending video to the conference, its corresponding media description SHOULD include an **a=x-source** attribute. If the media description from the participant does not contain **a=x-source** then it SHOULD include an **a=recvonly** stream direction attribute, to indicate that the participant intends to use the media stream only to receive video from the conference.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

4 Protocol Examples

4.1 Generic Examples

4.1.1 Client Makes an Offer using ICE as described in IETFDRAFT-ICENAT-06

Following are some SDP examples that demonstrate the offer with most of the extensions specified in this protocol.

The following example is an offer sent by a client.

```
v=0
o=- 0 0 IN IP4 10.56.65.184
s=session
c=IN IP4 10.56.65.184
b=CT:99980
t=0 0
m=audio 37632 RTP/AVP 114 9 111 112 115 116 4 8 0 97101
a=candidate:ir84fUlcDqYH50bs2M/Xn/pDNE+fVfxRTbXBWG34PM8 1 1vvq9h3j8xixI3npD0X9VA UDP 0.830
10.56.65.184 37632
a=candidate:ir84fUlcDqYH50bs2M/Xn/pDNE+fVfxRTbXBWG34PM8 2 1vvq9h3j8xixI3npD0X9VA UDP 0.830
10.56.65.184 63616
a=candidate:Mbmhbdy6gJ1nwkKtoJWa8h9LHlpQ90uT/EiBDOvBPB4 1 76CTu2GXyKtnYlu2ZydjXA TCP 0.190
172.29.105.45 50563
a=candidate:Mbmhbdy6gJ1nwkKtoJWa8h9LHlpQ90uT/EiBDOvBPB4 2 76CTu2GXyKtnYlu2ZydjXA TCP 0.190
172.29.105.45 50563
a=candidate:L6SFpc1rY2GenmqDg0N7eqYMWN0/jI3nH6vttRoU0VE 1 L4J04UBiONZgYNUCyOlT9Q UDP 0.490
172.29.105.45 50403
a=candidate:L6SFpc1rY2GenmqDg0N7eqYMWN0/jI3nH6vttRoU0VE 2 L4J04UBiONZgYNUCyOlT9Q UDP 0.490
172.29.105.45 57283
a=candidate:sct7Qs0hpryFGR/K94UBURz0NOWuThCD7a1iTJyLF8Q 1 ozhWUy01WJw83GTHGukOiw TCP 0.250
10.56.65.184 16512
a=candidate:sct7Qs0hpryFGR/K94UBURz0NOWuThCD7a1iTJyLF8Q 2 ozhWUy01WJw83GTHGukOiw TCP 0.250
10.56.65.184 16512
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:85Sm2QWogZ9N256qxTRhfIRxjUp9q1ISMxwbiloc|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:t20I47Tyj1NDG6H+gWNpIzAzRPfYeQg8pP+ukwoy|2^31|1:1
a=maxptime:200
a=rtcp:63616
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:9 G722/8000
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:101 telephone-event/8000
a=fmtp:101 0-16
m=video 24832 RTP/AVP 12134
a=candidate:kR94HVUEeM0GCz7TfUzEoBojVMo/V+fSSbYUv2MFCxg 1 VzH+zfgjCGjhGEF9aa6ujg UDP 0.840
10.56.65.184 24832
```

```
a=candidate:kR94HVUEeM0GCz7TfUzEoBojVMo/V+fSSbYUv2MFCxg 2 VzH+zfgjCGjhGEF9aa6ujg UDP 0.840
10.56.65.184 39552
a=candidate:Sluz8sKaw20lFkZ8/m6UjK9HU/hYudqY3Xv4yJ1QcQI 1 HX1SFTd1yDyb0gmg5Fl6wQ TCP 0.190
172.29.105.45 55585
a=candidate:Sluz8sKaw20lFkZ8/m6UjK9HU/hYudqY3Xv4yJ1QcQI 2 HX1SFTd1yDyb0gmg5Fl6wQ TCP 0.190
172.29.105.45 55585
a=candidate:J8ubfJUv8xZqKbnKzkH0MvqpRcQE+6jf4/22WG0qzPI 1 r14RJIjw2dTtunLCxLxNGw UDP 0.490
172.29.105.45 56913
a=candidate:J8ubfJUv8xZqKbnKzkH0MvqpRcQE+6jf4/22WG0qzPI 2 r14RJIjw2dTtunLCxLxNGw UDP 0.490
172.29.105.45 57169
a=candidate:Ya8xTTDo0z9kK5Ty6W++HLmVzc95OM1rFnaJ8TT9/hc 1 pt8XROAfQJ9Q0k9nFSaHGg TCP 0.250
10.56.65.184 7680
a=candidate:Ya8xTTDo0z9kK5Ty6W++HLmVzc95OM1rFnaJ8TT9/hc 2 pt8XROAfQJ9Q0k9nFSaHGg TCP 0.250
10.56.65.184 7680
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:BPTL7aWOS9oqHOexSUMoWRCBwGT00ATCrWDI8Pkl|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:N4XsS82yDHiZdPuG2xXvXplKbbPXjeuvup7B9M4H|2^31|1:1
a=maxptime:200
a=rtcp:39552
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:34 H263/90000
```

In the preceding example, the offerer is proposing audio and video as modalities. The offerer supports both SRTP and SSRTP as the mode for encryption and proposes that in its SDP offer, using the **a=crypto** and **a=cryptoscale** attributes. The offerer also requests to only encrypt the media optionally. This is described by specifying "RTP/AVP" as the transport, even though there are **a=crypto** and **a=cryptoscale** attributes present in the SDP message.

Also note that **RTAudio** and **RTVideo** codecs are represented in the codec using dynamic payloads of 114, 115, and 121 and are identified using their encoding names of "x-msrta" and "x-rtvc1" in their corresponding **a=rtpmap** attributes.

4.1.2 Client Receives Response with SSRTP to ICENAT-06 Offer

The following example is a response, or SDP answer, received for the preceding offer.

```
v=0
o=- 0 0 IN IP4 172.29.106.5
s=session
c=IN IP4 172.29.106.5
b=CT:1000
t=0 0
m=audio 57472 RTP/SAVP 9 111 8 0 97 101
c=IN IP4172.29.106.5
a=rtcp:59648
a=candidate:vu6VFdaIZf91YO6DePy/FBzJ0pHopn1lRD/vlUSSJU0 1 bhmEv8fu4QTnweUlMXuiiA UDP 0.900
172.29.106.5 57472
a=candidate:vu6VFdaIZf91YO6DePy/FBzJ0pHopn1lRD/vlUSSJU0 2 bhmEv8fu4QTnweUlMXuiiA UDP 0.900
172.29.106.5 59648
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:L1gAdIcRtzb7OdDbZJhf1PTH2Pjlkq7gxJWva7zX|2^31|1:1
a=label:main-audio
a=rtpmap:9 G722/8000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:97 RED/8000
```

```
a=fmtp:97 red/8000
a=rtpmap:101 telephone-event/8000
a=ptime:60
m=video 58496 RTP/SAVP 121
c=IN IP4 172.29.106.5
a=rtcp:54656
a=candidate:HfCkQziV8VGEey2/VVPSm3m8b0otY/xZilAoWGRo6BM 1 SzMsl46X7YwBpVsbapBY/g UDP 0.900
172.29.106.5 58496
a=candidate:HfCkQziV8VGEey2/VVPSm3m8b0otY/xZilAoWGRo6BM 2 SzMsl46X7YwBpVsbapBY/g UDP 0.900
172.29.106.5 54656
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:sCkL4JFpu5JbaworoJYsXuPvDbpJLavll5JLOJE6|2^31|1:1
a=label:main-video
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;VGA=15;PANO=15
a=x-sourceid:MainCamera
```

The answerer, or remote peer, also encrypts media using SRTP, and so it replies with an SDP answer that has "RTP/SAVP" in the transport in the **m=** line.

Also note that the remote peer prefers to do SSRTP, and thus returns only the **a=cryptoscale** attribute with the "server" value for the *scale-srtp-flavor* parameter. After this exchange of offer and answer, the call is set up and the media is encrypted using SSRTP.

4.1.3 Client Makes an Offer using ICE as described in IETFDRAFT-ICENAT-19

Following are some SDP examples that demonstrate the offer with most of the extensions specified in this document.

