MMUSIC
Internet-Draft
Obsoletes: 5245 (if approved)
Intended status: Standards Track
Expires: May 27, 2018

M. Petit-Huguenin
Impedance Mismatch
A. Keranen
Ericsson
S. Nandakumar
Cisco Systems

November 23, 2017

Session Description Protocol (SDP) Offer/Answer procedures for Interactive Connectivity Establishment (ICE) draft-ietf-mmusic-ice-sip-sdp-16

Abstract

This document describes Session Description Protocol (SDP) Offer/Answer procedures for carrying out Interactive Connectivity Establishment (ICE) between the agents.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 27, 2018.

Copyright Notice

Table of Contents

1. Introduction

This document describes how Interactive Connectivity Establishment (ICE) is used with Session Description Protocol (SDP) offer/answer [RFC3264]. The ICE specification [ICE-BIS] describes procedures that are common to all usages of ICE and this document gives the additional details needed to use ICE with SDP offer/answer.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Readers should be familiar with the terminology defined in [RFC3264], in [RFC7656], in [ICE-BIS] and the following:

Default Destination/Candidate: The default destination for a component of a media stream is the transport address that would be used by an agent that is not ICE aware. A default candidate for a component is one whose transport address matches the default destination for that component. For the RTP component, the default IP address is in the "c=" line of the SDP, and the port is in the "m=" line. For the RTCP component, the address and port are indicated using the "a=rtcp" attribute defined in [RFC3605], if present; otherwise, the RTCP component address is same as the address of the RTP component, and its port is one greater than the port of the RTP component.

3. ICE Candidate Exchange and Offer/Answer Mapping

[ICE-BIS] defines ICE candidate exchange as the process for ICE agents (Initiator and Responder) to exchange their candidate information required for ICE processing at the agents. For the purposes of this specification, the candidate exchange process corresponds to the [RFC3264] Offer/Answer protocol and the terminologies offerer and answerer correspond to the initiator and responder terminologies from [ICE-BIS] respectively.

4. SDP Offer/Answer Procedures

4.1. Initial Offer/Answer Exchange

4.1.1. Sending the Initial Offer

The offerer shall follow the procedures defined in section 5 of [ICE-BIS] to gather, prioritize and eliminate the redundant candidates. It then chooses the default candidates and encodes them in the SDP to be sent to its peer, the answerer.

4.1.1.1. Choosing Default Candidates

A candidate is said to be default if it would be the target of media from a non-ICE peer; that target is called the DEFAULT DESTINATION. An agent MUST choose a set of candidates, one for each component of each in-use media stream, to be default. A media stream is in-use if it does not have a port of zero (which is used in RFC 3264 to reject a media stream). Consequently, a media stream is in-use even if it is marked as a=inactive [RFC4566] or has a bandwidth value of zero.

An agent may choose any type of the candidate as the default, if the chosen candidates increases the likelihood of success with the peer

that is being contacted if ICE is not being used. It is recommended that, when multiple candidates are used, UDP based candidates SHOULD be included wherever possible and default candidate SHOULD be chosen from one of those UDP candidates. The proto value MUST match the transport protocol associated with the default candidate. If UDP transport is used for the default candidate, the 'proto' value MUST include UDP and the 'proto' value MUST be TCP when the transport is TCP for the default candidate.

Since it is RECOMMENDED that default candidates be chosen based on the likelihood of those candidates to work with the peer that is being contacted if ICE is not being used. Many factors may influence such a decision in a given agent. In scenarios where the agent is fully aware of its peer's location and can reach the peer directly, choosing the host candidates as the default may well be sufficient. If the network configuration under which the agents operates is static and known beforehand, either the host or the server reflexives candidates can serve as the default candidates (depending on if a given agent is behind NAT and their reachability). If the agent is completely unaware of the peer's location or no assumptions can be made of network characteristics and the connectivity, the relayed candidates might be the only option as the default candidate. Having the decision of choosing the default candidate as a configurable option in the implementations might provide agents the flexibility to take into account the aforementioned criteria. Barring such configuration flexibility, it is RECOMMENDED that the default candidates be the relayed candidates (if relayed candidates are available), server reflexive candidates (if server reflexive candidates are available), and finally host candidates.

4.1.1.2. Encoding the SDP

The process of encoding the SDP is identical between full and lite implementations.

The agent will include an "m=" line for each Source Stream [RFC7656] it wishes to use. The ordering of source streams in the SDP is relevant for ICE. ICE will perform its connectivity checks for the first "m=" line first, and consequently media will be able to flow for that stream first. Agents SHOULD place their most important source stream, if there is one, first in the SDP.

There will be a candidate attribute for each candidate for a particular source stream. Section 5 provides detailed rules for constructing this attribute.

STUN connectivity checks between agents are authenticated using the short-term credential mechanism defined for STUN [RFC5389]. This

mechanism relies on a username and password that are exchanged through protocol machinery between the client and server. The username fragment and password are exchanged in the ice-ufrag and ice-pwd attributes, respectively.

If an agent is a lite implementation, it MUST include an "a=ice-lite" session-level attribute in its SDP to indicate this. If an agent is a full implementation, it MUST NOT include this attribute.

Section 10 of [ICE-BIS] defines a new ICE option, 'ice2'. This option is used by ICE Agents to indicate their compliancy with [ICE-BIS] specification as compared to the [RFC5245]. If the Offering agent is a [ICE-BIS] compliant implementation, a session level ICE option to indicate the same (via the "a=ice-options:ice2" SDP line) MUST be included.

The default candidates are added to the SDP as the default destination for media. For source streams based on RTP, this is done by placing the IP address and port of the RTP candidate into the "c=" and "m=" lines, respectively. If the agent is utilizing RTCP and if RTCP candidate is present and is not equal to the same address and the next higher port number of the RTP candidate, the agent MUST encode the RTCP candidate using the a=rtcp attribute as defined in [RFC3605]. If RTCP is not in use, the agent MUST signal that using b=RS:0 and b=RR:0 as defined in [RFC3556]

The transport addresses that will be the default destination for media when communicating with non-ICE peers MUST also be present as candidates in one or more a=candidate lines.

ICE provides for extensibility by allowing an offer or answer to contain a series of tokens that identify the ICE extensions used by that agent. If an agent supports an ICE extension, it MUST include the token defined for that extension in the ice-options attribute.

The following is an example SDP message that includes ICE attributes (lines folded for readability):

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 10.0.1.1
s=
c=IN IP4 192.0.2.3
t=0 0
a=ice-options:ice2
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
m=audio 45664 RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 10.0.1.1 8998 typ host
a=candidate:2 1 UDP 1694498815 192.0.2.3 45664 typ srflx raddr
 10.0.1.1 rport 8998
```

Once an agent has sent its offer or its answer, that agent MUST be prepared to receive both STUN and media packets on each candidate. As discussed in section 12.1 of [ICE-BIS], media packets can be sent to a candidate prior to its appearance as the default destination for media in an offer or answer.

4.1.2. Receiving the Initial Offer

On receiving the offer, the answerer verifies the support for ICE (section 5.4 of [ICE-BIS]), determines its role (section 6.1.1 of [ICE-BIS]), gathers candidates (section 5 of [ICE-BIS]), encodes the candidates in an SDP answer and sends it to its peer, the offerer. The answerer shall then follow the steps defined in sections 6.1.3 and 6.1.4 of [ICE-BIS] to schedule the ICE connectivity checks.

The below sub-sections provide additional requirements associated with the processing of the offerer’s SDP pertaining to this specification.

4.1.2.1. ICE Option "ice2" considerations

If the SDP offer contains a session level ICE option, "ice2" , and if the answering ICE Agent is also an [ICE-BIS] compliant implementation, then the generated SDP answer MUST include the session level "a=ice-options:ice2" SDP line.

4.1.2.2. Choosing Default Candidates

The process for selecting default candidates at the answerer is identical to the process followed by the offerer, as described in Section 4.1.1.1 for full implementations in this specification and section 5.2 of [ICE-BIS] for lite implementations.

4.1.2.3. ICE Mismatch

The agent will proceed with the ICE procedures defined in [ICE-BIS] and this specification if, for each media stream in the SDP it received, the default destination for each component of that media stream appears in a candidate attribute. For example, in the case of RTP, the IP address and port in the "c=" and "m=" lines, respectively, appear in a candidate attribute and the value in the rtcp attribute appears in a candidate attribute.

If this condition is not met, the agent MUST process the SDP based on normal RFC 3264 procedures, without using any of the ICE mechanisms described in the remainder of this specification with the following exceptions:

1. The agent MUST follow the rules of section 11 of [ICE-BIS], which describe keepalive procedures for all agents.

2. If the agent is not proceeding with ICE because there were a=candidate attributes, but none that matched the default destination of the media stream, the agent MUST include an a=ice-mismatch attribute in its answer.

3. If the default candidates were relayed candidates learned through a TURN server, the agent MUST create permissions in the TURN server for the IP addresses learned from its peer in the SDP it just received. If this is not done, initial packets in the media stream from the peer may be lost.

4.1.2.4. Determining Role

In unusual cases, described in Appendix C, it is possible for both agents to mistakenly believe they are controlled or controlling. To resolve this, each agent MUST select a random number, called the tie-breaker, uniformly distributed between 0 and (2**64) - 1 (that is, a 64-bit positive integer). This number is used in connectivity checks to detect and repair this case, as described in section 7.1.3 of [ICE-BIS].

4.1.3. Receipt of the Initial Answer

On receiving the SDP answer, the offerer performs steps similar to answerer’s processing of the offer. The offerer verifies the answerer’s ICE support determines, its role, and processes the answerer’s candidates to schedule the connectivity checks (section 7 of [ICE-BIS]).

If the offerer had included the "ice2" ICE Option in the offer and the SDP answer also includes a similar session level ICE option, then the peers are [ICE-BIS] compliant implementations. On the other hand, if the SDP Answer lacks such a ICE option, the offerer defaults to the procedures that are backward compatible with the [RFC5245] specification.

4.1.3.1. ICE Mismatch

The logic at the offerer is identical to that of the answerer as described in section 5.4 of [ICE-BIS], with the exception that an offerer would not ever generate a=ice-mismatch attributes in an SDP.

In some cases, the answer may omit a=candidate attributes for the media streams, and instead include an a=ice-mismatch attribute for one or more of the media streams in the SDP. This signals to the offerer that the answerer supports ICE, but that ICE processing was not used for the session because a signaling intermediary modified the default destination for media components without modifying the corresponding candidate attributes. See Section 11.2.2 for a discussion of cases where this can happen. This specification provides no guidance on how an agent should proceed in such a failure case.

4.1.4. Performing Connectivity Checks

The possibility for role conflicts described in section 7.3.1.1 of [ICE-BIS] applies to this usage and hence all full agents MUST implement the role conflict repairing mechanism. Also both full and lite agents MUST utilize the ICE-CONTROLLED and ICE-CONTROLLING attributes as described in section 7.1.3 of [ICE-BIS].

4.1.5. Concluding ICE

Once the state of each check list is Completed, If an agent is controlling, it examines the highest-priority nominated candidate pair for each component of each media stream. If any of those candidate pairs differ from the default candidate pairs in the most recent offer/answer exchange, the controlling agent MUST generate an updated offer as described in Section 4.2.

However, If the support for ’ice2’ ICE Option is in use, the highest-priority nominated candidate is noted and sent in the subsequent offer/answer exchange as the default candidate and no updated offer is needed to fix the default candidate.

4.2. Subsequent Offer/Answer Exchanges

Either agent MAY generate a subsequent offer at any time allowed by [RFC3264]. This section defines rules for construction of subsequent offers and answers.

Should a subsequent offer fail, ICE processing continues as if the subsequent offer had never been made.

4.2.1. Generating the Offer

4.2.1.1. Procedures for All Implementations

4.2.1.1.1. ICE Restarts

An agent MAY restart ICE processing for an existing media stream as defined in section 9 of [ICE-BIS].

The rules governing the ICE restart imply that setting the IP address in the "c=" line to 0.0.0.0 will cause an ICE restart. Consequently, ICE implementations MUST NOT utilize this mechanism for call hold, and instead MUST use a=inactive and a=sendonly as described in [RFC3264].

To restart ICE, an agent MUST change both the ice-pwd and the ice-ufrag for the media stream in an offer. Note that it is permissible to use a session-level attribute in one offer, but to provide the same ice-pwd or ice-ufrag as a media-level attribute in a subsequent offer. This is not a change in password, just a change in its representation, and does not cause an ICE restart.

An agent sets the rest of the fields in the SDP for this media stream as it would in an initial offer of this media stream (see Section 4.1.1.2). Consequently, the set of candidates MAY include some, none, or all of the previous candidates for that stream and MAY include a totally new set of candidates.

4.2.1.1.2. Removing a Media Stream

If an agent removes a media stream by setting its port to zero, it MUST NOT include any candidate attributes for that media stream and SHOULD NOT include any other ICE-related attributes defined in Section 5 for that media stream.

4.2.1.1.3. Adding a Media Stream

If an agent wishes to add a new media stream, it sets the fields in the SDP for this media stream as if this was an initial offer for that media stream (see Section 4.1.1.2). This will cause ICE processing to begin for this media stream.

4.2.1.2. Procedures for Full Implementations

This section describes additional procedures for full implementations, covering existing media streams.

4.2.1.2.1. Existing Media Streams with ICE Running

If an agent generates an updated offer including a media stream that was previously established, and for which ICE checks are in the Running state, the agent follows the procedures defined here.

An agent MUST include candidate attributes for all local candidates it had signaled previously for that media stream. The properties of that candidate as signaled in SDP -- the priority, foundation, type, and related transport address -- SHOULD remain the same. The IP address, port, and transport protocol, which fundamentally identify that candidate, MUST remain the same (if they change, it would be a new candidate). The component ID MUST remain the same. The agent MAY include additional candidates it did not offer previously (see section 4.2.4.1.1), but which it has gathered since the last offer/answer exchange, including peer reflexive candidates.

The agent MAY change the default destination for media. As with initial offers, there MUST be a set of candidate attributes in the offer matching this default destination.

4.2.1.2.2. Existing Media Streams with ICE Completed

If an agent generates an updated offer including a media stream that was previously established, and for which ICE checks are in the Completed state, the agent follows the procedures defined here.

The default destination for media (i.e., the values of the IP addresses and ports in the "m=" and "c=" lines used for that media stream) MUST be the local candidate from the highest-priority nominated pair in the valid list for each component.

The agent MUST include candidate attributes for candidates matching the default destination for each component of the media stream, and MUST NOT include any other candidates.

In addition, if the agent is controlling, it MUST include the a=remote-candidates attribute for each media stream whose check list is in the Completed state. The attribute contains the remote candidates from the highest-priority nominated pair in the valid list for each component of that media stream. It is needed to avoid a race condition whereby the controlling agent chooses its pairs, but the updated offer beats the connectivity checks to the controlled agent, which doesn't even know these pairs are valid, let alone selected. See Appendix B for elaboration on this race condition.

4.2.1.3. Procedures for Lite Implementations

4.2.1.3.1. Existing Media Streams with ICE Running

This section describes procedures for lite implementations for existing streams for which ICE is running.

A lite implementation MUST include all of its candidates for each component of each media stream in an a=candidate attribute in any subsequent offer. These candidates are formed identically to the procedures for initial offers, as described in section 5.2 of [ICE-BIS].

A lite implementation MUST NOT add additional host candidates in a subsequent offer. If an agent needs to offer additional candidates, it MUST restart ICE.

The username fragments, password, and implementation level MUST remain the same as used previously. If an agent needs to change one of these, it MUST restart ICE for that media stream.

4.2.1.3.2. Existing Media Streams with ICE Completed

If ICE has completed for a media stream, the default destination for that media stream MUST be set to the remote candidate of the candidate pair for that component in the valid list. For a lite implementation, there is always just a single candidate pair in the valid list for each component of a media stream. Additionally, the agent MUST include a candidate attribute for each default destination.

Additionally, if the agent is controlling (which only happens when both agents are lite), the agent MUST include the a=remote-candidates attribute for each media stream. The attribute contains the remote candidates from the candidate pairs in the valid list (one pair for each component of each media stream).

4.2.2. Receiving the Offer and Generating an Answer

4.2.2.1. Procedures for All Implementations

When receiving a subsequent offer within an existing session, an agent MUST reapply the verification procedures in Section 4.1.2.3 without regard to the results of verification from any previous offer/answer exchanges. Indeed, it is possible that a previous offer/answer exchange resulted in ICE not being used, but it is used as a consequence of a subsequent exchange.

4.2.2.1.1. Detecting ICE Restart

If the offer contained a change in the a=ice-ufrag or a=ice-pwd attributes compared to the previous SDP from the peer, it indicates that ICE is restarting for this media stream. If all media streams are restarting, then ICE is restarting overall.

If ICE is restarting for a media stream:

- The agent MUST change the a=ice-ufrag and a=ice-pwd attributes in the answer.
- The agent MAY change its implementation level in the answer.

An agent sets the rest of the fields in the SDP for this media stream as it would in an initial answer to this media stream (see Section 4.1.1.2). Consequently, the set of candidates MAY include some, none, or all of the previous candidates for that stream and MAY include a totally new set of candidates.

4.2.2.1.2. New Media Stream

If the offer contains a new media stream, the agent sets the fields in the answer as if it had received an initial offer containing that media stream (see Section 4.1.1.2). This will cause ICE processing to begin for this media stream.

4.2.2.1.3. Removed Media Stream

If an offer contains a media stream whose port is zero, the agent MUST NOT include any candidate attributes for that media stream in its answer and SHOULD NOT include any other ICE-related attributes defined in Section 5 for that media stream.

4.2.2.2. Procedures for Full Implementations

Unless the agent has detected an ICE restart from the offer, the username fragments, password, and implementation level MUST remain the same as used previously. If an agent needs to change one of these it MUST restart ICE for that media stream by generating an offer; ICE cannot be restarted in an answer.

Additional behaviors depend on the state of ICE processing for that media stream.

4.2.2.2.1. Existing Media Streams with ICE Running and no remote-candidates

If ICE is running for a media stream, and the offer for that media stream lacked the remote-candidates attribute, the rules for construction of the answer are identical to those for the offerer as described in Section 4.2.1.2.1.

4.2.2.2.2. Existing Media Streams with ICE Completed and no remote-candidates

If ICE is Completed for a media stream, and the offer for that media stream lacked the remote-candidates attribute, the rules for construction of the answer are identical to those for the offerer as described in Section 4.2.1.2.2, except that the answerer MUST NOT include the a=remote-candidates attribute in the answer.

4.2.2.2.3. Existing Media Streams and remote-candidates

A controlled agent will receive an offer with the a=remote-candidates attribute for a media stream when its peer has concluded ICE processing for that media stream. This attribute is present in the offer to deal with a race condition between the receipt of the offer, and the receipt of the Binding Response that tells the answerer the candidate that will be selected by ICE. See Appendix B for an explanation of this race condition. Consequently, processing of an offer with this attribute depends on the winner of the race.

The agent forms a candidate pair for each component of the media stream by:

- Setting the remote candidate equal to the offerer’s default destination for that component (e.g., the contents of the "m=" and "c=" lines for RTP, and the a=rtcp attribute for RTCP)

- Setting the local candidate equal to the transport address for that same component in the a=remote-candidates attribute in the offer.

The agent then sees if each of these candidate pairs is present in the valid list. If a particular pair is not in the valid list, the check has "lost" the race. Call such a pair a "losing pair".

The agent finds all the pairs in the check list whose remote candidates equal the remote candidate in the losing pair:

- If none of the pairs are In-Progress, and at least one is Failed, it is most likely that a network failure, such as a network partition or serious packet loss, has occurred. The agent SHOULD generate an answer for this media stream as if the remote-candidates attribute had not been present, and then restart ICE for this stream.

- If at least one of the pairs is In-Progress, the agent SHOULD wait for those checks to complete, and as each completes, redo the processing in this section until there are no losing pairs.

Once there are no losing pairs, the agent can generate the answer. It MUST set the default destination for media to the candidates in the remote-candidates attribute from the offer (each of which will now be the local candidate of a candidate pair in the valid list). It MUST include a candidate attribute in the answer for each candidate in the remote-candidates attribute in the offer.

4.2.2.3. Procedures for Lite Implementations

If the received offer contains the remote-candidates attribute for a media stream, the agent forms a candidate pair for each component of the media stream by:

- Setting the remote candidate equal to the offerer’s default destination for that component (e.g., the contents of the "m=" and "c=" lines for RTP, and the a=rtcp attribute for RTCP).

- Setting the local candidate equal to the transport address for that same component in the a=remote-candidates attribute in the offer.

It then places those candidates into the Valid list for the media stream. The state of ICE processing for that media stream is set to Completed.

Furthermore, if the agent believed it was controlling, but the offer contained the remote-candidates attribute, both agents believe they are controlling. In this case, both would have sent updated offers around the same time. However, the signaling protocol carrying the offer/answer exchanges will have resolved this glare condition, so that one agent is always the 'winner' by having its offer received before its peer has sent an offer. The winner takes the role of controlling, so that the loser (the answerer under consideration in this section) MUST change its role to controlled. Consequently, if the agent was going to send an updated offer since, based on the rules in section 8.2 of [ICE-BIS], it was controlling, it no longer needs to.

Besides the potential role change, change in the Valid list, and state changes, the construction of the answer is performed identically to the construction of an offer as described in Section 4.2.1.3.

4.2.3. Receiving the Answer for a Subsequent Offer

Some deployments of ICE include e.g. SDP-Modifying Signaling-only Back-to-Back User Agents (B2BUAs) [RFC7092] that modify the SDP body during the subsequent offer/answer exchange. With the B2BUA being ICE-unaware, a subsequent answer might be manipulated and might not include ICE candidates although the initial answer did.

An example of a situation where such an "unexpected" answer might be experienced appears when such a B2BUA introduces a media server during call hold using 3rd party call-control procedures. Omitting further details how this is done this could result in an answer being received at the holding UA that was constructed by the B2BUA. With the B2BUA being ICE-unaware, that answer would not include ICE candidates.

Receiving an answer without ICE attributes in this situation might be unexpected, but would not necessarily impair the user experience.

In addition to procedures for the expected answer, the following section advices on how to recover from the unexpected situation.

4.2.3.1. Procedures for All Implementations

When receiving an answer within an existing session for a subsequent offer as specified in Section 4.2.1.2.2, an agent MUST verify ICE support as specified in Section 4.1.3.1.

If ICE support is indicated in the SDP answer and the offer was a restart, the agent MUST perform ICE restart procedures as specified

in Section 4.2.4. If ICE support is no longer indicated in the SDP answer, the agent MUST fall-back to [RFC3264] procedures and SHOULD NOT drop the dialog just because of missing ICE support. If the agent sends a new offer later on, it SHOULD perform an ICE restart as specified in Section 4.2.1.1.1.

If ICE support is indicated in the SDP answer and ICE is running, the agent MUST continue ICE procedures as specified in Section 4.2.4.1.4. If ICE support is no longer indicated in the SDP answer, the agent MUST abort the ongoing ICE processing and fall-back to [RFC3264] procedures. The agent SHOULD NOT drop the dialog just because of missing ICE support. If the agent sends a new offer later on, it SHOULD perform an ICE restart as specified in Section 4.2.1.1.1.

If ICE support is indicated in the SDP answer and if ICE is completed and the answer conforms to Section 4.2.2.2.3, the agent MUST remain in the ICE Completed state. If ICE support is no longer indicated in the SDP answer, the agent MUST fall-back to [RFC3264] procedures and SHOULD NOT drop the dialog just because of this unexpected answer. Once the agent sends a new offer later on it MUST perform an ICE restart.

4.2.4. Updating the Check and Valid Lists

4.2.4.1. Procedures for Full Implementations

4.2.4.1.1. ICE Restarts

The agent MUST remember the highest-priority nominated pairs in the Valid list for each component of the media stream, called the previous selected pairs, prior to the restart. The agent will continue to send media using these pairs, as described in Section 7.1. Once these destinations are noted, the agent MUST flush the valid and check lists, and then recompute the check list and its states as described in section 6.1.2 of [ICE-BIS].

4.2.4.1.2. New Media Stream

If the offer/answer exchange added a new media stream, the agent MUST create a new check list for it (and an empty Valid list to start of course), as described in section 6.1.2 of [ICE-BIS].

4.2.4.1.3. Removed Media Stream

If the offer/answer exchange removed a media stream, or an answer rejected an offered media stream, an agent MUST flush the Valid list for that media stream. It MUST terminate any STUN transactions in

progress for that media stream. An agent MUST remove the check list for that media stream and cancel any pending ordinary checks for it.

4.2.4.1.4. ICE Continuing for Existing Media Stream

The valid list is not affected by an updated offer/answer exchange unless ICE is restarting.

If an agent is in the Running state for that media stream, the check list is updated (the check list is irrelevant if the state is completed). To do that, the agent recomputes the check list using the procedures described in section 6.1.2 of [ICE-BIS]. If a pair on the new check list was also on the previous check list, and its state was Waiting, In-Progress, Succeeded, or Failed, its state is copied over. Otherwise, its state is set to Frozen.

If none of the check lists are active (meaning that the pairs in each check list are Frozen), the full-mode agent follows steps in Section 6.1.2.6 of [ICE-BIS] to place appropriate candidates in the Waiting state to further continue ICE processing.

4.2.4.2. Procedures for Lite Implementations

If ICE is restarting for a media stream, the agent MUST start a new Valid list for that media stream. It MUST remember the pairs in the previous Valid list for each component of the media stream, called the previous selected pairs, and continue to send media there as described in Section 7.1. The state of ICE processing for each media stream MUST change to Running, and the state of ICE processing MUST change to Running.

5. Grammar

This specification defines eight new SDP attributes -- the "candidate", "remote-candidates", "ice-lite", "ice-mismatch", "ice-ufrag", "ice-pwd", "ice-pacing", and "ice-options" attributes. This section also provides non-normative examples of the attributes defined.

The syntax for the attributes follow Augmented BNF as defined in [RFC5234].

5.1. "candidate" Attribute

The candidate attribute is a media-level attribute only. It contains a transport address for a candidate that can be used for connectivity checks.