The following example is an offer sent by a client.

```
v=0
o=- 0 0 IN IP4 172.24.32.152
s=session
c=IN IP4 172.24.32.152
b=CT:99980
t=0 0
a=x-mediabw:main-video send=585;recv=1416
a=x-devicecaps:audio:send,recv;video:send,recv
m=audio 50005 RTP/AVP 117 114 9 111 112 115 116 4 8 0 97 13 118 101
a=rtcp-fb:* x-message app send:dsh recv:dsh
a=rtcp-rsize
a=ice-ufrag: 6nxO
a=ice-pwd: G6rUJNNaobz8IdDZrAbyFDoO
a=candidate:1 1 UDP 2130706431 172.24.32.152 50005 typ host
a=candidate:1 2 UDP 2130705918 172.24.32.152 50009 typ host
a=candidate:2 1 TCP-PASS 6556159 172.29.105.171 53127 typ relay raddr 172.29.105.171 rport
53127
a=candidate:2 2 TCP-PASS 6556158 172.29.105.171 53127 typ relay raddr 172.29.105.171 rport
53127
a=candidate:3 1 UDP 16648703 172.29.105.171 59353 typ relay raddr 172.29.105.171 rport 59353
a=candidate:3 2 UDP 16648702 172.29.105.171 59627 typ relay raddr 172.29.105.171 rport 59627
a=candidate:4 1 TCP-ACT 7076863 172.29.105.171 53127 typ relay raddr 172.29.105.171 rport
53127
a=candidate:4 2 TCP-ACT 7076350 172.29.105.171 53127 typ relay raddr 172.29.105.171 rport
53127
a=candidate:5 1 TCP-ACT 1684797951 172.24.32.152 50004 typ srflx raddr 172.24.32.152 rport
50004
```

```
a=candidate:5 2 TCP-ACT 1684797438 172.24.32.152 50004 typ srflx raddr 172.24.32.152 rport
50004
a=label:main-audio
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:15PHFDbUI8l9/bOHUYM9geb2IakQY3tMe31TgoPC|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:C62B/j2xrqnk18t4bxXthuGv/Lxc9DmYDG4mnAOK|2^31|1:1
a=maxptime:200
a=rtcp:50009
a=rtpmap:117 G722/8000/2
a=fmtp:117 bitrate=128000
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:9 G722/8000
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
m=video 50012 RTP/AVP 122 121 34 123
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=ice-ufrag: m7A0
a=ice-pwd: yfKPbeepmE8/PvGoIDFq4OId
a=candidate:1 1 UDP 2130706431 172.24.32.152 50012 typ host
a=candidate:1 2 UDP 2130705918 172.24.32.152 50011 typ host
a=candidate:2 1 TCP-PASS 6556159 172.29.105.171 59400 typ relay raddr 172.29.105.171 rport
59400
a=candidate:2 2 TCP-PASS 6556158 172.29.105.171 59400 typ relay raddr 172.29.105.171 rport
59400
a=candidate:3 1 UDP 16648703 172.29.105.171 54004 typ relay raddr 172.29.105.171 rport 54004
a=candidate:3 2 UDP 16648702 172.29.105.171 58581 typ relay raddr 172.29.105.171 rport 58581
a=candidate:4 1 TCP-ACT 7076863 172.29.105.171 59400 typ relay raddr 172.29.105.171 rport
59400
a=candidate:4 2 TCP-ACT 7076350 172.29.105.171 59400 typ relay raddr 172.29.105.171 rport
59400
a=candidate:5 1 TCP-ACT 1684797951 172.24.32.152 50003 typ srflx raddr 172.24.32.152 rport
50003
a=candidate:5 2 TCP-ACT 1684797438 172.24.32.152 50003 typ srflx raddr 172.24.32.152 rport
50003
a=label:main-video
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:KaSgBMqbVyQDtY12ihKmnNslPtpYnq1X7xko32nY|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:1smNZ23vqTBP4oQmBHJ5NsGbSJZG/BWgS6onqlV8|2^31|1:1
a=rtcp:50011
a=x-ssrc-range:101-200
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:34 H263/90000
```

```
a=rtpmap:123 x-ulpfecuc/90000
```

In the previous example, the offerer is proposing audio and video as modalities. The offerer supports both SRTP and SSRTP as the mode for encryption, and proposes that in its SDP offer using the **a=crypto** and **a=cryptoscale** attributes. The offerer also only encrypts the media optionally. This is described by specifying "RTP/AVP" as the transport, even though there are **a=crypto** and **a=cryptoscale** attributes present in the SDP message.

Also note that **RTAudio**, **G722-Stereo**, **RTVideo** and **H.264UC** codecs are represented using dynamic payloads of 114, 115, 117, 121, and 122, and are identified using their encoding names of "x-msrta", "G722", "x-rtvc1" and "X-H264UC" in their corresponding **a=rtpmap** attributes. Note, the full media format name for **G722-Stereo** is "G722/8000/2".

The **ULPFEC-UC** video FEC payload format is also included in the **m=video** media description, using dynamic payload type 123.

4.1.4 Client Receives Response with SSRTP to ICENAT-19 Offer

The following example is a response, or SDP answer, received for the preceding offer.

```
v=0
o=- 0 0 IN IP4 172.24.32.125
s=session
c=IN IP4 172.24.32.125
b=CT:99980
t=0 0
a=x-mediabw:main-video send=585;recv=1416
a=x-devicecaps:audio:send,recv;video:send,recv
m=audio 50018 RTP/SAVP 117 114 9 111 112 115 116 4 8 0 97 13 118 101
a=rtcp-fb:* x-message app send:dsh recv:dsh
a=rtcp-rsize
a=ice-ufrag:yYmQ
a=ice-pwd:T8P5yKtikiFpupO0pOqGatje
a=candidate:1 1 UDP 2130706431 172.24.32.125 50018 typ host
a=candidate:1 2 UDP 2130705918 172.24.32.125 50007 typ host
a=label:main-audio
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:E8zKbdtM9sJdQenqGWVb3sYBp52rxFgS4uwMWy/k|2^31|1:1
a=remote-candidates:1 172.24.32.152 50005 2 172.24.32.152 50009
a=maxptime:200
a=rtcp:50007
a=rtpmap:117 G722/8000/2
a=fmtp:117 bitrate=128000
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:9 G722/8000
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
m=video 50002 RTP/SAVP 122 121 34 123
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=ice-ufrag: tTaJ
a=ice-pwd: 4jUT5Tp48gTR3iEvJiWVVDpG
a=x-caps:121
196611:640:480:25.0:600000:1;262148:352:288:15.0:256000:1;327685:176:144:15.0:180000:1
a=x-caps:34 65537:352:288:15.0:256000:1;131074:176:144:15.0:180000:1
a=candidate:1 1 UDP 2130706431 172.24.32.125 50002 typ host
a=candidate:1 2 UDP 2130705918 172.24.32.125 50008 typ host
a=label:main-video
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:o6ZolyOppaJqBxlYQ9/R4ykPCjgKDJMisiVvXSMb|2^31|1:1
a=remote-candidates:1 172.24.32.152 50012 2 172.24.32.152 50011
a=rtcp:50008
a=x-ssrc-range:101-200
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:34 H263/90000
a=rtpmap:123 x-ulpfecuc/90000
```

The answerer (remote peer) also encrypts the media using SRTP, so it replies with an SDP answer that has "RTP/SAVP" in the transport in the **m=** line.

4.2 Encryption Using SRTP Examples that Demonstrate Extensions

Following are some examples. For brevity, only the pertinent portions of the SDP are displayed.

The following example is an application optionally encrypting the media using either SRTP or Client Scale-SRTP.

```
m=audio 50004 RTP/AVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

The following example is an application optionally encrypting the media using either SRTP or Server SSRTP.

```
m=audio 50004 RTP/AVP 8 97 101
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

The following example is an application optionally encrypting the media using only SRTP.

```
m=audio 50004 RTP/AVP 8 97 101
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

The following example is an application compulsorily encrypting the media using either SRTP or Client Scale-SRTP.

```
m=audio 50004 RTP/SAVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

The following example is an application compulsorily encrypting the media using either SRTP or Server SSRTP.

```
m=audio 50004 RTP/SAVP 8 97 101
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

The following example is an application compulsorily encrypting the media using only SRTP.

```
m=audio 50004 RTP/SAVP 8 97 101
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

4.3 Offer/Answer Exchange for Various SRTP Encryption Scenarios

The following subsections contain examples. Only the relevant portion of the SDP message is included.

4.3.1 Offerer With SRTP or Client Scale-SRTP Encryption Optionally and Answerer With SRTP or Client Scale-SRTP Encryption Optionally

4.3.1.1 Offer

The following example is an offer from an offerer with SRTP or Client Scale-SRTP encryption optionally.

```
m=audio 50004 RTP/AVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

4.3.1.2 Answer

The following example is an answer to the offer in the previous section. This answerer supports SRTP or Client Scale-SRTP encryption optionally.

```
m=audio 50004 RTP/SAVP 8 97 101
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:z8aIuyfeJZ2bkLVNPadciqjXwGMXo0s0IomrZr|2^31|1:1
```

4.3.1.3 Noteworthy points

Following are noteworthy points regarding the situation in which the offerer has SRTP or Client Scale-SRTP encryption optionally and the answerer supports SRTP or Client Scale-SRTP encryption optionally, as shown in the previous sections.

- The answerer supported only SRTP or Client Scale-SRTP. Thus, it responds only to the **a=crypto** line of the offer. In this case, the offerer and answerer can only support the same flavor of the SSRTP, and SSRTP cannot be used.
- The answerer uses the same tag value for his **a=crypto** attribute to signify that it is in response to the **a=crypto** attribute with the same tag value in the offer.
- The answerer changes the transport profile from "AVP" to "SAVP" because both the offerer and answerer have negotiated SRTP for doing encryption.

4.3.2 Offerer With SRTP or Client Scale-SRTP Optionally and Answerer With SRTP or Server SSRTP Encryption Optionally

4.3.2.1 Offer

The following example is an offer from an offerer with SRTP or Client Scale-SRTP encryption optionally.

```
m=audio 50004 RTP/AVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

4.3.2.2 Answer

The following example is an answer to the offer in the previous section. This answerer supports SRTP or Server SSRTP encryption optionally.

```
m=audio 50004 RTP/SAVP 8 97 101
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:Qr98aafIklbkPOReAKItaeUjXwZrOadI893ilaD|2^31|1:1
```

4.3.2.3 Noteworthy points

Following are noteworthy points regarding the situation in which the offerer has SRTP or Client Scale-SRTP encryption optionally and the answerer supports SRTP or Server SSRTP encryption optionally, as shown in the previous sections.

- The answerer supported only SRTP or Server SSRTP, and thus responds only to the **a=cryptoscale** line of the offer. In this case, the offerer and answerer can support the different types of SSRTP, and SSRTP can be used.
- The answerer uses the same tag value for his **a=cryptoscale** attribute to signify that it is in response to the **a=cryptoscale** attribute with the same tag value in the offer.
- The answerer changes the transport profile from "AVP" to "SAVP" because both the offerer and answerer have negotiated SSRTP for doing encryption.

4.3.3 Offerer With SRTP or Client Scale-SRTP Encryption Optionally and Answerer With SRTP Encryption Optionally

4.3.3.1 Offer

The following example is an offer from an offerer with SRTP or Client Scale-SRTP encryption optionally.

```
m=audio 50004 RTP/AVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

4.3.3.2 Answer

The following example is an answer to the offer in the previous section. This answerer supports SRTP encryption optionally.

```
m=audio 50004 RTP/SAVP 8 97 101
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:z8aIuyfeJZ2bkLVNPadciqjXwGMXo0s0IomrZr|2^31|1:1
```

4.3.3.3 Noteworthy points

Following are noteworthy points regarding the situation in which the offerer has SRTP or Client Scale-SRTP encryption optionally and the answerer supports SRTP encryption optionally, as shown in the previous sections.

- The answerer supported only SRTP, and thus responds only to the **a=crypto** line of the offer.
- The answerer uses the same tag value for his **a=crypto** attribute to signify that it is in response to the **a=crypto** attribute with the same tag value in the offer.
- The answerer changes the transport profile from "AVP" to "SAVP" because both the offerer and answerer have negotiated SRTP for doing encryption.

4.3.4 Offerer With SRTP or Client Scale-SRTP Encryption Optionally and Answerer Cannot Support SRTP or SSRTP Encryption

4.3.4.1 Offer

The following example is an offer from an offerer with SRTP or Client Scale-SRTP encryption optionally.

```
m=audio 50004 RTP/AVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

4.3.4.2 Answer

The following example is an answer to the offer in the previous section. This answerer cannot support SRTP or SSRTP encryption.

```
m=audio 50004 RTP/AVP 8 97 101
```

4.3.4.3 Noteworthy points:

The answerer cannot support SRTP or SSRTP and does not respond with any **crypto** or **cryptoscale** attributes.

4.3.5 Offerer With SRTP or Client Scale-SRTP Encryption Compulsorily and Answerer With SRTP Encryption Optionally

4.3.5.1 Offer

The following example is an offer from an offerer with SRTP or Client Scale-SRTP encryption compulsorily.

```
m=audio 50004 RTP/SAVP 8 97 101
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:vV5wrmv9u07pd0QvyHw7rf6yL8e3xXt07AI74T3J|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:Oi0nVM8eJZ2bkLVNeRaqtUeqjXwGMXo0s0IrmoKh|2^31|1:1
```

4.3.5.2 Answer

The following example is an answer to the offer in the previous section. This answerer supports SRTP encryption optionally.

```
m=audio 50004 RTP/SAVP 8 97 101
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:z8aIuyfeJZ2bkLVNPadciqjXwGMXo0s0IomrZr|2^31|1:1
```

4.3.5.3 Noteworthy points

Following are noteworthy points regarding the situation in which the offerer has SRTP or Client Scale-SRTP encryption compulsorily and the answerer supports SRTP encryption optionally, as shown in the previous sections.

- The Offerer encrypts compulsorily using SRTP or SSRTP, and thus sets the transport profile to "SAVP".
- The answerer supports only SRTP, and thus responds only to the **a=crypto** line of the offer.
- The answerer uses the same tag value for the **a=crypto** attribute to signify that it is in response to the **a=crypto** attribute with the same tag value in the offer.

4.4 Restriction to the name and sampling rate for wide band comfort noise

Following is an example of an offer with support for comfort noise.

```
m=audio 57472 RTP/AVP 118 8 0 97 101
c=IN IP4172.29.106.5
a=rtpmap:118 CN/16000
```

4.5 Offer/Answer Exchange for application sharing

4.5.1 Offer

In the following example, the offerer proposes application sharing as a modality in the role of a viewer.

```
m=applicationsharing 25865 TCP/RTP/SAVP 127
a=ice-ufrag:YVBHg
a=ice-pwd:ttsbflut41Em7/nM7qBatyZKEV
a=candidate:1 1 TCP-PASS 2120613887 157.56.65.134 7967 typ host
a=candidate:1 2 TCP-PASS 2120613374 157.56.65.134 7967 typ host
a=candidate:2 1 TCP-ACT 2121006591 157.56.65.134 25865 typ host
a=candidate:2 2 TCP-ACT 2121006078 157.56.65.134 25865 typ host
a=candidate:3 1 TCP-PASS 6556159 172.29.105.171 57506 typ relay raddr 172.29.105.171 rport
57506
a=candidate:3 2 TCP-PASS 6556158 172.29.105.171 57506 typ relay raddr 172.29.105.171 rport
57506
a=candidate:4 1 TCP-ACT 7076607 172.29.105.171 57506 typ relay raddr 172.29.105.171 rport
57506
a=candidate:4 2 TCP-ACT 7076094 172.29.105.171 57506 typ relay raddr 172.29.105.171 rport
57506
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:/qIJxtX8+/VEpKGlTEgcQf84Hzq77umuaFL3y+fA|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:hhVTXYObDO1a5joyG5v0mnm+Djx7E6Hd01Y0Avkt|2^31|1:1
a=setup:passive
a=connection:new
a=rtcp:25865
a=mid:1
a=rtpmap:127 x-data/90000
a=x-applicationsharing-session-id:1
a=x-applicationsharing-role:sharer
a=x-applicationsharing-media-type:rdp
```

4.5.2 Answer

The answerer accepts the offer in the previous section in the role of a sharer.

```
m=applicationsharing 53076 TCP/RTP/SAVP 127
c=IN IP4 172.29.105.171
a=rtpmap:127 x-data/90000
a=mid:1
a=connection:new
a=setup:active
a=rtcp:53076
a=ice-ufrag:A0nvw
a=ice-pwd:dp7UG//SD5FPVC7kD4San8b1YsHaL
a=candidate:1 1 tcp-pass 2120613887 172.29.105.158 57857 typ host raddr 172.29.105.158 rport
57857
a=candidate:1 2 tcp-pass 2120613374 172.29.105.158 57857 typ host raddr 172.29.105.158 rport
57857
a=candidate:2 1 tcp-act 2121006591 172.29.105.158 55959 typ host raddr 172.29.105.158 rport
55959
a=candidate:2 2 tcp-act 2121006078 172.29.105.158 55959 typ host raddr 172.29.105.158 rport
55959
a=candidate:3 1 tcp-pass 6555135 172.29.105.171 53076 typ relay raddr 172.29.105.171 rport
53076
```

```
a=candidate:3 2 tcp-pass 6555134 172.29.105.171 53076 typ relay raddr 172.29.105.171 rport
53076
a=candidate:4 1 tcp-act 7076607 172.29.105.171 53076 typ relay raddr 172.29.105.171 rport
53076
a=candidate:4 2 tcp-act 7076094 172.29.105.171 53076 typ relay raddr 172.29.105.171 rport
53076
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:WgJ76m2+jmIICUHA4wWyrpVJJBoMlgGDuY+1Jz5R|2^31|1:1
a=label:applicationsharing
a=x-applicationsharing-session-id:1
a=x-applicationsharing-role:viewer
a=x-applicationsharing-media-type:rdp
```

4.5.3 Noteworthy points

Following are noteworthy points regarding the situation in which the offerer proposes application sharing as a modality in the role of a viewer and the answerer accepts the offer in the role of a sharer, as shown in the previous sections.