```
candidate-attribute   = "candidate" ":" foundation SP component-id SP
                        transport SP
                        priority SP
                        connection-address SP     ;from RFC 4566
                        port         ;port from RFC 4566
                        SP cand-type
                        [SP rel-addr]
                        [SP rel-port]
                        *(SP extension-att-name SP
                             extension-att-value)

foundation            = 1*32ice-char
component-id          = 1*5DIGIT
transport             = "UDP" / transport-extension
transport-extension   = token              ; from RFC 3261
priority              = 1*10DIGIT
cand-type             = "typ" SP candidate-types
candidate-types       = "host" / "srflx" / "prflx" / "relay" / token
rel-addr              = "raddr" SP connection-address
rel-port              = "rport" SP port
extension-att-name    = token
extension-att-value   = *VCHAR
ice-char              = ALPHA / DIGIT / "+" / "/"
```

This grammar encodes the primary information about a candidate: its IP address, port and transport protocol, and its properties: the foundation, component ID, priority, type, and related transport address:

<connection-address>: is taken from RFC 4566 [RFC4566]. It is the IP address of the candidate. When parsing this field, an agent can differentiate an IPv4 address and an IPv6 address by presence of a colon in its value -- the presence of a colon indicates IPv6. An agent MUST ignore candidate lines that include candidates with IP address versions that are not supported or recognized. An IP address SHOULD be used, but an FQDN MAY be used in place of an IP address. In that case, when receiving an offer or answer containing an FQDN in an a=candidate attribute, the FQDN is looked up in the DNS first using an AAAA record (assuming the agent supports IPv6), and if no result is found or the agent only supports IPv4, using an A record. The rules from section 6 of [RFC6724] is followed by fixing the source address to be one from the candidate pair to be matched against destination addresses reported by FQDN, in cases where the DNS query returns more than one IP address.

<port>: is also taken from RFC 4566 [RFC4566]. It is the port of the candidate.

<transport>: indicates the transport protocol for the candidate. This specification only defines UDP. However, extensibility is provided to allow for future transport protocols to be used with ICE, such as the Datagram Congestion Control Protocol (DCCP) [RFC4340].

<foundation>: is composed of 1 to 32 <ice-char>s. It is an identifier that is equivalent for two candidates that are of the same type, share the same base, and come from the same STUN server. The foundation is used to optimize ICE performance in the Frozen algorithm as described in section 6.1.2 of [ICE-BIS]

<component-id>: is a positive integer between 1 and 256 that identifies the specific component of the media stream for which this is a candidate. It MUST start at 1 and MUST increment by 1 for each component of a particular candidate. For media streams based on RTP, candidates for the actual RTP media MUST have a component ID of 1, and candidates for RTCP MUST have a component ID of 2. See section 14 in [ICE-BIS] for additional discussion on extending ICE to new media streams.

<priority>: is a positive integer between 1 and (2**31 - 1). The procedures for computing candidate's priority is described in section 5.1.2 of [ICE-BIS].

<cand-type>: encodes the type of candidate. This specification defines the values "host", "srflx", "prflx", and "relay" for host, server reflexive, peer reflexive, and relayed candidates, respectively. The set of candidate types is extensible for the future.

<rel-addr> and <rel-port>: convey transport addresses related to the candidate, useful for diagnostics and other purposes. <rel-addr> and <rel-port> MUST be present for server reflexive, peer reflexive, and relayed candidates. If a candidate is server or peer reflexive, <rel-addr> and <rel-port> are equal to the base for that server or peer reflexive candidate. If the candidate is relayed, <rel-addr> and <rel-port> are equal to the mapped address in the Allocate response that provided the client with that relayed candidate (see section Appendix B.3 of [ICE-BIS] for a discussion of its purpose). If the candidate is a host candidate, <rel-addr> and <rel-port> MUST be omitted.

In some cases, e.g., for privacy reasons, an agent may not want to reveal the related address and port. In this case the address MUST be set to "0.0.0.0" (for IPv4 candidates) or "::" (for IPv6 candidates) and the port to zero.

The candidate attribute can itself be extended. The grammar allows for new name/value pairs to be added at the end of the attribute. An implementation MUST ignore any name/value pairs it doesn't understand.

Example: SDP line for UDP server reflexive candidate attribute for the RTP compo
nent

a=candidate:2 1 UDP 1694498815 192.0.2.3 45664 typ
srflx raddr 10.0.1.1 rport 8998

5.2. "remote-candidates" Attribute

The syntax of the "remote-candidates" attribute is defined using Augmented BNF as defined in [RFC5234]. The remote-candidates attribute is a media-level attribute only.

```
remote-candidate-att = "remote-candidates:" remote-candidate
                         0*(SP remote-candidate)
remote-candidate = component-ID SP connection-address SP port
```

The attribute contains a connection-address and port for each component. The ordering of components is irrelevant. However, a value MUST be present for each component of a media stream. This attribute MUST be included in an offer by a controlling agent for a media stream that is Completed, and MUST NOT be included in any other case.

Example: Remote candidates SDP lines for the RTP and RTCP components:

```
a=remote-candidates:1 192.0.2.3 45664
a=remote-candidates:2 192.0.2.3 45665
```

5.3. "ice-lite" and "ice-mismatch" Attributes

The syntax of the "ice-lite" and "ice-mismatch" attributes, both of which are flags, is:

```
ice-lite              = "ice-lite"
ice-mismatch          = "ice-mismatch"
```

"ice-lite" is a session-level attribute only, and indicates that an agent is a lite implementation. "ice-mismatch" is a media-level attribute only, and when present in an answer, indicates that the offer arrived with a default destination for a media component that didn't have a corresponding candidate attribute.

5.4. "ice-ufrag" and "ice-pwd" Attributes

The "ice-ufrag" and "ice-pwd" attributes convey the username fragment and password used by ICE for message integrity. Their syntax is:

```
ice-pwd-att           = "ice-pwd:" password
ice-ufrag-att         = "ice-ufrag:" ufrag
password              = 22*256ice-char
ufrag                 = 4*256ice-char
```

The "ice-pwd" and "ice-ufrag" attributes can appear at either the session-level or media-level. When present in both, the value in the media-level takes precedence. Thus, the value at the session-level is effectively a default that applies to all media streams, unless overridden by a media-level value. Whether present at the session or media-level, there MUST be an ice-pwd and ice-ufrag attribute for each media stream. If two media streams have identical ice-ufrag's, they MUST have identical ice-pwd's.

The ice-ufrag and ice-pwd attributes MUST be chosen randomly at the beginning of a session. The ice-ufrag attribute MUST contain at least 24 bits of randomness, and the ice-pwd attribute MUST contain at least 128 bits of randomness. This means that the ice-ufrag attribute will be at least 4 characters long, and the ice-pwd at least 22 characters long, since the grammar for these attributes allows for 6 bits of information per character. The attributes MAY be longer than 4 and 22 characters, respectively, of course, up to 256 characters. The upper limit allows for buffer sizing in implementations. Its large upper limit allows for increased amounts of randomness to be added over time. For compatibility with the 512 character limitation for the STUN username attribute value and for bandwidth conservation considerations, the ice-ufrag attribute MUST NOT be longer than 32 characters when sending, but an implementation MUST accept up to 256 characters when receiving.

Example shows sample ice-ufrag and ice-pwd SDP lines:

```
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
```

5.5. "ice-pacing" Attribute

The "ice-pacing" attribute indicates the desired connectivity check pacing, in milliseconds, for this agent (see section 15 of [ICE-BIS]). The syntax is:

```
ice-pacing-att            = "ice-pacing:" pacing-value
pacing-value              = 1*10DIGIT
```

Example shows ice-pacing value of 5 ms:

a=ice-pacing:5

5.6. "ice-options" Attribute

The "ice-options" attribute is a session- and media-level attribute. It contains a series of tokens that identify the options supported by the agent. Its grammar is:

```
ice-options           = "ice-options:" ice-option-tag
                          0*(SP ice-option-tag)
ice-option-tag        = 1*ice-char
```

The existence of an ice-option in an offer indicates that a certain extension is supported by the agent and is willing to use it, if the peer agent also includes the same extension in the answer. There might be further extension specific negotiations needed between the agents that determine how the extensions gets used in a given session. The details of the negotiation procedures, if present, MUST be defined by the specification defining the extension.

Example shows 'rtp+ecn' ice-option SDP line from <<RFC6679>>:

a=ice-options:rtp+ecn

6. Keepalives

All the ICE agents MUST follow the procedures defined in section 11 of [ICE-BIS] for sending keepalives. The keepalives MUST be sent regardless of whether the media stream is currently inactive, sendonly, recvonly, or sendrecv, and regardless of the presence or value of the bandwidth attribute. An agent can determine that its peer supports ICE by the presence of a=candidate attributes for each media session.

7. Media Handling

7.1. Sending Media

The selected pair for a component of a media stream might not equal the default pair for that same component from the most recent offer/answer exchange. When this happens, the selected pair is used for media, not the default pair. When ICE first completes, if the selected pairs aren't a match for the default pairs, the controlling agent sends an updated offer/answer exchange to remedy this disparity. However, until that updated offer arrives, there will not

be a match. Furthermore, in very unusual cases, the default candidates in the updated offer/answer will not be a match.

7.1.1. Procedures for All Implementations

Section 12.1.3 of [ICE-BIS] defines procedures for sending media common across Full and Lite implementations.

7.2. Receiving Media

See section 12.2 of [ICE-BIS] for procedures on receiving media.

8. SIP Considerations

Note that ICE is not intended for NAT traversal for SIP, which is assumed to be provided via another mechanism [RFC5626].

When ICE is used with SIP, forking may result in a single offer generating a multiplicity of answers. In that case, ICE proceeds completely in parallel and independently for each answer, treating the combination of its offer and each answer as an independent offer/answer exchange, with its own set of local candidates, pairs, check lists, states, and so on.

Once ICE processing has reached the Completed state for all peers for media streams using those candidates, the agent SHOULD wait an additional three seconds, and then it MAY cease responding to checks or generating triggered checks on that candidate. It MAY free the candidate at that time. Freeing of server reflexive candidates is never explicit; it happens by lack of a keepalive. The three-second delay handles cases when aggressive nomination is used, and the selected pairs can quickly change after ICE has completed.

8.1. Latency Guidelines

ICE requires a series of STUN-based connectivity checks to take place between endpoints. These checks start from the answerer on generation of its answer, and start from the offerer when it receives the answer. These checks can take time to complete, and as such, the selection of messages to use with offers and answers can affect perceived user latency. Two latency figures are of particular interest. These are the post-pickup delay and the post-dial delay. The post-pickup delay refers to the time between when a user "answers the phone" and when any speech they utter can be delivered to the caller. The post-dial delay refers to the time between when a user enters the destination address for the user and ringback begins as a consequence of having successfully started alerting the called user agent.

Two cases can be considered -- one where the offer is present in the initial INVITE and one where it is in a response.

8.1.1. Offer in INVITE

To reduce post-dial delays, it is RECOMMENDED that the caller begin gathering candidates prior to actually sending its initial INVITE. This can be started upon user interface cues that a call is pending, such as activity on a keypad or the phone going off-hook.

On the receipt of the offer, the answerer SHOULD generate an answer in a provisional response once it has compelted candidate gathering. ICE requires that a provisional response with an SDP be transmitted reliably. This can be done through the existing Provisional Response Acknowledgment (PRACK) mechanism [RFC3262] or through an ICE specific optimization, wherein, the agent retransmits the provisional response with the exponential backoff timers described in [RFC3262]. Such retransmissions MUST cease on receipt of a STUN Binding request for one of the media streams signaled in that SDP or on transmission of the answer in a 2xx response. If no Binding request is received prior to the last retransmit, the agent does not consider the session terminated. For the ICE lite peers, the agent MUST cease retransmitting the 18x after sending it four times (ICE will actually work even if the peer never receives the 18x; however, experience has shown that sending it is important for middleboxes and firewall traversal).

It should be noted that the ICE specific optimization is very specific to provisional response carrying answers that start ICE processing and it is not a general technique for 1xx reliability. Also such an optimization SHOULD NOT be used if both agents support PRACK.

Despite the fact that the provisional response will be delivered reliably, the rules for when an agent can send an updated offer or answer do not change from those specified in [RFC3262]. Specifically, if the INVITE contained an offer, the same answer appears in all of the 1xx and in the 2xx response to the INVITE. Only after that 2xx has been sent can an updated offer/answer exchange occur.

Alternatively, an agent MAY delay sending an answer until the 200 OK; however, this results in a poor user experience and is NOT RECOMMENDED.

Once the answer has been sent, the agent SHOULD begin its connectivity checks. Once candidate pairs for each component of a

media stream enter the valid list, the answerer can begin sending media on that media stream.

However, prior to this point, any media that needs to be sent towards the caller (such as SIP early media [RFC3960]) MUST NOT be transmitted. For this reason, implementations SHOULD delay alerting the called party until candidates for each component of each media stream have entered the valid list. In the case of a PSTN gateway, this would mean that the setup message into the PSTN is delayed until this point. Doing this increases the post-dial delay, but has the effect of eliminating 'ghost rings'. Ghost rings are cases where the called party hears the phone ring, picks up, but hears nothing and cannot be heard. This technique works without requiring support for, or usage of, preconditions [RFC3312]. It also has the benefit of guaranteeing that not a single packet of media will get clipped, so that post-pickup delay is zero. If an agent chooses to delay local alerting in this way, it SHOULD generate a 180 response once alerting begins.

8.1.2. Offer in Response

In addition to uses where the offer is in an INVITE, and the answer is in the provisional and/or 200 OK response, ICE works with cases where the offer appears in the response. In such cases, which are common in third party call control [RFC3725], ICE agents SHOULD generate their offers in a reliable provisional response (which MUST utilize [RFC3262]), and not alert the user on receipt of the INVITE. The answer will arrive in a PRACK. This allows for ICE processing to take place prior to alerting, so that there is no post-pickup delay, at the expense of increased call setup delays. Once ICE completes, the callee can alert the user and then generate a 200 OK when they answer. The 200 OK would contain no SDP, since the offer/answer exchange has completed.

Alternatively, agents MAY place the offer in a 2xx instead (in which case the answer comes in the ACK). When this happens, the callee will alert the user on receipt of the INVITE, and the ICE exchanges will take place only after the user answers. This has the effect of reducing call setup delay, but can cause substantial post-pickup delays and media clipping.

8.2. SIP Option Tags and Media Feature Tags

[RFC5768] specifies a SIP option tag and media feature tag for usage with ICE. ICE implementations using SIP SHOULD support this specification, which uses a feature tag in registrations to facilitate interoperability through signaling intermediaries.

8.3. Interactions with Forking

ICE interacts very well with forking. Indeed, ICE fixes some of the problems associated with forking. Without ICE, when a call forks and the caller receives multiple incoming media streams, it cannot determine which media stream corresponds to which callee.

With ICE, this problem is resolved. The connectivity checks which occur prior to transmission of media carry username fragments, which in turn are correlated to a specific callee. Subsequent media packets that arrive on the same candidate pair as the connectivity check will be associated with that same callee. Thus, the caller can perform this correlation as long as it has received an answer.

8.4. Interactions with Preconditions

Quality of Service (QoS) preconditions, which are defined in [RFC3312] and [RFC4032], apply only to the transport addresses listed as the default targets for media in an offer/answer. If ICE changes the transport address where media is received, this change is reflected in an updated offer that changes the default destination for media to match ICE’s selection. As such, it appears like any other re-INVITE would, and is fully treated in RFCs 3312 and 4032, which apply without regard to the fact that the destination for media is changing due to ICE negotiations occurring "in the background".

Indeed, an agent SHOULD NOT indicate that QoS preconditions have been met until the checks have completed and selected the candidate pairs to be used for media.

ICE also has (purposeful) interactions with connectivity preconditions [RFC5898]. Those interactions are described there. Note that the procedures described in Section 8.1 describe their own type of "preconditions", albeit with less functionality than those provided by the explicit preconditions in [RFC5898].

8.5. Interactions with Third Party Call Control

ICE works with Flows I, III, and IV as described in [RFC3725]. Flow I works without the controller supporting or being aware of ICE. Flow IV will work as long as the controller passes along the ICE attributes without alteration. Flow II is fundamentally incompatible with ICE; each agent will believe itself to be the answerer and thus never generate a re-INVITE.

The flows for continued operation, as described in Section 7 of [RFC3725], require additional behavior of ICE implementations to support. In particular, if an agent receives a mid-dialog re-INVITE

that contains no offer, it MUST restart ICE for each media stream and go through the process of gathering new candidates. Furthermore, that list of candidates SHOULD include the ones currently being used for media.

9. Relationship with ANAT

[RFC4091], the Alternative Network Address Types (ANAT) Semantics for the SDP grouping framework, and [RFC4092], its usage with SIP, define a mechanism for indicating that an agent can support both IPv4 and IPv6 for a media stream, and it does so by including two "m=" lines, one for v4 and one for v6. This is similar to ICE, which allows for an agent to indicate multiple transport addresses using the candidate attribute. However, ANAT relies on static selection to pick between choices, rather than a dynamic connectivity check used by ICE.

It is RECOMMENDED that ICE be used in realizing the dual-stack use-cases in agents that support ICE.

10. Setting Ta and RTO for RTP Media Streams

During the gathering phase of ICE (section 5.1.1 [ICE-BIS]) and while ICE is performing connectivity checks (section 7 [ICE-BIS]), an agent sends STUN and TURN transactions. These transactions are paced at a rate of one every Ta milliseconds, and utilize a specific RTO. See Section 15 of [ICE-BIS] for details on how the values of Ta and RTO are computed with a real-time media stream of known maximum bandwidth to rate-control the ICE exchanges.

11. Security Considerations

11.1. Attacks on the Offer/Answer Exchanges

An attacker that can modify or disrupt the offer/answer exchanges themselves can readily launch a variety of attacks with ICE. They could direct media to a target of a DoS attack, they could insert themselves into the media stream, and so on. These are similar to the general security considerations for offer/answer exchanges, and the security considerations in [RFC3264] apply. These require techniques for message integrity and encryption for offers and answers, which are satisfied by the TLS mechanism [RFC3261] when SIP is used. As such, the usage of TLS with ICE is RECOMMENDED.

11.2. Insider Attacks

In addition to attacks where the attacker is a third party trying to insert fake offers, answers, or STUN messages, there are several

attacks possible with ICE when the attacker is an authenticated and valid participant in the ICE exchange.

11.2.1. The Voice Hammer Attack

The voice hammer attack is an amplification attack. In this attack, the attacker initiates sessions to other agents, and maliciously includes the IP address and port of a DoS target as the destination for media traffic signaled in the SDP. This causes substantial amplification; a single offer/answer exchange can create a continuing flood of media packets, possibly at high rates (consider video sources). This attack is not specific to ICE, but ICE can help provide remediation.

Specifically, if ICE is used, the agent receiving the malicious SDP will first perform connectivity checks to the target of media before sending media there. If this target is a third-party host, the checks will not succeed, and media is never sent.

Unfortunately, ICE doesn’t help if it’s not used, in which case an attacker could simply send the offer without the ICE parameters. However, in environments where the set of clients is known, and is limited to ones that support ICE, the server can reject any offers or answers that don’t indicate ICE support.

User Agents that are not willing to receive non-ICE answers MUST include an "ice" Option Tag in the Require Header Field in their offer. Clients that rejects non-ICE offers SHOULD use a 421 response code, together with an Option Tag "ice" in the Require Header Field in the response.

11.2.2. Interactions with Application Layer Gateways and SIP

Application Layer Gateways (ALGs) are functions present in a Network Address Translation (NAT) device that inspect the contents of packets and modify them, in order to facilitate NAT traversal for application protocols. Session Border Controllers (SBCs) are close cousins of ALGs, but are less transparent since they actually exist as application-layer SIP intermediaries. ICE has interactions with SBCs and ALGs.

If an ALG is SIP aware but not ICE aware, ICE will work through it as long as the ALG correctly modifies the SDP. A correct ALG implementation behaves as follows:

- The ALG does not modify the "m=" and "c=" lines or the rtcp attribute if they contain external addresses.

- o If the "m=" and "c=" lines contain internal addresses, the modification depends on the state of the ALG:
  - * If the ALG already has a binding established that maps an external port to an internal IP address and port matching the values in the "m=" and "c=" lines or rtcp attribute, the ALG uses that binding instead of creating a new one.
  - * If the ALG does not already have a binding, it creates a new one and modifies the SDP, rewriting the "m=" and "c=" lines and rtcp attribute.

Unfortunately, many ALGs are known to work poorly in these corner cases. ICE does not try to work around broken ALGs, as this is outside the scope of its functionality. ICE can help diagnose these conditions, which often show up as a mismatch between the set of candidates and the "m=" and "c=" lines and rtcp attributes. The ice-mismatch attribute is used for this purpose.

ICE works best through ALGs when the signaling is run over TLS. This prevents the ALG from manipulating the SDP messages and interfering with ICE operation. Implementations that are expected to be deployed behind ALGs SHOULD provide for TLS transport of the SDP.

If an SBC is SIP aware but not ICE aware, the result depends on the behavior of the SBC. If it is acting as a proper Back-to-Back User Agent (B2BUA), the SBC will remove any SDP attributes it doesn't understand, including the ICE attributes. Consequently, the call will appear to both endpoints as if the other side doesn't support ICE. This will result in ICE being disabled, and media flowing through the SBC, if the SBC has requested it. If, however, the SBC passes the ICE attributes without modification, yet modifies the default destination for media (contained in the "m=" and "c=" lines and rtcp attribute), this will be detected as an ICE mismatch, and ICE processing is aborted for the call. It is outside of the scope of ICE for it to act as a tool for "working around" SBCs. If one is present, ICE will not be used and the SBC techniques take precedence.

12. IANA Considerations

12.1. SDP Attributes

The original ICE specification defined seven new SDP attributes per the procedures of Section 8.2.4 of [RFC4566]. The registration information is reproduced here.

12.1.1. candidate Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: candidate

Long Form: candidate

Type of Attribute: media-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides one of many possible candidate addresses for communication. These addresses are validated with an end-to-end connectivity check using Session Traversal Utilities for NAT (STUN).

Appropriate Values: See Section 5 of RFC XXXX.

12.1.2. remote-candidates Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: remote-candidates

Long Form: remote-candidates

Type of Attribute: media-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides the identity of the remote candidates that the offerer wishes the answerer to use in its answer.

Appropriate Values: See Section 5 of RFC XXXX.

12.1.3. ice-lite Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: ice-lite

Long Form: ice-lite

Type of Attribute: session-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and indicates that an agent has the minimum functionality required to support ICE inter-operation with a peer that has a full implementation.

Appropriate Values: See Section 5 of RFC XXXX.

12.1.4. ice-mismatch Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: ice-mismatch

Long Form: ice-mismatch

Type of Attribute: session-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and indicates that an agent is ICE capable, but did not proceed with ICE due to a mismatch of candidates with the default destination for media signaled in the SDP.

Appropriate Values: See Section 5 of RFC XXXX.

12.1.5. ice-pwd Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: ice-pwd

Long Form: ice-pwd

Type of Attribute: session- or media-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides the password used to protect STUN connectivity checks.

Appropriate Values: See Section 5 of RFC XXXX.

12.1.6. ice-ufrag Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: ice-ufrag

Long Form: ice-ufrag

Type of Attribute: session- or media-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides the fragments used to construct the username in STUN connectivity checks.

Appropriate Values: See Section 5 of RFC XXXX.

12.1.7. ice-pacing Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: ice-pacing

Long Form: ice-pacing

Type of Attribute: session-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE) to indicate desired connectivity check pacing values.

Appropriate Values: See Section 5 of RFC XXXX.

12.1.8. ice-options Attribute

Contact Name: Jonathan Rosenberg, jdrosen@jdrosen.net.

Attribute Name: ice-options

Long Form: ice-options

Type of Attribute: session- or media-level

Charset Considerations: The attribute is not subject to the charset attribute.

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and indicates the ICE options or extensions used by the agent.

Appropriate Values: See Section 5 of RFC XXXX.

12.2. Interactive Connectivity Establishment (ICE) Options Registry

IANA maintains a registry for ice-options identifiers under the Specification Required policy as defined in "Guidelines for Writing an IANA Considerations Section in RFCs" [RFC5226].

ICE options are of unlimited length according to the syntax in Section 5.6; however, they are RECOMMENDED to be no longer than 20 characters. This is to reduce message sizes and allow for efficient parsing.

In [RFC5245] ICE options could only be defined at the session level. ICE options can now also be defined at the media level. This can be used when aggregating between different ICE agents in the same endpoint, but future options may require to be defined at the media-level. To ensure compatibility with legacy implementation, the media-level ICE options MUST be aggregated into a session-level ICE option. Because aggregation rules depend on the specifics of each option, all new ICE options MUST also define in their specification how the media-level ICE option values are aggregated to generate the value of the session-level ICE option.

[RFC6679] defines the "rtp+ecn" ICE option. The aggregation rule for this ICE option is that if all aggregated media using ICE contain a media-level "rtp+ecn" ICE option then an "rtp+ecn" ICE option MUST be inserted at the session-level. If one of the media does not contain the option, then it MUST NOT be inserted at the session-level.

Section 10 of [ICE-BIS] defines "ice2" ICE option. Since "ice2" is a session level ICE option, no aggregation rules apply.

A registration request MUST include the following information:

- The ICE option identifier to be registered
- Name, Email, and Address of a contact person for the registration

- Organization or individuals having the change control
- Short description of the ICE extension to which the option relates
- Reference(s) to the specification defining the ICE option and the related extensions

13. Acknowledgments

A large part of the text in this document was taken from [RFC5245], authored by Jonathan Rosenberg.

Some of the text in this document was taken from [RFC6336], authored by Magnus Westerlund and Colin Perkins.

Thanks to Thomas Stach for the text in Section 4.2.3, Roman Shpount for suggesting RTCP candidate handling in Section 4.1.1.2 and Simon Perreault for advising on IPV6 address selection when candidate-address includes FQDN.

Thanks to following experts for their reviews and constructive feedback: Christer Holmberg, Adam Roach and the MMUSIC WG.

14. References

14.1. Normative References

[ICE-BIS] Keranen, A. and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", draft-ietf-ice-rfc5245bis-00 (work in progress), March 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC3262] Rosenberg, J. and H. Schulzrinne, "Reliability of Provisional Responses in Session Initiation Protocol (SIP)", RFC 3262, DOI 10.17487/RFC3262, June 2002, <http://www.rfc-editor.org/info/rfc3262>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <http://www.rfc-editor.org/info/rfc3264>.

[RFC3312] Camarillo, G., Ed., Marshall, W., Ed., and J. Rosenberg, "Integration of Resource Management and Session Initiation Protocol (SIP)", RFC 3312, DOI 10.17487/RFC3312, October 2002, <http://www.rfc-editor.org/info/rfc3312>.

[RFC3556] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, DOI 10.17487/RFC3556, July 2003, <http://www.rfc-editor.org/info/rfc3556>.

[RFC3605] Huitema, C., "Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)", RFC 3605, DOI 10.17487/RFC3605, October 2003, <http://www.rfc-editor.org/info/rfc3605>.

[RFC4032] Camarillo, G. and P. Kyzivat, "Update to the Session Initiation Protocol (SIP) Preconditions Framework", RFC 4032, DOI 10.17487/RFC4032, March 2005, <http://www.rfc-editor.org/info/rfc4032>.

[RFC4091] Camarillo, G. and J. Rosenberg, "The Alternative Network Address Types (ANAT) Semantics for the Session Description Protocol (SDP) Grouping Framework", RFC 4091, June 2005, <http://www.rfc-editor.org/info/rfc4091>.

[RFC4092] Camarillo, G. and J. Rosenberg, "Usage of the Session Description Protocol (SDP) Alternative Network Address Types (ANAT) Semantics in the Session Initiation Protocol (SIP)", RFC 4092, June 2005, <http://www.rfc-editor.org/info/rfc4092>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <http://www.rfc-editor.org/info/rfc4566>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, DOI 10.17487/RFC5245, April 2010, <http://www.rfc-editor.org/info/rfc5245>.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <http://www.rfc-editor.org/info/rfc5389>.

[RFC5768] Rosenberg, J., "Indicating Support for Interactive Connectivity Establishment (ICE) in the Session Initiation Protocol (SIP)", RFC 5768, DOI 10.17487/RFC5768, April 2010, <http://www.rfc-editor.org/info/rfc5768>.

[RFC6336] Westerlund, M. and C. Perkins, "IANA Registry for Interactive Connectivity Establishment (ICE) Options", RFC 6336, April 2010, <http://www.rfc-editor.org/info/rfc6336>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <http://www.rfc-editor.org/info/rfc6679>.

[RFC6724] Thaler, D., Draves, R., Matsumoto, A., and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)", RFC 6724, September 2012, <http://www.rfc-editor.org/info/rfc6724>.

[RFC7092] Kaplan, H. and V. Pascual, "A Taxonomy of Session Initiation Protocol (SIP) Back-to-Back User Agents", RFC 7092, DOI 10.17487/RFC7092, December 2013, <http://www.rfc-editor.org/info/rfc7092>.

[RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", RFC 7656, DOI 10.17487/RFC7656, November 2015, <http://www.rfc-editor.org/info/rfc7656>.

14.2. Informative References

[RFC3725] Rosenberg, J., Peterson, J., Schulzrinne, H., and G. Camarillo, "Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)", BCP 85, RFC 3725, DOI 10.17487/RFC3725, April 2004, <http://www.rfc-editor.org/info/rfc3725>.

[RFC3960] Camarillo, G. and H. Schulzrinne, "Early Media and Ringing Tone Generation in the Session Initiation Protocol (SIP)", RFC 3960, DOI 10.17487/RFC3960, December 2004, <http://www.rfc-editor.org/info/rfc3960>.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, DOI 10.17487/RFC4340, March 2006, <http://www.rfc-editor.org/info/rfc4340>.

[RFC5626] Jennings, C., Ed., Mahy, R., Ed., and F. Audet, Ed., "Managing Client-Initiated Connections in the Session Initiation Protocol (SIP)", RFC 5626, DOI 10.17487/RFC5626, October 2009, <http://www.rfc-editor.org/info/rfc5626>.

[RFC5898] Andreasen, F., Camarillo, G., Oran, D., and D. Wing, "Connectivity Preconditions for Session Description Protocol (SDP) Media Streams", RFC 5898, DOI 10.17487/RFC5898, July 2010, <http://www.rfc-editor.org/info/rfc5898>.

Appendix A. Examples

For the example shown in section 16 of [ICE-BIS] the resulting offer (message 5) encoded in SDP looks like:

```
   v=0
   o=jdoe 2890844526 2890842807 IN IP6 $L-PRIV-1.IP
   s=
   c=IN IP6 $NAT-PUB-1.IP
   t=0 0
   a=ice-pwd:asd88fgpdd777uzjYhagZg
   a=ice-ufrag:8hhY
   m=audio $NAT-PUB-1.PORT RTP/AVP 0
   b=RS:0
   b=RR:0
   a=rtpmap:0 PCMU/8000
   a=candidate:1 1 UDP 2130706431 $L-PRIV-1.IP $L-PRIV-1.PORT typ host
   a=candidate:2 1 UDP 1694498815 $NAT-PUB-1.IP $NAT-PUB-1.PORT typ
    srflx raddr $L-PRIV-1.IP rport $L-PRIV-1.PORT
```

The offer, with the variables replaced with their values, will look like (lines folded for clarity):

```
v=0
o=jdoe 2890844526 2890842807 IN IP6 fe80::6676:baff:fe9c:ee4a
s=
c=IN IP6 2001:420:c0e0:1005::61
t=0 0
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
m=audio 45664 RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 fe80::6676:baff:fe9c:ee4a 8998 typ host
a=candidate:2 1 UDP 1694498815 2001:420:c0e0:1005::61 45664 typ srflx raddr
 fe80::6676:baff:fe9c:ee4a rport 8998
```

The resulting answer looks like:

```
   v=0
   o=bob 2808844564 2808844564 IN IP4 $R-PUB-1.IP
   s=
   c=IN IP4 $R-PUB-1.IP
   t=0 0
   a=ice-pwd:YH75Fviy6338Vbrhrlp8Yh
   a=ice-ufrag:9uB6
   m=audio $R-PUB-1.PORT RTP/AVP 0
   b=RS:0
   b=RR:0
   a=rtpmap:0 PCMU/8000
   a=candidate:1 1 UDP 2130706431 $R-PUB-1.IP $R-PUB-1.PORT typ host
```

With the variables filled in:

```
v=0
o=bob 2808844564 2808844564 IN IP4 192.0.2.1
s=
c=IN IP4 192.0.2.1
t=0 0
a=ice-pwd:YH75Fviy6338Vbrhrlp8Yh
a=ice-ufrag:9uB6
m=audio 3478 RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 192.0.2.1 3478 typ host
```

Appendix B. The remote-candidates Attribute

The a=remote-candidates attribute exists to eliminate a race condition between the updated offer and the response to the STUN Binding request that moved a candidate into the Valid list. This race condition is shown in Figure 1. On receipt of message 4, agent L adds a candidate pair to the valid list. If there was only a single media stream with a single component, agent L could now send an updated offer. However, the check from agent R has not yet generated a response, and agent R receives the updated offer (message 7) before getting the response (message 9). Thus, it does not yet know that this particular pair is valid. To eliminate this condition, the actual candidates at R that were selected by the offerer (the remote candidates) are included in the offer itself, and the answerer delays its answer until those pairs validate.

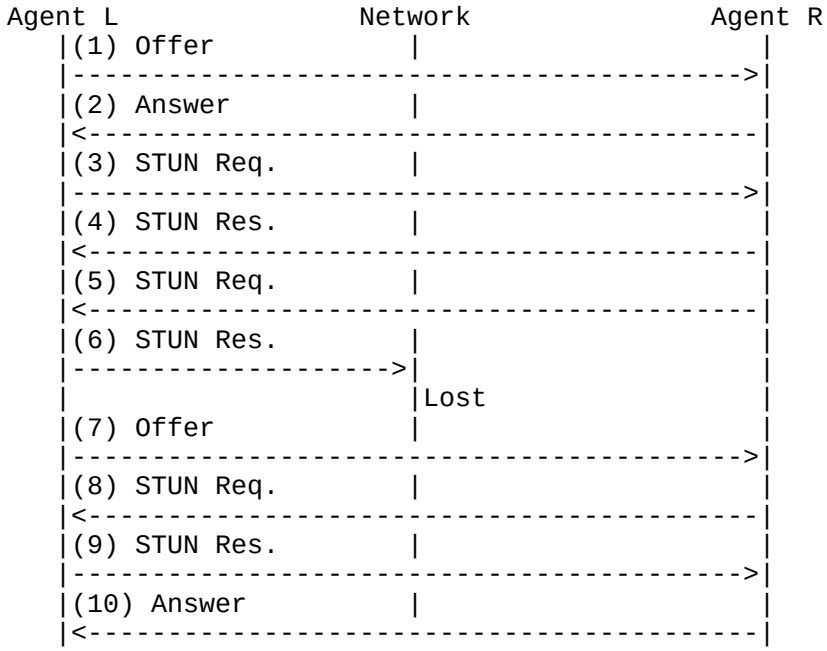

Figure 1: Race Condition Flow

Appendix C. Why Is the Conflict Resolution Mechanism Needed?

When ICE runs between two peers, one agent acts as controlled, and the other as controlling. Rules are defined as a function of implementation type and offerer/answerer to determine who is controlling and who is controlled. However, the specification mentions that, in some cases, both sides might believe they are controlling, or both sides might believe they are controlled. How can this happen?

The condition when both agents believe they are controlled shows up in third party call control cases. Consider the following flow:

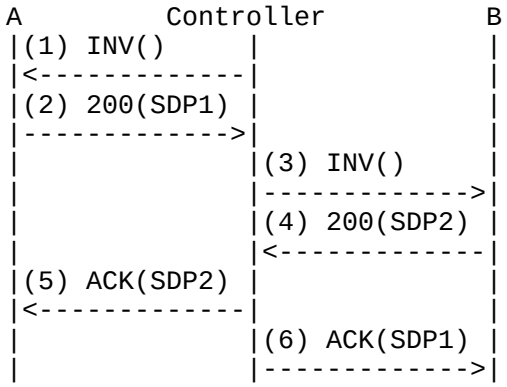

Figure 2: Role Conflict Flow

This flow is a variation on flow III of RFC 3725 [RFC3725]. In fact, it works better than flow III since it produces fewer messages. In this flow, the controller sends an offerless INVITE to agent A, which responds with its offer, SDP1. The agent then sends an offerless INVITE to agent B, which it responds to with its offer, SDP2. The controller then uses the offer from each agent to generate the answers. When this flow is used, ICE will run between agents A and B, but both will believe they are in the controlling role. With the role conflict resolution procedures, this flow will function properly when ICE is used.

At this time, there are no documented flows that can result in the case where both agents believe they are controlled. However, the conflict resolution procedures allow for this case, should a flow arise that would fit into this category.

Appendix D. Why Send an Updated Offer?

Section 11.1 describes rules for sending media. Both agents can send media once ICE checks complete, without waiting for an updated offer. Indeed, the only purpose of the updated offer is to "correct" the SDP so that the default destination for media matches where media is being sent based on ICE procedures (which will be the highest-priority nominated candidate pair).

This begs the question -- why is the updated offer/answer exchange needed at all? Indeed, in a pure offer/answer environment, it would not be. The offerer and answerer will agree on the candidates to use through ICE, and then can begin using them. As far as the agents themselves are concerned, the updated offer/answer provides no new information. However, in practice, numerous components along the signaling path look at the SDP information. These include entities

performing off-path QoS reservations, NAT traversal components such as ALGs and Session Border Controllers (SBCs), and diagnostic tools that passively monitor the network. For these tools to continue to function without change, the core property of SDP -- that the existing, pre-ICE definitions of the addresses used for media -- the "m=" and "c=" lines and the rtcp attribute -- must be retained. For this reason, an updated offer must be sent.

Authors' Addresses

Marc Petit-Huguenin
Impedance Mismatch

Email: marc@petit-huguenin.org

Ari Keranen
Ericsson
Jorvas 02420
Finland

Email: ari.keranen@ericsson.com

Suhas Nandakumar
Cisco Systems
707 Tasman Dr
Milpitas, CA 95035
USA

Email: snandaku@cisco.com

MMUSIC
Internet-Draft
Obsoletes: 5245 (if approved)
Intended status: Standards Track
Expires: February 14, 2020

M. Petit-Huguenin
Impedance Mismatch
S. Nandakumar
Cisco Systems
C. Holmberg
A. Keranen
Ericsson
R. Shpount
TurboBridge

August 13, 2019

Session Description Protocol (SDP) Offer/Answer procedures for Interactive Connectivity Establishment (ICE) draft-ietf-mmusic-ice-sip-sdp-39

Abstract

This document describes Session Description Protocol (SDP) Offer/Answer procedures for carrying out Interactive Connectivity Establishment (ICE) between the agents.

This document obsoletes RFC 5245.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 14, 2020.

Table of Contents

1. Introduction

This document describes how Interactive Connectivity Establishment (ICE) is used with Session Description Protocol (SDP) offer/answer [RFC3264]. The ICE specification [RFC8445] describes procedures that are common to all usages of ICE and this document gives the additional details needed to use ICE with SDP offer/answer.

This document obsoletes RFC 5245.

NOTE: Previously both the common ICE procedures, and the SDP offer/answer specific details, were described in[RFC5245]. [RFC8445] obsoleted [RFC5245], and the SDP offer/answer specific details were removed from the document. Section 12 describes the changes to the SDP offer/answer specific details specified in this document.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

Readers should be familiar with the terminology defined in [RFC3264], in [RFC8445] and the following:

Default Destination/Candidate: The default destination for a component of a data stream is the transport address that would be used by an agent that is not ICE aware. A default candidate for a component is one whose transport address matches the default destination for that component. For the RTP component, the default connection address is in the "c=" line of the SDP, and the port and transport protocol are in the "m=" line. For the RTCP component, the address and port are indicated using the "a=rtcp" attribute defined in [RFC3605], if present; otherwise, the RTCP component address is the same as the address of the RTP component, and its port is one greater than the port of the RTP component.

4. SDP Offer/Answer Procedures

4.1. Introduction

[RFC8445] defines ICE candidate exchange as the process for ICE agents (Initiator and Responder) to exchange their candidate information required for ICE processing at the agents. For the purposes of this specification, the candidate exchange process corresponds to the [RFC3264] Offer/Answer protocol and the terms "offerer" and "answerer" correspond to the initiator and responder roles from [RFC8445] respectively.

Once the initiating agent has gathered, pruned, and prioritized its set of candidates [RFC8445], the candidate exchange with the peer agent begins.

4.2. Generic Procedures

4.2.1. Encoding

Section 5 provides detailed rules for constructing various SDP attributes defined in this specification.

4.2.1.1. Data Streams

Each data stream [RFC8445] is represented by an SDP media description ("m=" section).

4.2.1.2. Candidates

Within an "m=" section, each candidate (including the default candidate) associated with the data stream is represented by an SDP candidate attribute.

Prior to nomination, the "c=" line associated with an "m=" section contains the connection address of the default candidate, while the "m=" line contains the port and transport protocol of the default candidate for that "m=" section.

After nomination, the "c=" line for a given "m=" section contains the connection address of the nominated candidate (the local candidate of the nominated candidate pair) and the "m=" line contains the port and transport protocol corresponding to the nominated candidate for that "m=" section.

4.2.1.3. Username and Password

The ICE username is represented by an SDP ice-ufrag attribute and the ICE password is represented by an SDP ice-pwd attribute.

4.2.1.4. Lite Implementations

An ICE lite implementation [RFC8445] MUST include an SDP ice-lite attribute. A full implementation MUST NOT include that attribute.

4.2.1.5. ICE Extensions

An agent uses the SDP ice-options attribute to indicate support of ICE extensions.

An agent compliant to this specification MUST include an SDP ice-options attribute with an "ice2" attribute value [RFC8445]. If an agent receives an SDP offer or answer that indicates ICE support, but that does not contain an SDP ice-options attribute with an "ice2"

attribute value, the agent can assume that the peer is compliant to [RFC5245].

4.2.1.6. Inactive and Disabled Data Streams

If an "m=" section is marked as inactive [RFC4566], or has a bandwidth value of zero [RFC4566], the agent MUST still include ICE-related SDP attributes.

If the port value associated with an "m=" section is set to zero (implying a disabled stream) as defined in section 8.2 of [RFC3264], the agent SHOULD NOT include ICE-related SDP candidate attributes in that "m=" section, unless an SDP extension specifying otherwise is used.

4.2.2. RTP/RTCP Considerations

If an agent utilizes both RTP and RTCP, and separate ports are used for RTP and RTCP, the agent MUST include SDP candidate attributes for both the RTP and RTCP components.

The agent includes an SDP rtcp attribute following the procedures in [RFC3605]. Hence, in the cases where the RTCP port value is one higher than the RTP port value and the RTCP component address the same as the address of the RTP component, the SDP rtcp attribute might be omitted.

NOTE: [RFC5245] required that an agent always includes the SDP rtcp attribute, even if the RTCP port value was one higher than the RTP port value. This specification aligns the rtcp attribute procedures with [RFC3605].

If the agent does not utilize RTCP, it indicates that by including b=RS:0 and b=RR:0 SDP attributes, as described in [RFC3556].

4.2.3. Determining Role

The offerer acts as the Initiating agent. The answerer acts as the Responding agent. The ICE roles (controlling and controlled) are determined using the procedures in [RFC8445].

4.2.4. STUN Considerations

Once an agent has provided its local candidates to its peer in an SDP offer or answer, the agent MUST be prepared to receive STUN connectivity check Binding requests on those candidates.

4.2.5. Verifying ICE Support Procedures

An ICE agent is considered to indicate support of ICE by including at least the SDP ice-pwd and ice-ufrag attributes in an offer or answer. An ICE agent compliant with this specification MUST also include an SDP ice-options attribute with an "ice2" attribute value.

The agents will proceed with the ICE procedures defined in [RFC8445] and this specification if, for each data stream in the SDP it received, the default destination for each component of that data stream appears in a candidate attribute. For example, in the case of RTP, the connection address, port, and transport protocol in the "c=" and "m=" lines, respectively, appear in a candidate attribute and the value in the rtcp attribute appears in a candidate attribute.

This specification provides no guidance on how an agent should proceed in the cases where the above condition is not met with the few exceptions noted below:

1. The presence of certain application layer gateways might modify the transport address information as described in Section 8. The behavior of the responding agent in such a situation is implementation dependent. Informally, the responding agent might consider the mismatched transport address information as a plausible new candidate learnt from the peer and continue its ICE processing with that transport address included. Alternatively, the responding agent MAY include an "a=ice-mismatch" attribute in its answer for such data streams. If an agent chooses to include an "a=ice-mismatch" attribute in its answer for a data stream, then it MUST also omit "a=candidate" attributes, MUST terminate the usage of ICE procedures and [RFC3264] procedures MUST be used instead for this data stream.

2. The transport address from the peer for the default destination is set to IPv4/IPv6 address values "0.0.0.0"/"::" and port value of "9". This MUST NOT be considered as a ICE failure by the peer agent and the ICE processing MUST continue as usual.

3. In some cases, the controlling/initiator agent may receive the SDP answer that may omit "a=candidate" attributes for the data stream, and instead include a media level "a=ice-mismatch" attribute. This signals to the offerer that the answerer supports ICE, but that ICE processing was not used for this data stream. In this case, ICE processing MUST be terminated for this data stream and [RFC3264] procedures MUST be followed instead.

4. The transport address from the peer for the default destination is an FQDN. Regardless of the procedures used to resolve FQDN or

the resolution result, this MUST NOT be considered as a ICE failure by the peer agent and the ICE processing MUST continue as usual.

4.2.6. SDP Example

The following is an example SDP message that includes ICE attributes (lines folded for readability):

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 203.0.113.141
s=
c=IN IP4 192.0.2.3
t=0 0
a=ice-options:ice2
a=ice-pacing:50
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
m=audio 45664 RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 203.0.113.141 8998 typ host
a=candidate:2 1 UDP 1694498815 192.0.2.3 45664 typ srflx raddr
 203.0.113.141 rport 8998
```