- The offerer has a role of a viewer, while the answerer has a role of a sharer.
- The offerer encrypts compulsorily using SRTP, and thus sets the transport profile to "SAVP". SSRTP is not used.
- The answerer supports only SRTP, and thus responds only to the **a=crypto** line of the offer.
- The answerer uses the same tag value for the **a=crypto** attribute to signify that it is in response to the **a=crypto** attribute with the same tag value in the offer.
- RTP is used over the protocol described in [MS-ICE2] using TCP.

4.6 Offer/Answer Exchange with optimized media path to a gateway

This section describes examples of inbound and outbound calls between the client and a gateway with the media path bypassing OCS.

4.6.1 Incoming call from gateway to client

Note: There is a **CONTENT-ID** MIME header associated with each **application/sdp** and **application/gw-sdp** that are part of the multipart/alternative offer. In the following example, the **application/GW-SDP** offered to the client indicates that the gateway does not amplify media and its bypass id is "9CD08A01-E998-456a-AC8A-D028930E5933".

```
Content-Type: application/sdp
Content-ID: <f5806c1e-a58b-492f-a274-27e84ea28920>
Content-Disposition: Session;handling=optional;ms-proxy-2007fallback
v=0
o=- 5 0 IN IP4 192.168.104.102
s=session
c=IN IP4 192.168.104.102
b=CT:1000000
t=0 0
m=audio 56868 RTP/AVP 0 8 115 13 118 97 101
c=IN IP4 192.168.104.102
a=rtcp:56869
```

```
a=candidate:E3q9M8OJWFaigFVFtD0+u6FqPp0nkHYGAePLOMBTJRc 1 HYiiMeZUh7p4AUdo6XSncw UDP 0.830
192.168.104.102 56868
a=candidate:E3q9M8OJWFaigFVFtD0+u6FqPp0nkHYGAePLOMBTJRc 2 HYiiMeZUh7p4AUdo6XSncw UDP 0.830
192.168.104.102 56869
a=candidate:UzFFBI7awxelfHqPVFlhESQbd1jrYZ5PTn5+6tyH3aU 1 LF4n5rfHFi1/rLoHFWHUPw TCP 0.150
10.9.66.105 56821
a=candidate:UzFFBI7awxelfHqPVFlhESQbd1jrYZ5PTn5+6tyH3aU 2 LF4n5rfHFi1/rLoHFWHUPw TCP 0.150
10.9.66.105 56821
a=candidate:25u0MNHHZjaAh9RPPkpVe7Ba7EdCaxjUYRRqvoIfRkY 1 1sK0tfrBJVJiw820Lcvj3w UDP 0.450
10.9.66.105 59709
a=candidate:25u0MNHHZjaAh9RPPkpVe7Ba7EdCaxjUYRRqvoIfRkY 2 1sK0tfrBJVJiw820Lcvj3w UDP 0.450
10.9.66.105 52813
a=candidate:2DxliVgEarpkaYkb05bFTo8qq9e7BH3eW8ijJ2E3k4M 1 jJ5nCZyij2lvPO66RpXZpA TCP 0.250
192.168.104.102 55429
a=candidate:2DxliVgEarpkaYkb05bFTo8qq9e7BH3eW8ijJ2E3k4M 2 jJ5nCZyij2lvPO66RpXZpA TCP 0.250
192.168.104.102 55429
a=label:main-audio
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:2eyQLFO8vaoOX2GBLg9Qx9mMIJhsuGlL3Vfy65YG|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:QwzL7xoJ9BOMU50/FI72zI9Uh9joIO+LvBKmw+Q0|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:wBfC6An1JRyv1gwfQEkUnPekR6eGRVUyobeGbJHp|2^31
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=rtpmap:97 RED/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16,36
--aE6c0vI9iMfFYM08fnyeWG1occh4Naqt
Content-Type: application/sdp
Content-ID: <713e032e-fde6-48e5-83be-738f1bfdfe36>
v=0
o=- 6 0 IN IP4 192.168.104.102
s=session
c=IN IP4 192.168.104.102
b=CT:1000000
t=0 0
m=audio 51390 RTP/AVP 0 8 115 13 118 97 101
c=IN IP4 192.168.104.102
a=rtcp:51391
a=ice-ufrag:fAgr
a=ice-pwd:fUzyxypL9YjgIpFilSuHWHjW
a=candidate:1 1 UDP 2130706431 192.168.104.102 51390 typ host
a=candidate:1 2 UDP 2130705918 192.168.104.102 51391 typ host
a=candidate:2 1 tcp-pass 6555135 10.9.66.105 57678 typ relay raddr 192.168.104.102 rport
53641
a=candidate:2 2 tcp-pass 6555134 10.9.66.105 57678 typ relay raddr 192.168.104.102 rport
53641
a=candidate:3 1 UDP 16647679 10.9.66.105 53655 typ relay raddr 192.168.104.102 rport 55932
a=candidate:3 2 UDP 16647678 10.9.66.105 54870 typ relay raddr 192.168.104.102 rport 55933
a=candidate:4 1 tcp-act 7076863 10.9.66.105 57678 typ relay raddr 192.168.104.102 rport 53641
a=candidate:4 2 tcp-act 7076350 10.9.66.105 57678 typ relay raddr 192.168.104.102 rport 53641
a=candidate:5 1 tcp-act 1684797951 192.168.104.102 53641 typ srflx raddr 192.168.104.102
rport 53641
a=candidate:5 2 tcp-act 1684797438 192.168.104.102 53641 typ srflx raddr 192.168.104.102
rport 53641
a=label:main-audio
```

```
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:2eyQLFO8vaoOX2GBLg9Qx9mMIJhsuGlL3Vfy65YG|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:QwzL7xoJ9BOMU50/FI72zI9Uh9jolO+LvBKmw+Q0|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:wBfC6An1JRyv1gwfQEkUnPekR6eGRVUyobeGbJHp|2^31
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=rtpmap:97 RED/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16,36
--aE6c0vI9iMfFYM08fnyeWG1occh4Naqt
Content-Type: application/gw-sdp; x-bypassid=9CD08A01-E998-456a-AC8A-D028930E5933
Content-ID: <bcb3fcca-1dc7-4632-aefc-3d4e9947c64f>
Content-Disposition: Session;handling=optional
v=0
o=PSTNgateway1 1344430046 1344429731 IN IP4 192.168.107.12
s=session
c=IN IP4 192.168.107.12
t=0 0
m=audio 6390 RTP/SAVP 0 8 4 2 3 101
c=IN IP4 192.168.107.12
a=rtcp:6391
a=x-bypassid:9CD08A01-E998-456a-AC8A-D028930E5933
a=crypto:1 AES_CM_128_HMAC_SHA1_80 inline:EtVylZp2HonR5Vwd7PFV8kKILnC4P3sKILMY3mAy|2^31|203:1
a=sendrecv
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:4 G723/8000
a=fmtp:4 annexa=no
a=rtpmap:2 G726-32/8000
a=rtpmap:3 GSM/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
a=ptime:20
a=x-mediasettings:signalboostunsupported
--aE6c0vI9iMfFYM08fnyeWG1occh4Naqt—
```

The following code is the answer from the OC, assuming it is in the same location as the gateway and has selected the gateway SDP.

```
ms-accepted-content-id: <bcb3fcca-1dc7-4632-aefc-3d4e9947c64f>
v=0
o=- 0 0 IN IP4 192.168.40.165
s=session
c=IN IP4 192.168.40.165
b=CT:99980
t=0 0
m=audio 28636 RTP/SAVP 0 8 4 101
a=crypto:1 AES_CM_128_HMAC_SHA1_80
inline:2y9P12hgW0bgz/t8CRurDcRQjjOmEpqbztRk2O/L|2^31|1:1
a=maxptime:200
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:4 G723/8000
```

```
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=x-bypass
```

4.6.2 Outbound call from client to gateway

In the following example, the offer from the client indicates that its bypass id is "9CD08A01-E998-456a-AC8A-D028930E5933".

```
------=_NextPart_000_0754_01CAA68D.A421F990
Content-Type: application/sdp
Content-Transfer-Encoding: 7bit
Content-ID: <b36a0b797c2d448684b4cd88213e687b>
Content-Disposition: session; handling=optional; ms-proxy-2007fallback
v=0
o=- 0 0 IN IP4 192.168.40.165
s=session
c=IN IP4 192.168.40.165
b=CT:99980
t=0 0
m=audio 31984 RTP/AVP 114 9 112 111 0 8 116 115 4 97 13 118 101
a=candidate:Eo6N4ZUF5f08I+5P0uqR1tY20IRoszjUqaAq/X2kIts 1 U3pPAm1UlyRGOdhpy2femA UDP 0.830
192.168.40.165 31984
a=candidate:Eo6N4ZUF5f08I+5P0uqR1tY20IRoszjUqaAq/X2kIts 2 U3pPAm1UlyRGOdhpy2femA UDP 0.830
192.168.40.165 31985
a=candidate:3DOp61IDKKyJMLBEbV3e1xQLe4NJD1SlXVzafFyiqgk 1 Obw5WAGIyt1kU/zo7ons/Q TCP 0.190
10.9.66.105 59349
a=candidate:3DOp61IDKKyJMLBEbV3e1xQLe4NJD1SlXVzafFyiqgk 2 Obw5WAGIyt1kU/zo7ons/Q TCP 0.190
10.9.66.105 59349
a=candidate:PzI3B9tYBN+qhYwJcb0j0C42c5ZTR5TyoDWRfb7yXXk 1 eiWWwDXKkSxP58wBK+R/hQ UDP 0.490
10.9.66.105 51744
a=candidate:PzI3B9tYBN+qhYwJcb0j0C42c5ZTR5TyoDWRfb7yXXk 2 eiWWwDXKkSxP58wBK+R/hQ UDP 0.490
10.9.66.105 52795
a=candidate:2kUGrKpDjD4YDF2AS9k1NvGLoCeIEYHSaUfgLxEfdcQ 1 ZFTOm8nfx79vTVzbFxMaKQ TCP 0.250
192.168.40.165 16567
a=candidate:2kUGrKpDjD4YDF2AS9k1NvGLoCeIEYHSaUfgLxEfdcQ 2 ZFTOm8nfx79vTVzbFxMaKQ TCP 0.250
192.168.40.165 16567
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:8XdWiybvpW9FAj7ItDedcqhWjHCKr7gCVq0q56ek|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:s3BkRJUaog532qlFBRdFTpcSCYhoa/hwzr8wV39v|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:nYToZWyhCoDcl/CQLFTE4bTJiCv8YqDlnfF9CVv/|2^31
a=maxptime:200
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:9 G722/8000
a=fmtp:9 bitrate=64000
a=rtpmap:112 G7221/16000
a=fmtp:112 bitrate=24000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:116 AAL2-G726-32/8000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:4 G723/8000
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=x-bypassid:9CD08A01-E998-456a-AC8A-D028930E5933
------=_NextPart_000_0754_01CAA68D.A421F990
Content-Type: application/sdp
Content-Transfer-Encoding: 7bit
Content-ID: <6d62e6a07ddd4ddbab5d9f6474f4175c>
Content-Disposition: session; handling=optional
v=0
o=- 0 0 IN IP4 192.168.40.165
s=session
c=IN IP4 192.168.40.165
b=CT:99980
t=0 0
m=audio 13510 RTP/AVP 114 9 112 111 0 8 116 115 4 97 13 118 101
a=ice-ufrag:oTw+
a=ice-pwd:J7jponEfEPTn6YX8lbeaImJh
a=candidate:1 1 UDP 2130706431 192.168.40.165 13510 typ host
a=candidate:1 2 UDP 2130705918 192.168.40.165 13511 typ host
a=candidate:2 1 TCP-PASS 6556159 10.9.66.105 56378 typ relay raddr 192.168.40.165 rport 12134
a=candidate:2 2 TCP-PASS 6556158 10.9.66.105 56378 typ relay raddr 192.168.40.165 rport 12134
a=candidate:3 1 UDP 16648703 10.9.66.105 58427 typ relay raddr 192.168.40.165 rport 17214
a=candidate:3 2 UDP 16648702 10.9.66.105 52415 typ relay raddr 192.168.40.165 rport 17215
a=candidate:4 1 TCP-ACT 7076863 10.9.66.105 56378 typ relay raddr 192.168.40.165 rport 12134
a=candidate:4 2 TCP-ACT 7076350 10.9.66.105 56378 typ relay raddr 192.168.40.165 rport 12134
a=candidate:5 1 TCP-ACT 1684797951 192.168.40.165 12134 typ srflx raddr 192.168.40.165 rport
12134
a=candidate:5 2 TCP-ACT 1684797438 192.168.40.165 12134 typ srflx raddr 192.168.40.165 rport
12134
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:8XdWiybvpW9FAj7ItDedcqhWjHCKr7gCVq0q56ek|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:s3BkRJUaog532qlFBRdFTpcSCYhoa/hwzr8wV39v|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:nYToZWyhCoDcl/CQLFTE4bTJiCv8YqDlnfF9CVv/|2^31
a=maxptime:200
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:9 G722/8000
a=fmtp:9 bitrate=64000
a=rtpmap:112 G7221/16000
a=fmtp:112 bitrate=24000
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:116 AAL2-G726-32/8000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
a=rtpmap:4 G723/8000
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=x-bypassid:9CD08A01-E998-456a-AC8A-D028930E5933
------=_NextPart_000_0754_01CAA68D.A421F990—
```

The following example is the answer received by the OC, assuming the gateway is in the same location as OC and has opted to bypass.

```
Ms-Accepted-Content-ID: <6d62e6a07ddd4ddbab5d9f6474f4175c>
Rseq: 1
v=0
o=PSTNgateway1 696126319 696126004 IN IP4 192.168.107.12
s=session
c=IN IP4 192.168.107.12
t=0 0
m=audio 6470 RTP/SAVP 0 101
c=IN IP4 192.168.107.12
a=rtcp:6471
a=x-bypass
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:SnMuCMywfjsGUGMiv2Q7aky9OFeHzcZ35VgIllsv|2^31|129:1
a=sendrecv
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
a=ptime:20
a=x-mediasettings:signalboostunsupported
```

4.7 Extensions for music-on-hold

Note that the following examples illustrate SDP only. They do not show **sip.rendering** in the SIP **Contact** header.

4.7.1 Offer specifying music-on-hold

```
m=audio 52033 RTP/SAVP 114 111 112 115 116 4 8 0 97 13 118 101
a=sendonly
a=feature:MoH
```

This offer includes **a=sendonly** and **a=feature:MoH** under the **m=audio** line, indicating that that audio channel is streaming music-on-hold.

4.7.2 Offer removing music-on-hold

```
m=audio 52033 RTP/SAVP 114 111 112 115 116 4 8 0 97 13 118 101
a=sendrecv
```

This offer has **a=sendrecv** and no **a=feature:MoH**, indicating that the audio session (3) is no longer on hold, and is no longer streaming music-on-hold.

4.8 Offer/Answer Exchange for multi-channel main-video modality

This section shows an abbreviated SDP offer/answer sample for multi-channel main-video modality negotiation. In this example, the offerer is a client user agent and the answerer is a user agent for a conference hosted by an audio/video MCU. The client has discovered through some other means that the MCU conference supports a multi-channel main-video modality supporting up to six main-video streams.