4.3. Initial Offer/Answer Exchange

4.3.1. Sending the Initial Offer

When an offerer generates the initial offer, in each "m=" section it MUST include SDP candidate attributes for each available candidate associated with the "m=" section. In addition, the offerer MUST include an SDP ice-ufrag attribute, an SDP ice-pwd attribute and an SDP ice-options attribute with an "ice2" attribute value in the offer. If the offerer is a full ICE implementation, it SHOULD include an ice-pacing attribute in the offer (if not included, the default value will apply). A lite ICE implementation MUST NOT included the ice-pacing attribute in the offer (as it will not perform connectivity checks).

It is valid for an offer "m=" line to include no SDP candidate attributes and with default destination set to the IP address values "0.0.0.0"/"::" and port value of "9". This implies that the offering agent is only going to use peer reflexive candidates or that additional candidates would be provided in subsequent signaling messages.

Note: Within the scope of this document, "Initial Offer" refers to the first SDP offer that is sent in order to negotiate usage of ICE. It might, or might not, be the initial SDP offer of the SDP session.

Note: The procedures in this document only consider "m=" sections associated with data streams where ICE is used.

4.3.2. Sending the Initial Answer

When an answerer receives an initial offer that indicates that the offerer supports ICE, and if the answerer accepts the offer and the usage of ICE, in each "m=" section within the answer, it MUST include SDP candidate attributes for each available candidate associated with the "m=" section. In addition, the answerer MUST include an SDP ice-ufrag attribute, an SDP ice-pwd attribute and an SDP ice-options attribute with an "ice2" attribute value in the answer. If the answerer is a full ICE implementation, it SHOULD include an ice-pacing attribute in the answerer (if not included, the default value will apply). A lite ICE implementation MUST NOT included the ice-pacing attribute in the answer (as it will not perform connectivity chekcks).

In each "m=" line, the answerer MUST use the same transport protocol as was used in the offer "m=" line. If none of the candidates in the "m=" line in the answer use the same transport protocol as indicated in the offer "m=" line, then, in order to avoid ICE mismatch, the default destination MUST be set to IP address values "0.0.0.0"/"::" and port value of "9".

It is also valid for an answer "m=" line to include no SDP candidate attributes and with default destination set to the IP address values "0.0.0.0"/"::" and port value of "9". This implies that the answering agent is only going to use peer reflexive candidates or that additional candidates would be provided in subsequent signaling messages.

Once the answerer has sent the answer, it can start performing connectivity checks towards the peer candidates that were provided in the offer.

If the offer does not indicate support of ICE Section 4.2.5, the answerer MUST NOT accept the usage of ICE. If the answerer still accepts the offer, the answerer MUST NOT include any ICE-related SDP attributes in the answer. Instead the answerer will generate the answer according to normal offer/answer procedures [RFC3264].

If the answerer detects a possibility of an ICE mismatch, procedures described in Section 4.2.5 are followed.

4.3.3. Receiving the Initial Answer

When an offerer receives an initial answer that indicates that the answerer supports ICE, it can start performing connectivity checks towards the peer candidates that were provided in the answer.

If the answer does not indicate that the answerer supports ICE, or if the answerer included "a=ice-mismatch" attributes for all the active data streams in the answer, the offerer MUST terminate the usage of ICE for the entire session and [RFC3264] procedures MUST be followed instead.

On the other hand, if the answer indicates support for ICE but includes "a=ice-mismatch" in certain active data streams, then the offerer MUST terminate the usage of ICE procedures and [RFC3264] procedures MUST be used instead for only these data streams. Also, ICE procedures MUST be used for data streams where an "a=ice-mismatch" attribute was not included.

If the offerer detects an ICE mismatch for one or more data streams in the answer, as described in Section 4.2.5, the offerer MUST terminate the usage of ICE for the entire session. The subsequent actions taken by the offerer are implementation dependent and are out of the scope of this specification.

4.3.4. Concluding ICE

Once the agent has successfully nominated a pair [RFC8445], the state of the checklist associated with the pair is set to Completed. Once the state of each checklist is set to either Completed or Failed, for each Completed checklist the agent checks whether the nominated pair matches the default candidate pair. If there are one or more pairs that do not match, and the peer did not indicate support for the 'ice2' ice-option, the controlling agent MUST generate a subsequent offer, in which the connection address, port and transport protocol in the "c=" and "m=" lines associated with each data stream match the corresponding local information of the nominated pair for that data stream (Section 4.4.1.2.2). If the peer did indicate support for the 'ice2' ice-option, the controlling agent does not immediately need to generate an updated offer in order to align a connection address, port and protocol with a nominated pair. However, later in the session, whenever the controlling agent does sent a subsequent offer, it MUST do the alignment as described above.

If there are one or more checklists with the state set to Failed, the controlling agent MUST generate a subsequent offer in order to remove the associated data streams by setting the port value of the data streams to zero (Section 4.4.1.1.2), even if the peer did indicate support for the 'ice2' ice-option. If needed, such offer is used to align the connection address, port and transport protocol, as described above.

As described in [RFC8445], once the controlling agent has nominated a candidate pair for a checklist, the agent MUST NOT nominate another pair for that checklist during the lifetime of the ICE session (i.e. until ICE is restarted).

[draft-ietf-ice-pac] provides a mechanism for allowing the ICE process to run long enough in order to find working candidate pairs, by waiting for potential peer-reflexive candidates, even though no candidate pairs were received from the peer or all current candidate pairs associated with a checklist have either failed or been discarded. It is OPTIONAL for an ICE agent to support the mechanism.

4.4. Subsequent Offer/Answer Exchanges

Either agent MAY generate a subsequent offer at any time allowed by [RFC3264]. This section defines rules for construction of subsequent offers and answers.

Should a subsequent offer fail, ICE processing continues as if the subsequent offer had never been made.

4.4.1. Sending Subsequent Offer

4.4.1.1. Procedures for All Implementations

4.4.1.1.1. ICE Restart

An agent MAY restart ICE processing for an existing data stream [RFC8445].

The rules governing the ICE restart imply that setting the connection address in the "c=" line to "0.0.0.0" (for IPv4)/ "::" (for IPv6) will cause an ICE restart. Consequently, ICE implementations MUST NOT utilize this mechanism for call hold, and instead MUST use "a=inactive" and "a=sendonly" as described in [RFC3264].

To restart ICE, an agent MUST change both the ice-pwd and the ice-ufrag for the data stream in an offer. However, it is permissible to use a session-level attribute in one offer, but to provide the same

ice-pwd or ice-ufrag as a media-level attribute in a subsequent offer. This MUST NOT be considered as ICE restart.

An agent sets the rest of the ICE-related fields in the SDP for this data stream as it would in an initial offer of this data stream (Section 4.2.1). Consequently, the set of candidates MAY include some, none, or all of the previous candidates for that data stream and MAY include a totally new set of candidates. The agent MAY modify the attribute values of the SDP ice-options and SDP ice-pacing attributes, and it MAY change its role using the SDP ice-lite attribute. The agent MUST NOT modify the SDP ice-options, ice-pacing and ice-lite attributes in a subsequent offer unless the offer is sent in order to request an ICE restart.

4.4.1.1.2. Removing a Data Stream

If an agent removes a data stream by setting its port to zero, it MUST NOT include any candidate attributes for that data stream and SHOULD NOT include any other ICE-related attributes defined in Section 5 for that data stream.

4.4.1.1.3. Adding a Data Stream

If an agent wishes to add a new data stream, it sets the fields in the SDP for this data stream as if this were an initial offer for that data stream (Section 4.2.1). This will cause ICE processing to begin for this data stream.

4.4.1.2. Procedures for Full Implementations

This section describes additional procedures for full implementations, covering existing data streams.

4.4.1.2.1. Before Nomination

When an offerer sends a subsequent offer; in each "m=" section for which a candidate pair has not yet been nominated, the offer MUST include the same set of ICE-related information that the offerer included in the previous offer or answer. The agent MAY include additional candidates it did not offer previously, but which it has gathered since the last offer/answer exchange, including peer reflexive candidates.

The agent MAY change the default destination for media. As with initial offers, there MUST be a set of candidate attributes in the offer matching this default destination.

4.4.1.2.2. After Nomination

Once a candidate pair has been nominated for a data stream, the connection address, port and transport protocol in each "c=" and "m=" line associated with that data stream MUST match the data associated with the nominated pair for that data stream. In addition, the offerer only includes SDP candidates (one per component) representing the local candidates of the nominated candidate pair. The offerer MUST NOT include any other SDP candidate attributes in the subsequent offer.

In addition, if the agent is controlling, it MUST include the "a=remote-candidates" attribute for each data stream whose checklist is in the Completed state. The attribute contains the remote candidates corresponding to the nominated pair in the valid list for each component of that data stream. It is needed to avoid a race condition whereby the controlling agent chooses its pairs, but the updated offer beats the connectivity checks to the controlled agent, which doesn't even know these pairs are valid, let alone selected. See Appendix B for elaboration on this race condition.

4.4.1.3. Procedures for Lite Implementations

If the ICE state is Running, a lite implementation MUST include all of its candidates for each component of each data stream in "a=candidate" attributes in any subsequent offer. The candidates are formed identically to the procedures for initial offers.

A lite implementation MUST NOT add additional host candidates in a subsequent offer, and MUST NOT modify the username fragments and passwords. If an agent needs to offer additional candidates, or modify the username fragments and passwords, it MUST request an ICE restart (Section 4.4.1.1.1) for that data stream.

If ICE has completed for a data stream and if the agent is controlled, the default destination for that data stream MUST be set to the remote candidate of the candidate pair for that component in the valid list. For a lite implementation, there is always just a single candidate pair in the valid list for each component of a data stream. Additionally, the agent MUST include a candidate attribute for each default destination.

If the ICE state is Completed and if the agent is controlling (which only happens when both agents are lite), the agent MUST include the "a=remote-candidates" attribute for each data stream. The attribute contains the remote candidates from the candidate pairs in the valid list (one pair for each component of each data stream).

4.4.2. Sending Subsequent Answer

If ICE is Completed for a data stream, and the offer for that data stream lacked the "a=remote-candidates" attribute, the rules for construction of the answer are identical to those for the offerer, except that the answerer MUST NOT include the "a=remote-candidates" attribute in the answer.

A controlled agent will receive an offer with the "a=remote-candidates" attribute for a data stream when its peer has concluded ICE processing for that data stream. This attribute is present in the offer to deal with a race condition between the receipt of the offer, and the receipt of the Binding Response that tells the answerer the candidate that will be selected by ICE. See Appendix B for an explanation of this race condition. Consequently, processing of an offer with this attribute depends on the winner of the race.

The agent forms a candidate pair for each component of the data stream by:

- Setting the remote candidate equal to the offerer's default destination for that component (i.e. the contents of the "m=" and "c=" lines for RTP, and the "a=rtcp" attribute for RTCP)

- Setting the local candidate equal to the transport address for that same component in the "a=remote-candidates" attribute in the offer.

The agent then sees if each of these candidate pairs is present in the valid list. If a particular pair is not in the valid list, the check has "lost" the race. Call such a pair a "losing pair".

The agent finds all the pairs in the checklist whose remote candidates equal the remote candidate in the losing pair:

- If none of the pairs are In-Progress, and at least one is Failed, it is most likely that a network failure, such as a network partition or serious packet loss, has occurred. The agent SHOULD generate an answer for this data stream as if the remote-candidates attribute had not been present, and then restart ICE for this stream.

- If at least one of the pairs is In-Progress, the agent SHOULD wait for those checks to complete, and as each completes, redo the processing in this section until there are no losing pairs.

Once there are no losing pairs, the agent can generate the answer. It MUST set the default destination for media to the candidates in

the remote-candidates attribute from the offer (each of which will now be the local candidate of a candidate pair in the valid list). It MUST include a candidate attribute in the answer for each candidate in the remote-candidates attribute in the offer.

4.4.2.1. ICE Restart

If the offerer in a subsequent offer requested an ICE restart (Section 4.4.1.1.1) for a data stream, and if the answerer accepts the offer, the answerer follows the procedures for generating an initial answer.

For a given data stream, the answerer MAY include the same candidates that were used in the previous ICE session, but it MUST change the SDP ice-pwd and ice-ufrag attribute values.

The answerer MAY modify the attribute values of the SDP ice-options and SDP ice-pacing attributes, and it MAY change its role using the SDP ice-lite attribute. The answerer MUST NOT modify the SDP ice-options, ice-pacing and ice-lite attributes in a subsequent answer unless the answer is sent for an offer that was used to request an ICE restart (Section 4.4.1.1.1). If any of the SDP attributes have been modified in a subsequent offer that is not used to request an ICE restart, the answerer MUST reject the offer.