4.8.1 Offer from client

In the following offer, the **m=video** lines are highlighted for convenience.

```
v=0
o=- 0 0 IN IP4 10.56.65.184
s=session
c=IN IP4 10.56.65.184
b=CT:99980
t=0 0
a=x-mediabw:main-video send=3300;recv=5000
a=x-devicecaps:audio:send,recv;video:send,recv
m=audio 25520 RTP/SAVP 114 9 0 8 115 97 13 118 101
a=x-ssrc-range:1883723781-1883723781
a=rtcp-fb:* x-message app send:dsh recv:dsh
a=rtcp-rsize
a=label:main-audio
a=x-source:main-audio
a=ice-ufrag:624R
a=ice-pwd:HDoXgdbTbZKVdiqXdrsI9NKJ
a=candidate:1 1 UDP 2130706431 10.56.65.184 25520 typ host
a=candidate:1 2 UDP 2130705918 10.56.65.184 25521 typ host
a=x-candidate-ipv6:2 1 UDP 2130705919 2001:DB8:1e:0:8d22:df2b:9a4d:cdae 5028 typ host
a=x-candidate-ipv6:2 2 UDP 2130705406 2001:DB8:1e:0:8d22:df2b:9a4d:cdae 5029 typ host
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:lKTCvIwuVih2YVzHpQz+HdQ8fvPMYW5UYp/NNWqK|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:VvjQlVF1JMfU/+IbwCDtK9EUwKH3hGXWTBvelNw5|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:W2KYCQ7jENkMU9LWlC4rWGSQPr600V7ghDe/ppb9|2^31
a=rtpmap:114 x-msrta/16000
a=fmtp:114 bitrate=29000
a=rtpmap:9 G722/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:115 x-msrta/8000
a=fmtp:115 bitrate=11800
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
m=video 4460 RTP/SAVP 121 122 123
a=x-ssrc-range:1883723782-1883723881
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=label:main-video
a=x-source:main-video
a=ice-ufrag:DTKC
a=ice-pwd:1IOLIp9EgfHHCu2LgWMNyKph
a=x-caps:121
263:1920:1080:30.0:2000000:1;4359:1280:720:30.0:1500000:1;8455:640:480:30.0:600000:1;12551:64
0:360:30.0:600000:1;16647:352:288:15.0:250000:1;20743:424:240:15.0:250000:1;24839:176:144:15.
0:180000:1
a=candidate:1 1 UDP 2130706431 10.56.65.184 4460 typ host
a=candidate:1 2 UDP 2130705918 10.56.65.184 4461 typ host
a=x-candidate-ipv6:2 1 UDP 2130705919 2001:DB8:1e:0:8d22:df2b:9a4d:cdae 17678 typ host
a=x-candidate-ipv6:2 2 UDP 2130705406 2001:DB8:1e:0:8d22:df2b:9a4d:cdae 17679 typ host
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:PZkRn5zYjVyQJ9SWHJzSaOLWVcqYoXlSgVisknwt|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:KGjadQZQeAiQnSWNFktQihS0f7lVvjfsgSLrv4Du|2^31|1:1
```

```
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:gx00Ws/WZvUGzhx8uPCCzpPvisFqNrT2oPu0AvKn|2^31
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:123 x-ulpfecuc/90000
m=video 4460 RTP/SAVP 121 122 123
a=x-ssrc-range:1883723882-1883723981
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=label:main-video
a=ice-ufrag:DTKC
a=ice-pwd:1IOLIp9EgfHHCu2LgWMNyKph
a=x-caps:121
263:1920:1080:30.0:2000000:1;4359:1280:720:30.0:1500000:1;8455:640:480:30.0:600000:1;12551:64
0:360:30.0:600000:1;16647:352:288:15.0:250000:1;20743:424:240:15.0:250000:1;24839:176:144:15.
0:180000:1
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:5bpiD3MK556jXtoMroaq2NttPaQp2TDEXcU27zFB|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:ty0aWrM0dGKL79vjLonD0h0xeS1KKOMVEcX4HpFf|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:EIQCPyBMn5+BcMqRhcCgrfCWLARoUiinLxqNmXpS|2^31
a=recvonly
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:123 x-ulpfecuc/90000
m=video 4460 RTP/SAVP 121 122 123
a=x-ssrc-range:1883723982-1883724081
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=label:main-video
a=ice-ufrag:DTKC
a=ice-pwd:1IOLIp9EgfHHCu2LgWMNyKph
a=x-caps:121
263:1920:1080:30.0:2000000:1;4359:1280:720:30.0:1500000:1;8455:640:480:30.0:600000:1;12551:64
0:360:30.0:600000:1;16647:352:288:15.0:250000:1;20743:424:240:15.0:250000:1;24839:176:144:15.
0:180000:1
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:c9XI3nPgIwt6Zo/RQv0YWt0HkYp+PNmlnCofxCTV|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:JhUFm1dcl7kx7iHtFn5//lmdTmqdc7p1sqRzo4yf|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:hzjkOCD7H4E3Pnj3pHzZNlLth8FfJWmPtY1QxjtX|2^31
a=recvonly
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:123 x-ulpfecuc/90000
m=video 4460 RTP/SAVP 121 122 123
a=x-ssrc-range:1883724082-1883724181
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=label:main-video
a=ice-ufrag:DTKC
a=ice-pwd:1IOLIp9EgfHHCu2LgWMNyKph
a=x-caps:121
263:1920:1080:30.0:2000000:1;4359:1280:720:30.0:1500000:1;8455:640:480:30.0:600000:1;12551:64
0:360:30.0:600000:1;16647:352:288:15.0:250000:1;20743:424:240:15.0:250000:1;24839:176:144:15.
0:180000:1
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:PrhXgDL89LAYoZGElXTQ80hQWgSLSuo4FzSL9ceC|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:sl4ObHS2Z4xLrgWCV5q9A2ymHuXWgVxgY30BCGAG|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:SwGhiHG69u/7bCIc/c6VCevSoFBnw9nNsZzNr4C6|2^31
```

```
a=recvonly
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:123 x-ulpfecuc/90000
m=video 4460 RTP/SAVP 121 122 123
a=x-ssrc-range:1883724182-1883724281
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=label:main-video
a=ice-ufrag:DTKC
a=ice-pwd:1IOLIp9EgfHHCu2LgWMNyKph
a=x-caps:121
263:1920:1080:30.0:2000000:1;4359:1280:720:30.0:1500000:1;8455:640:480:30.0:600000:1;12551:64
0:360:30.0:600000:1;16647:352:288:15.0:250000:1;20743:424:240:15.0:250000:1;24839:176:144:15.
0:180000:1
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:+aUcI3QNwn5LDiK/fD3mIe1prPeZM7Ycc1l/xZdA|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:57YcGomxEITJazTDR2wSzJg9YBQeiYJCA09uAEUz|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:97UchPM75rLXUrSiS1rgrO5RF+HU3UYRV7T0/mBL|2^31
a=recvonly
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:123 x-ulpfecuc/90000
m=video 4460 RTP/SAVP 121 122 123
a=x-ssrc-range:1883724282-1883724381
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=rtcp-rsize
a=label:main-video
a=ice-ufrag:DTKC
a=ice-pwd:1IOLIp9EgfHHCu2LgWMNyKph
a=x-caps:121
263:1920:1080:30.0:2000000:1;4359:1280:720:30.0:1500000:1;8455:640:480:30.0:600000:1;12551:64
0:360:30.0:600000:1;16647:352:288:15.0:250000:1;20743:424:240:15.0:250000:1;24839:176:144:15.
0:180000:1
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:QgZ6e1f4vImyzmuw8+vwXhF3aIxThEJeMZ3X/e6h|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:HWx/XLi8g70y0gYvWIFDF/zf4uHwhsauStEBnLD6|2^31|1:1
a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:rWRxIlm8P5UqLsimYFGM0vrRPq9pRIah8jxZmy4F|2^31
a=recvonly
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:122 X-H264UC/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:123 x-ulpfecuc/90000
```

4.8.2 Answer from MCU

```
v=0
o=- 124 0 IN IP4 157.56.65.134
s=session
c=IN IP4 157.56.65.134
b=CT:4294967
t=0 0
a=x-mediabw:main-video send=585;recv=1416
m=audio 54262 RTP/SAVP 9 0 8 97 101 13 118
c=IN IP4 157.56.65.134
a=rtcp-rsize
```

```
a=rtcp-fb:* x-message app send:dsh recv:dsh
a=x-ssrc-range:1000-1000
a=x-source-streamid:2000000
a=rtcp:54263
a=ice-ufrag:71s4
a=ice-pwd:Dec85qLpoNfQ/nt+Hto/3pPc
a=candidate:1 1 UDP 2130706431 157.56.65.134 54262 typ host
a=candidate:1 2 UDP 2130705918 157.56.65.134 54263 typ host
a=candidate:2 1 UDP 2130705919 2001:DB8:1e:0:1d46:c7f2:b7bf:b968 56050 typ host
a=candidate:2 2 UDP 2130705406 2001:DB8:1e:0:1d46:c7f2:b7bf:b968 56051 typ host
a=label:main-audio
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:63okg2lFXVufjbwPo+Zzs3ut8ZTEDd8gLtqTJuMB|2^31|1:1
a=rtpmap:9 g722/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:97 RED/8000
a=rtpmap:101 telephone-event/8000
a=rtpmap:13 CN/8000
a=rtpmap:118 CN/16000
a=ptime:20
m=video 62084 RTP/SAVP 122 121 123
c=IN IP4 157.56.65.134
a=rtcp-rsize
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=x-ssrc-range:1001-1100
a=x-source-streamid:2000256
a=x-sourceid:MainCamera
a=rtcp:62085
a=ice-ufrag:IGiI
a=ice-pwd:Lx7x2fIv72zcqtn6gTdIzXWm
a=candidate:1 1 UDP 2130706431 157.56.65.134 62084 typ host
a=candidate:1 2 UDP 2130705918 157.56.65.134 62085 typ host
a=candidate:2 1 UDP 2130705919 2001:DB8:1e:0:1d46:c7f2:b7bf:b968 65356 typ host
a=candidate:2 2 UDP 2130705406 2001:DB8:1e:0:1d46:c7f2:b7bf:b968 65357 typ host
a=label:main-video
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:UfGmYYpS/4qZeo3s1A29oUvd3VsWL+pQQdeFXRgJ|2^31|1:1
a=rtpmap:122 x-h264uc/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;PANO=15;VGA=15
a=x-caps:121
263:640:480:30.0:600000:1;4359:640:360:30.0:600000:1;8455:352:288:15.0:250000:1;12551:424:240
:15.0:250000:1;16647:176:144:15.0:180000:1
a=rtpmap:123 x-ulpfecuc/90000
m=video 62084 RTP/SAVP 122 121 123
c=IN IP4 157.56.65.134
a=rtcp-rsize
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=x-ssrc-range:1101-1200
a=x-source-streamid:2000768
a=x-sourceid:MainCamera
a=rtcp:62085
a=ice-ufrag:IGiI
a=ice-pwd:Lx7x2fIv72zcqtn6gTdIzXWm
a=label:main-video
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:UfGmYYpS/4qZeo3s1A29oUvd3VsWL+pQQdeFXRgJ|2^31|1:1
```

```
a=sendonly
a=rtpmap:122 x-h264uc/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;PANO=15;VGA=15
a=x-caps:121
263:640:480:30.0:600000:1;4359:640:360:30.0:600000:1;8455:352:288:15.0:250000:1;12551:424:240
:15.0:250000:1;16647:176:144:15.0:180000:1
a=rtpmap:123 x-ulpfecuc/90000
m=video 62084 RTP/SAVP 122 121 123
c=IN IP4 157.56.65.134
a=rtcp-rsize
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=x-ssrc-range:1201-1300
a=x-source-streamid:2001024
a=x-sourceid:MainCamera
a=rtcp:62085
a=ice-ufrag:IGiI
a=ice-pwd:Lx7x2fIv72zcqtn6gTdIzXWm
a=label:main-video
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:UfGmYYpS/4qZeo3s1A29oUvd3VsWL+pQQdeFXRgJ|2^31|1:1
a=sendonly
a=rtpmap:122 x-h264uc/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;PANO=15;VGA=15
a=x-caps:121
263:640:480:30.0:600000:1;4359:640:360:30.0:600000:1;8455:352:288:15.0:250000:1;12551:424:240
:15.0:250000:1;16647:176:144:15.0:180000:1
a=rtpmap:123 x-ulpfecuc/90000
m=video 62084 RTP/SAVP 122 121 123
c=IN IP4 157.56.65.134
a=rtcp-rsize
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=x-ssrc-range:1301-1400
a=x-source-streamid:2001280
a=x-sourceid:MainCamera
a=rtcp:62085
a=ice-ufrag:IGiI
a=ice-pwd:Lx7x2fIv72zcqtn6gTdIzXWm
a=label:main-video
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:UfGmYYpS/4qZeo3s1A29oUvd3VsWL+pQQdeFXRgJ|2^31|1:1
a=sendonly
a=rtpmap:122 x-h264uc/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;PANO=15;VGA=15
a=x-caps:121
263:640:480:30.0:600000:1;4359:640:360:30.0:600000:1;8455:352:288:15.0:250000:1;12551:424:240
:15.0:250000:1;16647:176:144:15.0:180000:1
a=rtpmap:123 x-ulpfecuc/90000
m=video 62084 RTP/SAVP 122 121 123
c=IN IP4 157.56.65.134
a=rtcp-rsize
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=x-ssrc-range:1401-1500
a=x-source-streamid:2001536
a=x-sourceid:MainCamera
```

```
a=rtcp:62085
a=ice-ufrag:IGiI
a=ice-pwd:Lx7x2fIv72zcqtn6gTdIzXWm
a=label:main-video
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:UfGmYYpS/4qZeo3s1A29oUvd3VsWL+pQQdeFXRgJ|2^31|1:1
a=sendonly
a=rtpmap:122 x-h264uc/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;PANO=15;VGA=15
a=x-caps:121
263:640:480:30.0:600000:1;4359:640:360:30.0:600000:1;8455:352:288:15.0:250000:1;12551:424:240
:15.0:250000:1;16647:176:144:15.0:180000:1
a=rtpmap:123 x-ulpfecuc/90000
m=video 62084 RTP/SAVP 122 121 123
c=IN IP4 157.56.65.134
a=rtcp-rsize
a=rtcp-fb:* x-message app send:src,x-pli recv:src,x-pli
a=x-ssrc-range:1501-1600
a=x-source-streamid:2001792
a=x-sourceid:MainCamera
a=rtcp:62085
a=ice-ufrag:IGiI
a=ice-pwd:Lx7x2fIv72zcqtn6gTdIzXWm
a=label:main-video
a=cryptoscale:1 server AES_CM_128_HMAC_SHA1_80
inline:UfGmYYpS/4qZeo3s1A29oUvd3VsWL+pQQdeFXRgJ|2^31|1:1
a=sendonly
a=rtpmap:122 x-h264uc/90000
a=fmtp:122 packetization-mode=1;mst-mode=NI-TC
a=rtpmap:121 x-rtvc1/90000
a=fmtp:121 CIF=15;PANO=15;VGA=15
a=x-caps:121
263:640:480:30.0:600000:1;4359:640:360:30.0:600000:1;8455:352:288:15.0:250000:1;12551:424:240
:15.0:250000:1;16647:176:144:15.0:180000:1
a=rtpmap:123 x-ulpfecuc/90000
```

5 Security

5.1 Security Considerations for Implementers

Although media encryption is supported, the exchange of encryption information to encrypt the media is not encrypted. To protect the encryption information during the exchange, the application can use TLS to carry the SIP traffic. Any other security considerations are covered by SIP and SDP.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® 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
- Microsoft® Lync 15 Technical Preview
- Microsoft® Lync Server 15 Technical Preview

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 3.1.5.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, these parameters are required.

<2> Section 3.1.5.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The H.264UC video codec is not supported.

<3> Section 3.1.5.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The G722-Stereo audio codec is not supported.

<4> Section 3.1.5.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The ULPFEC-UC video payload format is not supported.

<5> Section 3.1.5.3: Office Communications Server 2007, Office Communications Server 2007 R2, Lync Server 2010: For RTVideo, the RTP payload type value MUST be 121.

<6> Section 3.1.5.4: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: The ordering of the media formats in a received offer is ignored.

<7> Section 3.1.5.5: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010: The RTP payload type number used to specify DTMF MUST be "101".

<8> Section 3.1.5.7: Office Communications Server 2007, Office Communicator 2007: Comfort noise is not supported. For all other products, the name of the payload used for comfort noise is required to be "CN".

<9> Section 3.1.5.7: Office Communications Server 2007, Office Communicator 2007: Comfort noise is not supported. For all other products, the sampling rate is required to be 8,000 or 16,000.

<10> Section 3.1.5.9: Office Communications Server 2007, Office Communicator 2007: Application sharing is not supported. For all other product, application sharing is required to use ICE over TCP.

<11> Section 3.1.5.9: Office Communications Server 2007, Office Communications Server 2007 R2, Lync Server 2010: Due to an error in these products, a connection-oriented transport specified in the m= line of the first SDP offer is not rejected. However, these products do not support a connection-oriented transport for the initial active media address for audio or video.

<12> Section 3.1.5.10.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This requirement is not enforced. For all other products, when used in the context of this protocol, the **a=connection** attribute is required to have the value "existing".

<13> Section 3.1.5.12.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported. For all other products, when the offer is forked, SDP answers not in reliable provisional responses are required to be sent only from a 0 or 1 device.

<14> Section 3.1.5.12.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<15> Section 3.1.5.12.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<16> Section 3.1.5.12.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<17> Section 3.1.5.12.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<18> Section 3.1.5.12.3: Office Communications Server 2007, Office Communicator 2007: Early media is not supported.

<19> Section 3.1.5.12.3: Office Communicator 2007 R2, Lync 2010, Lync 15 Technical Preview: The SDP answer in the final 200 response might contain some differences from the SDP answer in the provisional 18x-level response.