4.4.2.2. Lite Implementation specific procedures

If the received offer contains the remote-candidates attribute for a data stream, the agent forms a candidate pair for each component of the data stream by:

- Setting the remote candidate equal to the offerer's default destination for that component (i.e., the contents of the "m=" and "c=" lines for RTP, and the "a=rtcp" attribute for RTCP).

- Setting the local candidate equal to the transport address for that same component in the "a=remote-candidates" attribute in the offer.

The state of the checklist associated with that data stream is set to Completed.

Furthermore, if the agent believed it was controlling, but the offer contained the "a=remote-candidates" attribute, both agents believe they are controlling. In this case, both would have sent updated offers around the same time.

However, the signaling protocol carrying the offer/answer exchanges will have resolved this glare condition, so that one agent is always the 'winner' by having its offer received before its peer has sent an offer. The winner takes the role of controlling, so that the loser (the answerer under consideration in this section) MUST change its role to controlled.

Consequently, if the agent was going to send an updated offer since, based on the rules in section 8.2 of [RFC8445], it was controlling, it no longer needs to.

Besides the potential role change, change in the Valid list, and state changes, the construction of the answer is performed identically to the construction of an offer.

4.4.3. Receiving Answer for a Subsequent Offer

4.4.3.1. Procedures for Full Implementations

There may be certain situations where the offerer receives an SDP answer that lacks ICE candidates although the initial answer included them. One example of such an "unexpected" answer might be happen when an ICE-unaware Back-to-Back User Agent (B2BUA) introduces a media server during call hold using 3rd party call-control procedures [RFC3725]. Omitting further details how this is done, this could result in an answer being received at the holding UA that was constructed by the B2BUA. With the B2BUA being ICE-unaware, that answer would not include ICE candidates.

Receiving an answer without ICE attributes in this situation might be unexpected, but would not necessarily impair the user experience.

When the offerer receives an answer indicating support for ICE, the offer performs one of the following actions:

- If the offer was a restart, the agent MUST perform ICE restart procedures as specified in Section 4.4.3.1.1

- If the offer/answer exchange removed a data stream, or an answer rejected an offered data stream, an agent MUST flush the Valid list for that data stream. It MUST also terminate any STUN transactions in progress for that data stream.

- If the offer/answer exchange added a new data stream, the agent MUST create a new checklist for it (and an empty Valid list to start of course) which in turn triggers the candidate processing procedures [RFC8445].

- If the checklist state associated with a data stream is Running, the agent recomputes the checklist. If a pair on the new checklist was also on the previous checklist, its candidate pair state is copied over. Otherwise, its candidate pair state is set to Frozen. If none of the checklists are active (meaning that the candidate pair states in each checklist are Frozen), appropriate procedures in [RFC8445] are performed to move candidate pair(s) to the Waiting state to further continue ICE processing.

- If the ICE state is Completed and the SDP answer conforms to Section 4.4.2, the agent MUST remain in the Completed ICE state.

However, if the ICE support is no longer indicated in the SDP answer, the agent MUST fall-back to [RFC3264] procedures and SHOULD NOT drop the dialog because of the missing ICE support or unexpected answer. Once the agent sends a new offer later on, it MUST perform an ICE restart.

4.4.3.1.1. ICE Restarts

The agent MUST remember the nominated pair in the Valid list for each component of the data stream, called the "previous selected pair", prior to the restart. The agent will continue to send media using this pair, as described in section 12 of [RFC8445]. Once these destinations are noted, the agent MUST flush the Valid lists and checklists, and then recompute the checklist and its states, thus triggering the candidate processing procedures [RFC8445]

4.4.3.2. Procedures for Lite Implementations

If ICE is restarting for a data stream, the agent MUST create a new Valid list for that data stream. It MUST remember the nominated pair in the previous Valid list for each component of the data stream, called the "previous selected pairs", and continue to send media there as described in section 12 of [RFC8445]. The state of each checklist for each data stream MUST change to Running, and the ICE state MUST be set to Running.

5. Grammar

This specification defines eight new SDP attributes -- the "candidate", "remote-candidates", "ice-lite", "ice-mismatch", "ice-ufrag", "ice-pwd", "ice-pacing", and "ice-options" attributes.

This section also provides non-normative examples of the attributes defined.

The syntax for the attributes follow Augmented BNF as defined in [RFC5234].

5.1. "candidate" Attribute

The candidate attribute is a media-level attribute only. It contains a transport address for a candidate that can be used for connectivity checks.

```
candidate-attribute   = "candidate" ":" foundation SP component-id SP
                        transport SP
                        priority SP
                        connection-address SP     ;from RFC 4566
                        port         ;port from RFC 4566
                        SP cand-type
                        [SP rel-addr]
                        [SP rel-port]
                        *(SP cand-extension)

foundation            = 1*32ice-char
component-id          = 1*3DIGIT
transport             = "UDP" / transport-extension
transport-extension   = token              ; from RFC 3261
priority              = 1*10DIGIT
cand-type             = "typ" SP candidate-types
candidate-types       = "host" / "srflx" / "prflx" / "relay" / token
rel-addr              = "raddr" SP connection-address
rel-port              = "rport" SP port
cand-extension        = extension-att-name SP extension-att-value
extension-att-name    = token
extension-att-value   = *VCHAR
ice-char              = ALPHA / DIGIT / "+" / "/"
```

This grammar encodes the primary information about a candidate: its IP address, port and transport protocol, and its properties: the foundation, component ID, priority, type, and related transport address:

<connection-address>: is taken from RFC 4566 [RFC4566]. It is the IP address of the candidate, allowing for IPv4 addresses, IPv6 addresses, and fully qualified domain names (FQDNs). When parsing this field, an agent can differentiate an IPv4 address and an IPv6 address by presence of a colon in its value - the presence of a colon indicates IPv6. An agent generating local candidates MUST NOT use FQDN addresses. An agent processing remote candidates MUST ignore candidate lines that include candidates with FQDN or IP address versions that are not supported or recognized. The procedures for generation and handling of FQDN candidates, as well

as, how agents indicate support for such procedures, need to be specified in an extension specification.

<port>: is also taken from RFC 4566 [RFC4566]. It is the port of the candidate.

<transport>: indicates the transport protocol for the candidate. This specification only defines UDP. However, extensibility is provided to allow for future transport protocols to be used with ICE by extending the sub-registry "ICE Transport Protocols" under "Interactive Connectivity Establishment (ICE)" registry.

<foundation>: is composed of 1 to 32 <ice-char>s. It is an identifier that is equivalent for two candidates that are of the same type, share the same base, and come from the same STUN server. The foundation is used to optimize ICE performance in the Frozen algorithm as described in [RFC8445]

<component-id>: is a positive integer between 1 and 256 (inclusive) that identifies the specific component of the data stream for which this is a candidate. It MUST start at 1 and MUST increment by 1 for each component of a particular candidate. For data streams based on RTP, candidates for the actual RTP media MUST have a component ID of 1, and candidates for RTCP MUST have a component ID of 2. See section 13 in [RFC8445] for additional discussion on extending ICE to new data streams.

<priority>: is a positive integer between 1 and (2**31 - 1) inclusive. The procedures for computing candidate's priority is described in section 5.1.2 of [RFC8445].

<cand-type>: encodes the type of candidate. This specification defines the values "host", "srflx", "prflx", and "relay" for host, server reflexive, peer reflexive, and relayed candidates, respectively. Specifications for new candidate types MUST define how, if at all, various steps in the ICE processing differ from the ones defined by this specification.

<rel-addr> and <rel-port>: convey transport addresses related to the candidate, useful for diagnostics and other purposes. <rel-addr> and <rel-port> MUST be present for server reflexive, peer reflexive, and relayed candidates. If a candidate is server or peer reflexive, <rel-addr> and <rel-port> are equal to the base for that server or peer reflexive candidate. If the candidate is relayed, <rel-addr> and <rel-port> are equal to the mapped address in the Allocate response that provided the client with that relayed candidate (see Appendix B.3 of [RFC8445] for a discussion

of its purpose). If the candidate is a host candidate, <rel-addr> and <rel-port> MUST be omitted.

In some cases, e.g., for privacy reasons, an agent may not want to reveal the related address and port. In this case the address MUST be set to "0.0.0.0" (for IPv4 candidates) or "::" (for IPv6 candidates) and the port to '9'.

The candidate attribute can itself be extended. The grammar allows for new name/value pairs to be added at the end of the attribute. Such extensions MUST be made through IETF Review or IESG Approval [RFC8126] and the assignments MUST contain the specific extension and a reference to the document defining the usage of the extension.

An implementation MUST ignore any name/value pairs it doesn't understand.

Example: SDP line for UDP server reflexive candidate attribute for the RTP component

a=candidate:2 1 UDP 1694498815 192.0.2.3 45664 typ srflx raddr 203.0.113.141 rport 8998

5.2. "remote-candidates" Attribute

The syntax of the "remote-candidates" attribute is defined using Augmented BNF as defined in [RFC5234]. The remote-candidates attribute is a media-level attribute only.

```
remote-candidate-att = "remote-candidates:" remote-candidate
                          0*(SP remote-candidate)
remote-candidate = component-ID SP connection-address SP port
```

The attribute contains a connection-address and port for each component. The ordering of components is irrelevant. However, a value MUST be present for each component of a data stream. This attribute MUST be included in an offer by a controlling agent for a data stream that is Completed, and MUST NOT be included in any other case.

Example: Remote candidates SDP lines for the RTP and RTCP components:

a=remote-candidates:1 192.0.2.3 45664
a=remote-candidates:2 192.0.2.3 45665

5.3. "ice-lite" and "ice-mismatch" Attributes

The syntax of the "ice-lite" and "ice-mismatch" attributes, both of which are flags, is:

```
ice-lite               = "ice-lite"
ice-mismatch           = "ice-mismatch"
```

"ice-lite" is a session-level attribute only, and indicates that an agent is a lite implementation. "ice-mismatch" is a media-level attribute and only reported in the answer. It indicates that the offer arrived with a default destination for a media component that didn't have a corresponding candidate attribute. Inclusion of "a=ice-mismatch" attribute for a given data stream implies that even though both agents support ICE, ICE procedures MUST NOT be used for this data stream and [RFC3264] procedures MUST be used instead.

5.4. "ice-ufrag" and "ice-pwd" Attributes

The "ice-ufrag" and "ice-pwd" attributes convey the username fragment and password used by ICE for message integrity. Their syntax is:

```
ice-pwd-att           = "ice-pwd:" password
ice-ufrag-att         = "ice-ufrag:" ufrag
password              = 22*256ice-char
ufrag                 = 4*256ice-char
```

The "ice-pwd" and "ice-ufrag" attributes can appear at either the session-level or media-level. When present in both, the value in the media-level takes precedence. Thus, the value at the session-level is effectively a default that applies to all data streams, unless overridden by a media-level value. Whether present at the session or media-level, there MUST be an ice-pwd and ice-ufrag attribute for each data stream. If two data streams have identical ice-ufrag's, they MUST have identical ice-pwd's.

The ice-ufrag and ice-pwd attributes MUST be chosen randomly at the beginning of a session (the same applies when ICE is restarting for an agent).

[RFC8445] requires the ice-ufrag attribute to contain at least 24 bits of randomness, and the ice-pwd attribute to contain at least 128 bits of randomness. This means that the ice-ufrag attribute will be at least 4 characters long, and the ice-pwd at least 22 characters long, since the grammar for these attributes allows for 6 bits of information per character. The attributes MAY be longer than 4 and 22 characters, respectively, of course, up to 256 characters. The upper limit allows for buffer sizing in implementations. Its large

upper limit allows for increased amounts of randomness to be added over time. For compatibility with the 512 character limitation for the STUN username attribute value and for bandwidth conservation considerations, the ice-ufrag attribute MUST NOT be longer than 32 characters when sending, but an implementation MUST accept up to 256 characters when receiving.

Example shows sample ice-ufrag and ice-pwd SDP lines:

```
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
```

5.5. "ice-pacing" Attribute

The "ice-pacing" is a session level attribute that indicates the desired connectivity check pacing (Ta interval), in milliseconds, that the sender wishes to use. See section 14.2 of [RFC8445] for more information regarding selecting a pacing value. The syntax is:

```
ice-pacing-att            = "ice-pacing:" pacing-value
pacing-value              = 1*10DIGIT
```

If absent in an offer or answer the default value of the attribute is 50 ms, which is the recommended value specified in [RFC8445].

Once both agents have indicated the pacing value they with to use, both agents MUST use the larger of the indicated values.

Example shows an ice-pacing SDP line with value ’50’:

```
a=ice-pacing:50
```

5.6. "ice-options" Attribute

The "ice-options" attribute is a session- and media-level attribute. It contains a series of tokens that identify the options supported by the agent. Its grammar is:

```
ice-options           = "ice-options:" ice-option-tag
                            *(SP ice-option-tag)
ice-option-tag        = 1*ice-char
```

The existence of an ice-option in an offer indicates that a certain extension is supported by the agent and it is willing to use it, if the peer agent also includes the same extension in the answer. There might be further extension specific negotiation needed between the agents that determine how the extension gets used in a given session. The details of the negotiation procedures, if present, MUST be defined by the specification defining the extension (Section 10.2).

Example shows an ice-options SDP line with 'ice2' and 'rtp+ecn' [RFC6679] values :

a=ice-options:ice2 rtp+ecn

6. Keepalives

All the ICE agents MUST follow the procedures defined in section 11 of [RFC8445] for sending keepalives. The keepalives MUST be sent regardless of whether the data stream is currently inactive, sendonly, recvonly, or sendrecv, and regardless of the presence or value of the bandwidth attribute. An agent can determine that its peer supports ICE by the presence of "a=candidate" attributes for each media session.

7. SIP Considerations

Note that ICE is not intended for NAT traversal for SIP signaling, which is assumed to be provided via another mechanism [RFC5626].

When ICE is used with SIP, forking may result in a single offer generating a multiplicity of answers. In that case, ICE proceeds completely in parallel and independently for each answer, treating the combination of its offer and each answer as an independent offer/answer exchange, with its own set of local candidates, pairs, checklists, states, and so on.

7.1. Latency Guidelines

ICE requires a series of STUN-based connectivity checks to take place between endpoints. These checks start from the answerer on generation of its answer, and start from the offerer when it receives the answer. These checks can take time to complete, and as such, the selection of messages to use with offers and answers can affect perceived user latency. Two latency figures are of particular interest. These are the post-pickup delay and the post-dial delay. The post-pickup delay refers to the time between when a user "answers the phone" and when any speech they utter can be delivered to the caller. The post-dial delay refers to the time between when a user enters the destination address for the user and ringback begins as a consequence of having successfully started alerting the called user agent.

Two cases can be considered -- one where the offer is present in the initial INVITE and one where it is in a response.

7.1.1. Offer in INVITE

To reduce post-dial delays, it is RECOMMENDED that the caller begin gathering candidates prior to actually sending its initial INVITE, so that the candidates can be provided in the INVITE. This can be started upon user interface cues that a call is pending, such as activity on a keypad or the phone going off-hook.

On the receipt of the offer, the answerer SHOULD generate an answer in a provisional response as soon as it has completed gathering the candidates. ICE requires that a provisional response with an SDP be transmitted reliably. This can be done through the existing Provisional Response Acknowledgment (PRACK) mechanism [RFC3262] or through an ICE specific optimization, wherein, the agent retransmits the provisional response with the exponential backoff timers described in [RFC3262]. Such retransmissions MUST cease on receipt of a STUN Binding request with the transport address matching the candidate address for one of the data streams signaled in that SDP or on transmission of the answer in a 2xx response. If no Binding request is received prior to the last retransmit, the agent does not consider the session terminated. For the ICE lite peers, the agent MUST cease retransmitting the 18x after sending it four times since there will be no Binding request sent and the number four is arbitrarily chosen to limit the number of 18x retransmits.

Once the answer has been sent, the agent SHOULD begin its connectivity checks. Once candidate pairs for each component of a data stream enter the valid list, the answerer can begin sending media on that data stream.

However, prior to this point, any media that needs to be sent towards the caller (such as SIP early media [RFC3960]) MUST NOT be transmitted. For this reason, implementations SHOULD delay alerting the called party until candidates for each component of each data stream have entered the valid list. In the case of a PSTN gateway, this would mean that the setup message into the PSTN is delayed until this point. Doing this increases the post-dial delay, but has the effect of eliminating 'ghost rings'. Ghost rings are cases where the called party hears the phone ring, picks up, but hears nothing and cannot be heard. This technique works without requiring support for, or usage of, preconditions [RFC3312]. It also has the benefit of guaranteeing that not a single packet of media will get clipped, so that post-pickup delay is zero. If an agent chooses to delay local alerting in this way, it SHOULD generate a 180 response once alerting begins.

7.1.2. Offer in Response

In addition to uses where the offer is in an INVITE, and the answer is in the provisional and/or 200 OK response, ICE works with cases where the offer appears in the response. In such cases, which are common in third party call control [RFC3725], ICE agents SHOULD generate their offers in a reliable provisional response (which MUST utilize [RFC3262]), and not alert the user on receipt of the INVITE. The answer will arrive in a PRACK. This allows for ICE processing to take place prior to alerting, so that there is no post-pickup delay, at the expense of increased call setup delays. Once ICE completes, the callee can alert the user and then generate a 200 OK when they answer. The 200 OK would contain no SDP, since the offer/answer exchange has completed.