<20> Section 3.1.5.12.3: Office Communications Server 2007, Office Communicator 2007: Early media is not supported.

<21> Section 3.1.5.12.3: Office Communicator 2007 R2, Lync 2010, Lync 15 Technical Preview: The SDP answer in the final 200 response might contain some differences from the SDP answer in the provisional 18x-level response.

<22> Section 3.1.5.12.4: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported. For all other products, when a SIP INVITE request is NOT forked and an SDP answer is received in the provisional response, ICE processing is required to proceed as if the SDP was received in the final response.

<23> Section 3.1.5.13: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other products are required to follow the exceptions as specified.

<24> Section 3.1.5.13: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other products are required to follow the exceptions as specified.

<25> Section 3.1.5.13: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<26> Section 3.1.5.16: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010: The IP address type in a c= line MUST be IPv4.

<27> Section 3.1.5.18.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other products MUST include this attribute.

<28> Section 3.1.5.18.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other products, the sharing party MUST set the role to "sharer."

<29> Section 3.1.5.18.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other products, the viewing party MUST set the role to "viewer."

<30> Section 3.1.5.18.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, the (session-id, role) pair is required to be unique for each active **m=** line.

<31> Section 3.1.5.18.4: Office Communications Server 2007, Office Communicator 2007, Lync Server 2010, Lync 2010: This behavior is not supported. For all other products, the **a=mid** attribute is required.

<32> Section 3.1.5.19: Lync Server 2010: The sess-version parameter of the o= line is incremented in subsequent SDP offers and SDP answers.

<33> Section 3.1.5.19: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The addrtype parameter MUST be "IP4".

<34> Section 3.1.5.21.1: Office Communicator 2007 R2, Lync 2010: An **a=candidate** attribute containing an IPv6 connection-address field causes an SDP parsing error.

<35> Section 3.1.5.21.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other product are required to gather the Relayed, Server Reflexive candidates and perform connectivity checks.

<36> Section 3.1.5.21.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other products are not allowed to generate an SDP with the ice-options attribute.

<37> Section 3.1.5.21.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. All other products are not allowed to generate an SDP with the ice-mismatch attribute.

<38> Section 3.1.5.21.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, the default destination for a media component is required to have a corresponding candidate attribute.

<39> Section 3.1.5.21.5: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<40> Section 3.1.5.22.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, for audio/video calls, the default candidate is not allowed to be TCP.

<41> Section 3.1.5.22.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, for application sharing calls, the default candidate is required to be TCP.

<42> Section 3.1.5.22.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, for application sharing media type, the local candidates are required to be TCP.

<43> Section 3.1.5.23.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, the client is required to change the direction of all streams to inactive.

<44> Section 3.1.5.23.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<45> Section 3.1.5.23.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, the client is required to change the direction of all streams to sendrecv.

<46> Section 3.1.5.23.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<47> Section 3.1.5.24: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: The x-caps attribute is not supported, and is ignored.

<48> Section 3.1.5.24: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2 This behavior is not supported. For all other products, the protocol peer is required to follow the requirements listed.

<49> Section 3.1.5.25: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2 This behavior is not supported.

<50> Section 3.1.5.26: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<51> Section 3.1.5.28.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The x-mediabw attribute is not supported. It is ignored if present in a received SDP message.

<52> Section 3.1.5.28.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The x-mediabw attribute is not supported. It is ignored if present in a received SDP message.

<53> Section 3.1.5.28.2: Lync Server 15 Technical Preview, Lync 15 Technical Preview:An x-mediabw attribute specifying a "main-audio" label is ignored.

<54> Section 3.1.5.29.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The x-devicecaps attribute is not supported. It is ignored if present in a received SDP message.

<55> Section 3.1.5.30.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The rtcp-rsize and rtcp-fb attributes are not supported. They are ignored if present in a received SDP message.

<56> Section 3.1.5.31.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The a=x-ssrc-range attribute is not supported. It is ignored if present in a received SDP message

<57> Section 3.1.5.32.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The a=x-source-streamid attribute is not supported. It is ignored if present in a received SDP message.

<58> Section 3.1.5.33.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The a=x-source attribute is not supported. It is ignored if present in a received SDP message.

<59> Section 3.1.5.34: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: Multiplexing is not supported.

<60> Section 3.1.5.34.2: Office Communicator 2007 R2, Lync 2010: This is a MUST in order to ensure interop with these products.

<61> Section 3.1.5.35: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: A "main-video" modality with more than one channel is not supported.

7 Change Tracking

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

8 Index

A

Abstract data model 17
Applicability 14

C

Capability negotiation 14
Change tracking 77
Client makes an offer using ICE V19
  example 50
Client makes an offer using ICE-06
  example 48
Client receives response to ICE V19 offer
  example 52
Client receives response to ICE-06 offer
  example 49

D

Data model - abstract 17

E

Examples
  application sharing
    offer/answer exchanges
      answer 58
      noteworthy points 59
      offer 58
  client makes an offer using ICE V19 50
  client makes an offer using ICE-06 48
  client receives response to ICE V19 offer 52
  client receives response to ICE-06 offer 49
  music-on-hold 64
    offer removing 64
    offer specifying 64
  name and sampling rate restrictions 57
  optimized media path to gateway
    offer/answer exchange 59
      incoming call 59
      outbound call 62
  SRTP encryption 53
    offer/answer exchanges 54

F

Fields - vendor-extensible 15

G

Glossary 8

H

Higher-layer triggered events 17

I

ICE V19
  example
    client makes an offer 50
    client receives response with SSRTP 52
  message processing
    deviations 31
ICE-06
  example
    client makes an offer 48
    client receives response with SSRTP 49
  message processing
    deviations 30
Implementer - security considerations 71
Index of security parameters 71
Informative references 11
Initialization 17
Introduction 8

L

Local events 47

M

Message processing 17
  a=connection attribute 24
  a=crypto attribute 17
  a=rtcp attribute 28
  a=setup attribute 24
  a=x-caps attribute 34
  applicationsharing media type 29
  call hold and retrieve 33
  connection-oriented media address support 24
  deviations from ICE V19 31
  deviations from ICE-06 30
  deviations from ICE-TCP-07 33
  diagnostic info in SDP 36
  dual-tone multi-frequency(DTMF) in SDP 22
  early media support 26
  format preference 22
  music-on-hold 38
  negotiating SRTP optionally 23
  new payload types 21
  o= line in SDP 30
  optimize media path to gateway 35
  reliable provisional response processing 27
  renegotiation of SRTP or SSRTP encryption 28
  restriction on name and sampling rate 23
  restriction on name of RTP payload 23
  specifying and negotiating SSRTP 18
  text telephony support 25
Messages
  syntax 16
  transport 16
Music-on-hold
  example 64
    offer removing 64
    offer specifying 64
  message processing 38

N

Name and sampling rate restrictions
  example 57
  message processing 23
Normative references 9

O

Offer/answer exchange with optimized media path to gateway
  example 59
    incoming call 59
    outbound call 62
Offer/answer exchanges for application sharing examples
    answer 58
    noteworthy points 59
    offer 58
Offer/answer exchanges for SRTP encryption
  examples 54
Optimize media path to gateway
  example 59
    incoming call 59
    outbound call 62
  message processing 35
Overview (synopsis) 11

P

Parameters - security index 71
Preconditions 14
Prerequisites 14
Product behavior 72

R

References 9
  informative 11
  normative 9
Relationship to other protocols 14

S

Security
  implementer considerations 71
  parameter index 71
Sequencing rules 17
  a=connection attribute 24
  a=crypto attribute 17
  a=rtcp attribute 28
  a=setup attribute 24
  a=x-caps attribute 34
  applicationsharing media type 29
  call hold and retrieve 33
  connection-oriented media address support 24
  deviations from ICE V19 31
  deviations from ICE-06 30
  deviations from ICE-TCP-07 33
  diagnostic info in SDP 36
  dual-tone multi-frequency(DTMF) in SDP 22
  early media support 26
  format preference 22
  music-on-hold 38
  negotiating SRTP optionally 23
  new payload types 21
  o= line in SDP 30
  optimize media path to gateway 35
  reliable provisional response processing 27
  renegotiation of SRTP or SSRTP encryption 28
  restriction on name and sampling rate 23
  restriction on name of RTP payload 23
  specifying and negotiating SSRTP 18
  text telephony support 25
SRTP encryption
  example 53
SRTP encryption offer/answer exchanges
  examples 54
SSRTP
  message processing 18
Standards assignments 15

T

Timer events 47
Timers 17
Tracking changes 77
Transport 16
Triggered events 17

V

Vendor-extensible fields 15
Versioning 14