Alternatively, agents MAY place the offer in a 2xx instead (in which case the answer comes in the ACK). When this happens, the callee will alert the user on receipt of the INVITE, and the ICE exchanges will take place only after the user answers. This has the effect of reducing call setup delay, but can cause substantial post-pickup delays and media clipping.

7.2. SIP Option Tags and Media Feature Tags

[RFC5768] specifies a SIP option tag and media feature tag for usage with ICE. ICE implementations using SIP SHOULD support this specification, which uses a feature tag in registrations to facilitate interoperability through signaling intermediaries.

7.3. Interactions with Forking

ICE interacts very well with forking. Indeed, ICE fixes some of the problems associated with forking. Without ICE, when a call forks and the caller receives multiple incoming data streams, it cannot determine which data stream corresponds to which callee.

With ICE, this problem is resolved. The connectivity checks which occur prior to transmission of media carry username fragments, which in turn are correlated to a specific callee. Subsequent media packets that arrive on the same candidate pair as the connectivity check will be associated with that same callee. Thus, the caller can perform this correlation as long as it has received an answer.

7.4. Interactions with Preconditions

Quality of Service (QoS) preconditions, which are defined in [RFC3312] and [RFC4032], apply only to the transport addresses listed as the default targets for media in an offer/answer. If ICE changes

the transport address where media is received, this change is reflected in an updated offer that changes the default destination for media to match ICE's selection. As such, it appears like any other re-INVITE would, and is fully treated in RFCs 3312 and 4032, which apply without regard to the fact that the destination for media is changing due to ICE negotiations occurring "in the background".

Indeed, an agent SHOULD NOT indicate that QoS preconditions have been met until the checks have completed and selected the candidate pairs to be used for media.

ICE also has (purposeful) interactions with connectivity preconditions [RFC5898]. Those interactions are described there. Note that the procedures described in Section 7.1 describe their own type of "preconditions", albeit with less functionality than those provided by the explicit preconditions in [RFC5898].

7.5. Interactions with Third Party Call Control

ICE works with Flows I, III, and IV as described in [RFC3725]. Flow I works without the controller supporting or being aware of ICE. Flow IV will work as long as the controller passes along the ICE attributes without alteration. Flow II is fundamentally incompatible with ICE; each agent will believe itself to be the answerer and thus never generate a re-INVITE.

The flows for continued operation, as described in Section 7 of [RFC3725], require additional behavior of ICE implementations to support. In particular, if an agent receives a mid-dialog re-INVITE that contains no offer, it MUST restart ICE for each data stream and go through the process of gathering new candidates. Furthermore, that list of candidates SHOULD include the ones currently being used for media.

8. Interactions with Application Layer Gateways and SIP

Application Layer Gateways (ALGs) are functions present in a Network Address Translation (NAT) device that inspect the contents of packets and modify them, in order to facilitate NAT traversal for application protocols. Session Border Controllers (SBCs) are close cousins of ALGs, but are less transparent since they actually exist as application-layer SIP intermediaries. ICE has interactions with SBCs and ALGs.

If an ALG is SIP aware but not ICE aware, ICE will work through it as long as the ALG correctly modifies the SDP. A correct ALG implementation behaves as follows:

- The ALG does not modify the "m=" and "c=" lines or the rtcp attribute if they contain external addresses.

- If the "m=" and "c=" lines contain internal addresses, the modification depends on the state of the ALG:

  * If the ALG already has a binding established that maps an external port to an internal connection address and port matching the values in the "m=" and "c=" lines or rtcp attribute, the ALG uses that binding instead of creating a new one.

  * If the ALG does not already have a binding, it creates a new one and modifies the SDP, rewriting the "m=" and "c=" lines and rtcp attribute.

Unfortunately, many ALGs are known to work poorly in these corner cases. ICE does not try to work around broken ALGs, as this is outside the scope of its functionality. ICE can help diagnose these conditions, which often show up as a mismatch between the set of candidates and the "m=" and "c=" lines and rtcp attributes. The ice-mismatch attribute is used for this purpose.

ICE works best through ALGs when the signaling is run over TLS. This prevents the ALG from manipulating the SDP messages and interfering with ICE operation. Implementations that are expected to be deployed behind ALGs SHOULD provide for TLS transport of the SDP.

If an SBC is SIP aware but not ICE aware, the result depends on the behavior of the SBC. If it is acting as a proper Back-to-Back User Agent (B2BUA), the SBC will remove any SDP attributes it doesn't understand, including the ICE attributes. Consequently, the call will appear to both endpoints as if the other side doesn't support ICE. This will result in ICE being disabled, and media flowing through the SBC, if the SBC has requested it. If, however, the SBC passes the ICE attributes without modification, yet modifies the default destination for media (contained in the "m=" and "c=" lines and rtcp attribute), this will be detected as an ICE mismatch, and ICE processing is aborted for the call. It is outside of the scope of ICE for it to act as a tool for "working around" SBCs. If one is present, ICE will not be used and the SBC techniques take precedence.

9. Security Considerations

The generic ICE security considerations are defined in [RFC8445], and the generic SDP offer/answer security considerations are defined in [RFC3264]. These security considerations also apply to implementations of this document.

9.1. IP Address Privacy

In some cases, e.g., for privacy reasons, an agent may not want to reveal the related address and port. In this case the address MUST be set to "0.0.0.0" (for IPv4 candidates) or "::" (for IPv6 candidates) and the port to '9'.

9.2. Attacks on the Offer/Answer Exchanges

An attacker that can modify or disrupt the offer/answer exchanges themselves can readily launch a variety of attacks with ICE. They could direct media to a target of a DoS attack, they could insert themselves into the data stream, and so on. These are similar to the general security considerations for offer/answer exchanges, and the security considerations in [RFC3264] apply. These require techniques for message integrity and encryption for offers and answers, which are satisfied by the TLS mechanism [RFC3261] when SIP is used. As such, the usage of TLS with ICE is RECOMMENDED.

9.3. The Voice Hammer Attack

The voice hammer attack is an amplification attack, and can be triggered even if the attacker is an authenticated and valid participant in a session. In this attack, the attacker initiates sessions to other agents, and maliciously includes the connection address and port of a DoS target as the destination for media traffic signaled in the SDP. This causes substantial amplification; a single offer/answer exchange can create a continuing flood of media packets, possibly at high rates (consider video sources). The use of ICE can help to prevent against this attack.

Specifically, if ICE is used, the agent receiving the malicious SDP will first perform connectivity checks to the target of media before sending media there. If this target is a third-party host, the checks will not succeed, and media is never sent.

Unfortunately, ICE doesn't help if it's not used, in which case an attacker could simply send the offer without the ICE parameters. However, in environments where the set of clients is known, and is limited to ones that support ICE, the server can reject any offers or answers that don't indicate ICE support.

SIP User Agents (UA) [RFC3261] that are not willing to receive non-ICE answers MUST include an "ice" Option Tag [RFC5768] in the SIP Require Header Field in their offer. UAs that reject non-ICE offers will generally use a 421 response code, together with an Option Tag "ice" in the Require Header Field in the response.

10. IANA Considerations

10.1. SDP Attributes

The original ICE specification defined seven new SDP attributes per the procedures of Section 8.2.4 of [RFC4566]. The registration information from the original specification is included here with modifications to include Mux Category and also defines a new SDP attribute 'ice-pacing'.

10.1.1. candidate Attribute

Attribute Name: candidate

Type of Attribute: media-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides one of many possible candidate addresses for communication. These addresses are validated with an end-to-end connectivity check using Session Traversal Utilities for NAT (STUN).

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact Email: iesg@ietf.org

Reference: RFCXXXX

Mux Category: TRANSPORT

10.1.2. remote-candidates Attribute

Attribute Name: remote-candidates

Type of Attribute: media-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides the identity of the remote candidates that the offerer wishes the answerer to use in its answer.

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact Email: iesg@ietf.org

Reference: RFCXXXX

Mux Category: TRANSPORT

10.1.3. ice-lite Attribute

Attribute Name: ice-lite

Type of Attribute: session-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and indicates that an agent has the minimum functionality required to support ICE inter-operation with a peer that has a full implementation.

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact Email: iesg@ietf.org

Reference: RFCXXXX

Mux Category: NORMAL

10.1.4. ice-mismatch Attribute

Attribute Name: ice-mismatch

Type of Attribute: media-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and indicates that an agent is ICE capable, but did not proceed with ICE due to a mismatch of candidates with the default destination for media signaled in the SDP.

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact e-mail: iesg@ietf.org

Reference: RFCXXXX

Mux Category: NORMAL

10.1.5. ice-pwd Attribute

Attribute Name: ice-pwd

Type of Attribute: session- or media-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides the password used to protect STUN connectivity checks.

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact e-mail: iesg@ietf.org

Reference: RFCXXXX

Mux Category: TRANSPORT

10.1.6. ice-ufrag Attribute

Attribute Name: ice-ufrag

Type of Attribute: session- or media-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and provides the fragments used to construct the username in STUN connectivity checks.

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact e-mail: iesg@ietf.org

Reference: RFCXXXX

Mux Category: TRANSPORT

10.1.7. ice-options Attribute

Attribute Name: ice-options

Long Form: ice-options

Type of Attribute: session-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE), and indicates the ICE options or extensions used by the agent.

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact e-mail: iesg@ietf.org

Reference: RFCXXXX

Mux Category: NORMAL

10.1.8. ice-pacing Attribute

This specification also defines a new SDP attribute, "ice-pacing" according to the following data:

Attribute Name: ice-pacing

Type of Attribute: session-level

Subject to charset: No

Purpose: This attribute is used with Interactive Connectivity Establishment (ICE) to indicate desired connectivity check pacing values.

Appropriate Values: See Section 5 of RFC XXXX.

Contact Name: IESG

Contact e-mail: iesg@ietf.org

Reference: RFCXXXX

Mux Category: NORMAL

10.2. Interactive Connectivity Establishment (ICE) Options Registry

IANA maintains a registry for ice-options identifiers under the Specification Required policy as defined in "Guidelines for Writing an IANA Considerations Section in RFCs" [RFC8126].

ICE options are of unlimited length according to the syntax in Section 5.6; however, they are RECOMMENDED to be no longer than 20 characters. This is to reduce message sizes and allow for efficient parsing. ICE options are defined at the session level.

A registration request MUST include the following information:

- The ICE option identifier to be registered
- Short description of the ICE extension to which the option relates
- Reference(s) to the specification defining the ICE option and the related extensions

10.3. Candidate Attribute Extension Subregistry Establishment

This section creates a new sub-registry, "Candidate Attribute Extensions", under the sdp-parameters registry: http://www.iana.org/assignments/sdp-parameters.

The purpose of the sub-registry is to register SDP candidate attribute extensions.

When a candidate extension is registered in the sub-registry, it needs to meet the "Specification Required" policies defined in [RFC8126].

Candidate attribute extensions MUST follow the 'cand-extension' syntax. The attribute extension name MUST follow the 'extension-att-name' syntax, and the attribute extension value MUST follow the 'extension-att-value' syntax.

A registration request MUST include the following information:

- The name of the attribute extension.
- A short description of the attribute extension.
- A reference to a specification that describes the semantics, usage and possible values of the attribute extension.

11. Acknowledgments

A large part of the text in this document was taken from [RFC5245], authored by Jonathan Rosenberg.

Some of the text in this document was taken from [RFC6336], authored by Magnus Westerlund and Colin Perkins.

Many thanks to Flemming Andreasen for shepherd review feedback.

Thanks to following experts for their reviews and constructive feedback: Thomas Stach, Adam Roach, Peter Saint-Andre, Roman Danyliw, Alissa Cooper, Benjamin Kaduk, Mirja Kuhlewind, Alexey Melnikov, Eric Vyncke for their detailed reviews.

12. Changes from RFC 5245

[RFC8445] describes the changes that were done to the common SIP procedures, including removal of aggressive nomination, modifying the procedures for calculating candidate pair states and scheduling connectivity checks and the calculation of timer values.

This document defines the following SDP offer/answer specific changes:

- SDP offer/answer realization and usage of of 'ice2' option.
- Definition and usage of SDP 'ice-pacing' attribute.
- Explicit text that an ICE agent must not generate candidates with FQDNs, and must discard such candidates if received from the peer agent.
- Relax requirement to include SDP 'rtcp' attribute.
- Generic clarifications of SDP offer/answer procedures.

13. References

13.1. Normative References

[draft-ietf-ice-pac]
Holmberg, C. and J. Uberti, "Interactive Connectivity Establishment Patiently Awaiting Connectivity (ICE PAC)", draft-ietf-ice-pac-02 (work in progress), July 2019, <http://www.ietf.org/internet-drafts/draft-ietf-ice-pac-02.txt>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

[RFC3262] Rosenberg, J. and H. Schulzrinne, "Reliability of Provisional Responses in Session Initiation Protocol (SIP)", RFC 3262, DOI 10.17487/RFC3262, June 2002, <https://www.rfc-editor.org/info/rfc3262>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <https://www.rfc-editor.org/info/rfc3264>.

[RFC3312] Camarillo, G., Ed., Marshall, W., Ed., and J. Rosenberg, "Integration of Resource Management and Session Initiation Protocol (SIP)", RFC 3312, DOI 10.17487/RFC3312, October 2002, <https://www.rfc-editor.org/info/rfc3312>.

[RFC3556] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, DOI 10.17487/RFC3556, July 2003, <https://www.rfc-editor.org/info/rfc3556>.

[RFC3605] Huitema, C., "Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)", RFC 3605, DOI 10.17487/RFC3605, October 2003, <https://www.rfc-editor.org/info/rfc3605>.

[RFC4032] Camarillo, G. and P. Kyzivat, "Update to the Session Initiation Protocol (SIP) Preconditions Framework", RFC 4032, DOI 10.17487/RFC4032, March 2005, <https://www.rfc-editor.org/info/rfc4032>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <https://www.rfc-editor.org/info/rfc4566>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC5768] Rosenberg, J., "Indicating Support for Interactive Connectivity Establishment (ICE) in the Session Initiation Protocol (SIP)", RFC 5768, DOI 10.17487/RFC5768, April 2010, <https://www.rfc-editor.org/info/rfc5768>.

[RFC6336] Westerlund, M. and C. Perkins, "IANA Registry for Interactive Connectivity Establishment (ICE) Options", RFC 6336, DOI 10.17487/RFC6336, July 2011, <https://www.rfc-editor.org/info/rfc6336>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8445] Keranen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", RFC 8445, DOI 10.17487/RFC8445, July 2018, <https://www.rfc-editor.org/info/rfc8445>.

13.2. Informative References

[RFC3725] Rosenberg, J., Peterson, J., Schulzrinne, H., and G. Camarillo, "Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)", BCP 85, RFC 3725, DOI 10.17487/RFC3725, April 2004, <https://www.rfc-editor.org/info/rfc3725>.

[RFC3960] Camarillo, G. and H. Schulzrinne, "Early Media and Ringing Tone Generation in the Session Initiation Protocol (SIP)", RFC 3960, DOI 10.17487/RFC3960, December 2004, <https://www.rfc-editor.org/info/rfc3960>.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, DOI 10.17487/RFC5245, April 2010, <https://www.rfc-editor.org/info/rfc5245>.

[RFC5626] Jennings, C., Ed., Mahy, R., Ed., and F. Audet, Ed., "Managing Client-Initiated Connections in the Session Initiation Protocol (SIP)", RFC 5626, DOI 10.17487/RFC5626, October 2009, <https://www.rfc-editor.org/info/rfc5626>.

[RFC5898] Andreasen, F., Camarillo, G., Oran, D., and D. Wing, "Connectivity Preconditions for Session Description Protocol (SDP) Media Streams", RFC 5898, DOI 10.17487/RFC5898, July 2010, <https://www.rfc-editor.org/info/rfc5898>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <https://www.rfc-editor.org/info/rfc6679>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

Appendix A. Examples

For the example shown in section 15 of [RFC8445] the resulting offer (message 5) encoded in SDP looks like:

```
v=0
o=jdoe 2890844526 2890842807 IN IP6 $L-PRIV-1.IP
s=
c=IN IP6 $NAT-PUB-1.IP
t=0 0
a=ice-options:ice2
a=ice-pacing:50
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
m=audio $NAT-PUB-1.PORT RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 $L-PRIV-1.IP $L-PRIV-1.PORT typ host
a=candidate:2 1 UDP 1694498815 $NAT-PUB-1.IP $NAT-PUB-1.PORT typ
 srflx raddr $L-PRIV-1.IP rport $L-PRIV-1.PORT
```

The offer, with the variables replaced with their values, will look like (lines folded for clarity):

```
v=0
o=jdoe 2890844526 2890842807 IN IP6 fe80::6676:baff:fe9c:ee4a
s=
c=IN IP6 2001:db8:8101:3a55:4858:a2a9:22ff:99b9
t=0 0
a=ice-options:ice2
a=ice-pacing:50
a=ice-pwd:asd88fgpdd777uzjYhagZg
a=ice-ufrag:8hhY
m=audio 45664 RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 fe80::6676:baff:fe9c:ee4a 8998 typ host
a=candidate:2 1 UDP 1694498815 2001:db8:8101:3a55:4858:a2a9:22ff:99b9
45664 typ srflx raddr fe80::6676:baff:fe9c:ee4a rport 8998
```

The resulting answer looks like:

```
v=0
o=bob 2808844564 2808844564 IN IP4 $R-PUB-1.IP
s=
c=IN IP4 $R-PUB-1.IP
t=0 0
a=ice-options:ice2
a=ice-pacing:50
a=ice-pwd:YH75Fviy6338Vbrhrlp8Yh
a=ice-ufrag:9uB6
m=audio $R-PUB-1.PORT RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 $R-PUB-1.IP $R-PUB-1.PORT typ host
```

With the variables filled in:

```
v=0
o=bob 2808844564 2808844564 IN IP4 192.0.2.1
s=
c=IN IP4 192.0.2.1
t=0 0
a=ice-options:ice2
a=ice-pacing:50
a=ice-pwd:YH75Fviy6338Vbrhrlp8Yh
a=ice-ufrag:9uB6
m=audio 3478 RTP/AVP 0
b=RS:0
b=RR:0
a=rtpmap:0 PCMU/8000
a=candidate:1 1 UDP 2130706431 192.0.2.1 3478 typ host
```

Appendix B. The remote-candidates Attribute

The "a=remote-candidates" attribute exists to eliminate a race condition between the updated offer and the response to the STUN Binding request that moved a candidate into the Valid list. This race condition is shown in Figure 1. On receipt of message 4, agent L adds a candidate pair to the valid list. If there was only a single data stream with a single component, agent L could now send an updated offer. However, the check from agent R has not yet generated a response, and agent R receives the updated offer (message 7) before getting the response (message 9). Thus, it does not yet know that this particular pair is valid. To eliminate this condition, the actual candidates at R that were selected by the offerer (the remote candidates) are included in the offer itself, and the answerer delays its answer until those pairs validate.

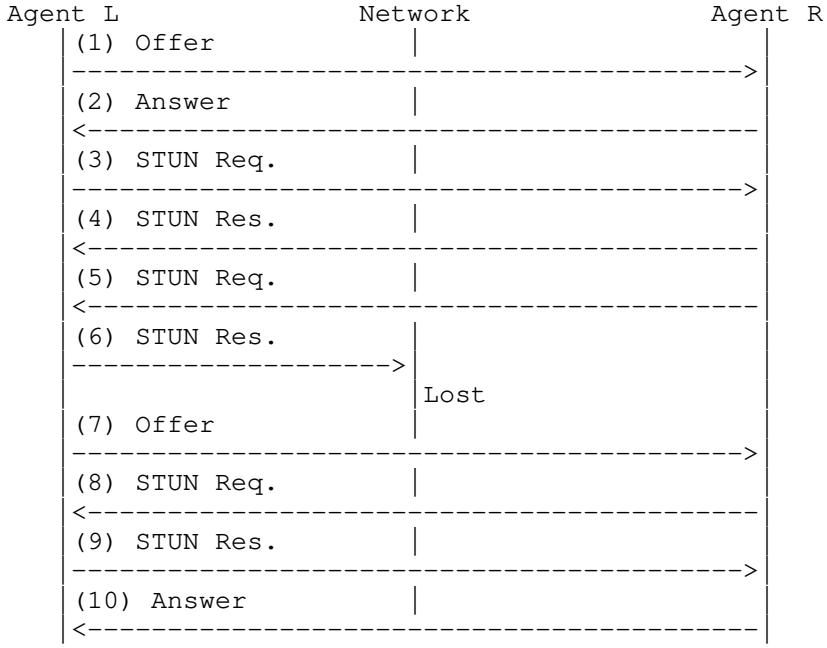

Figure 1: Race Condition Flow

Appendix C. Why Is the Conflict Resolution Mechanism Needed?

When ICE runs between two peers, one agent acts as controlled, and the other as controlling. Rules are defined as a function of implementation type and offerer/answerer to determine who is controlling and who is controlled. However, the specification mentions that, in some cases, both sides might believe they are controlling, or both sides might believe they are controlled. How can this happen?

The condition when both agents believe they are controlled shows up in third party call control cases. Consider the following flow:

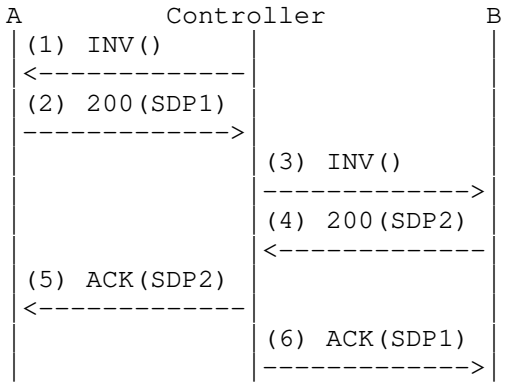

Figure 2: Role Conflict Flow

This flow is a variation on flow III of RFC 3725 [RFC3725]. In fact, it works better than flow III since it produces fewer messages. In this flow, the controller sends an offerless INVITE to agent A, which responds with its offer, SDP1. The agent then sends an offerless INVITE to agent B, which it responds to with its offer, SDP2. The controller then uses the offer from each agent to generate the answers. When this flow is used, ICE will run between agents A and B, but both will believe they are in the controlling role. With the role conflict resolution procedures, this flow will function properly when ICE is used.

At this time, there are no documented flows that can result in the case where both agents believe they are controlled. However, the conflict resolution procedures allow for this case, should a flow arise that would fit into this category.

Appendix D. Why Send an Updated Offer?

Section 11.1 describes rules for sending media. Both agents can send media once ICE checks complete, without waiting for an updated offer. Indeed, the only purpose of the updated offer is to "correct" the SDP so that the default destination for media matches where media is being sent based on ICE procedures (which will be the highest-priority nominated candidate pair).

This raises the question -- why is the updated offer/answer exchange needed at all? Indeed, in a pure offer/answer environment, it would not be. The offerer and answerer will agree on the candidates to use through ICE, and then can begin using them. As far as the agents themselves are concerned, the updated offer/answer provides no new information. However, in practice, numerous components along the signaling path look at the SDP information. These include entities

performing off-path QoS reservations, NAT traversal components such as ALGs and Session Border Controllers (SBCs), and diagnostic tools that passively monitor the network. For these tools to continue to function without change, the core property of SDP -- that the existing, pre-ICE definitions of the addresses used for media -- the "m=" and "c=" lines and the rtcp attribute -- must be retained. For this reason, an updated offer must be sent.

Appendix E. Contributors

Following experts have contributed textual and structural improvements for this work

1. Thomas Stach

    * thomass.stach@gmail.com

Authors' Addresses

Marc Petit-Huguenin
Impedance Mismatch

Email: marc@petit-huguenin.org

Suhas Nandakumar
Cisco Systems
707 Tasman Dr
Milpitas, CA 95035
USA

Email: snandaku@cisco.com

Christer Holmberg
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: christer.holmberg@ericsson.com

Ari Keranen
Ericsson
Jorvas 02420
Finland

Email: ari.keranen@ericsson.com

Roman Shpount
TurboBridge
4905 Del Ray Avenue, Suite 300
Bethesda, MD 20814
USA

Phone: +1 (240) 292-6632
Email: rshpount@turbobridge.com

Network Working Group
Internet-Draft
Obsoletes: 4566 (if approved)
Intended status: Standards Track
Expires: February 10, 2020

A. Begen
Networked Media
P. Kyzivat
C. Perkins
University of Glasgow
M. Handley
UCL

August 9, 2019

SDP: Session Description Protocol
draft-ietf-mmusic-rfc4566bis-37

Abstract

This memo defines the Session Description Protocol (SDP). SDP is intended for describing multimedia sessions for the purposes of session announcement, session invitation, and other forms of multimedia session initiation. This document obsoletes RFC 4566.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 10, 2020.

Table of Contents

1. Introduction

When initiating multimedia teleconferences, voice-over-IP calls, streaming video, or other sessions, there is a requirement to convey media details, transport addresses, and other session description metadata to the participants.

SDP provides a standard representation for such information, irrespective of how that information is transported. SDP is purely a format for session description -- it does not incorporate a transport

protocol, and it is intended to use different transport protocols as appropriate, including the Session Announcement Protocol (SAP) [RFC2974], Session Initiation Protocol (SIP) [RFC3261], Real Time Streaming Protocol (RTSP) [RFC7826], electronic mail [RFC5322] using the MIME extensions [RFC2045], and the Hypertext Transport Protocol (HTTP) [RFC7230].

SDP is intended to be general purpose so that it can be used in a wide range of network environments and applications. However, it is not intended to support negotiation of session content or media encodings: this is viewed as outside the scope of session description.

This memo obsoletes [RFC4566]. The changes relative to [RFC4566] are outlined in Section 10 of this memo.

2. Glossary of Terms

The following terms are used in this document and have specific meaning within the context of this document.

Session Description: A well-defined format for conveying sufficient information to discover and participate in a multimedia session.

Media Description: A Media Description contains the information needed for one party to establish an application layer network protocol connection to another party. It starts with an "m=" line and is terminated by either the next "m=" line or by the end of the session description.

Session-level Section: This refers to the parts that are not media descriptions, whereas the session description refers to the whole body that includes the session-level section and the media description(s).

The terms "multimedia conference" and "multimedia session" are used in this document as defined in [RFC7656]. The terms "session" and "multimedia session" are used interchangeably in this document.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Examples of SDP Usage

3.1. Session Initiation

The Session Initiation Protocol (SIP) [RFC3261] is an application-layer control protocol for creating, modifying, and terminating sessions such as Internet multimedia conferences, Internet telephone calls, and multimedia distribution. The SIP messages used to create sessions carry session descriptions that allow participants to agree on a set of compatible media types [RFC6838]. These session descriptions are commonly formatted using SDP. When used with SIP, the offer/answer model [RFC3264] provides a limited framework for negotiation using SDP.

3.2. Streaming Media

The Real Time Streaming Protocol (RTSP) [RFC7826], is an application-level protocol for control over the delivery of data with real-time properties. RTSP provides an extensible framework to enable controlled, on-demand delivery of real-time data, such as audio and video. An RTSP client and server negotiate an appropriate set of parameters for media delivery, partially using SDP syntax to describe those parameters.

3.3. Email and the World Wide Web

Alternative means of conveying session descriptions include electronic mail and the World Wide Web (WWW). For both email and WWW distribution, the media type "application/sdp" is used. This enables the automatic launching of applications for participation in the session from the WWW client or mail reader in a standard manner.

Note that descriptions of multicast sessions sent only via email or the WWW do not have the property that the receiver of a session description can necessarily receive the session because the multicast sessions may be restricted in scope, and access to the WWW server or reception of email is possible outside this scope.

3.4. Multicast Session Announcement

In order to assist the advertisement of multicast multimedia conferences and other multicast sessions, and to communicate the relevant session setup information to prospective participants, a distributed session directory may be used. An instance of such a session directory periodically sends packets containing a description of the session to a well-known multicast group. These advertisements are received by other session directories such that potential remote

participants can use the session description to start the tools required to participate in the session.

One protocol used to implement such a distributed directory is the SAP [RFC2974]. SDP provides the recommended session description format for such session announcements.

4. Requirements and Recommendations

The purpose of SDP is to convey information about media streams in multimedia sessions to allow the recipients of a session description to participate in the session. SDP is primarily intended for use with Internet protocols, although it is sufficiently general that it can describe multimedia conferences in other network environments. Media streams can be many-to-many. Sessions need not be continually active.

Thus far, multicast-based sessions on the Internet have differed from many other forms of conferencing in that anyone receiving the traffic can join the session (unless the session traffic is encrypted). In such an environment, SDP serves two primary purposes. It is a means to communicate the existence of a session, and it is a means to convey sufficient information to enable joining and participating in the session. In a unicast environment, only the latter purpose is likely to be relevant.

An SDP description includes the following:

- Session name and purpose
- Time(s) the session is active
- The media comprising the session
- Information needed to receive those media (addresses, ports, formats, etc.)

As resources necessary to participate in a session may be limited, some additional information may also be desirable:

- Information about the bandwidth to be used by the session
- Contact information for the person responsible for the session

In general, SDP must convey sufficient information to enable applications to join a session (with the possible exception of encryption keys) and to announce the resources to be used to any non-participants that may need to know. (This latter feature is

primarily useful when SDP is used with a multicast session announcement protocol.)

4.1. Media and Transport Information

An SDP description includes the following media information:

- The type of media (video, audio, etc.)
- The media transport protocol (RTP/UDP/IP, H.320, etc.)
- The format of the media (H.261 video, MPEG video, etc.)

In addition to media format and transport protocol, SDP conveys address and port details. For an IP multicast session, these comprise:

- The multicast group address for media
- The transport port for media

This address and port are the destination address and destination port of the multicast stream, whether being sent, received, or both.

For unicast IP sessions, the following are conveyed:

- The remote address for media
- The remote transport port for media

The semantics of the address and port depend on context. Typically, this SHOULD be the remote address and remote port to which media is to be sent or received. Details may differ based on the network type, address type, protocol and media specified, and whether the SDP is being distributed as an advertisement or negotiated in an offer/answer [RFC3264] exchange. (E.g., Some address types or protocols may not have a notion of port.) Deviating from typical behavior should be done cautiously since this complicates implementations (including middleboxes that must parse the addresses to open Network Address Translation (NAT) or firewall pinholes).

4.2. Timing Information

Sessions may be either bounded or unbounded in time. Whether or not they are bounded, they may be only active at specific times. SDP can convey:

- An arbitrary list of start and stop times bounding the session

- For each bound, repeat times such as "every Wednesday at 10am for one hour"

This timing information is globally consistent, irrespective of local time zone or daylight saving time (see Section 5.9).

4.3. Obtaining Further Information about a Session

A session description could convey enough information to decide whether or not to participate in a session. SDP may include additional pointers in the form of Uniform Resource Identifiers (URIs) [RFC3986] for more information about the session. (Note that use of URIs to indicate remote resources is subject to the security considerations from [RFC3986].)

4.4. Internationalization

The SDP specification recommends the use of the ISO 10646 character set in the UTF-8 encoding [RFC3629] to allow many different languages to be represented. However, to assist in compact representations, SDP also allows other character sets such as [ISO.8859-1.1998] to be used when desired. Internationalization only applies to free-text sub-fields (session name and background information), and not to SDP as a whole.

5. SDP Specification

An SDP description is denoted by the media type "application/sdp" (See Section 8).

An SDP description is entirely textual. SDP field names and attribute names use only the US-ASCII subset of UTF-8 [RFC3629], but textual fields and attribute values MAY use the full ISO 10646 character set in UTF-8 encoding, or some other character set defined by the "a=charset:" attribute. Field and attribute values that use the full UTF-8 character set are never directly compared, hence there is no requirement for UTF-8 normalization. The textual form, as opposed to a binary encoding such as ASN.1 or XDR, was chosen to enhance portability, to enable a variety of transports to be used, and to allow flexible, text-based toolkits to be used to generate and process session descriptions. However, since SDP may be used in environments where the maximum permissible size of a session description is limited, the encoding is deliberately compact. Also, since descriptions may be transported via very unreliable means or damaged by an intermediate caching server, the encoding was designed with strict order and formatting rules so that most errors would result in malformed session descriptions that could be detected easily and discarded.

An SDP description consists of a number of lines of text of the form:

<type>=<value>

where <type> is exactly one case-significant character and <value> is structured text whose format depends on <type>. In general, <value> is either a number of sub-fields delimited by a single space character or a free format string, and is case-significant unless a specific field defines otherwise. Whitespace separators are not used on either side of the "=" sign, however, the value can contain a leading whitespace as part of its syntax, i.e., that whitespace is part of the value.

An SDP description MUST conform to the syntax defined in Section 9. The following is an overview of the syntax:

An SDP description consists of a session-level section followed by zero or more media descriptions. The session-level section starts with a "v=" line and continues to the first media description (or the end of the whole description, whichever comes first). Each media description starts with an "m=" line and continues to the next media description or the end of the whole session description, whichever comes first. In general, session-level values are the default for all media unless overridden by an equivalent media-level value.

Some lines in each description are required and some are optional, but when present must appear in exactly the order given here. (The fixed order greatly enhances error detection and allows for a simple parser). In the following overview optional items are marked with a "*".

```
      Session description
         v=  (protocol version)
         o=  (originator and session identifier)
         s=  (session name)
         i=* (session information)
         u=* (URI of description)
         e=* (email address)
         p=* (phone number)
         c=* (connection information -- not required if included in
              all media descriptions)
         b=* (zero or more bandwidth information lines)
         One or more time descriptions:
           ("t=", "r=" and "z=" lines; see below)
         k=* (obsolete)
         a=* (zero or more session attribute lines)
         Zero or more media descriptions

      Time description
         t=  (time the session is active)
         r=* (zero or more repeat times)
         z=* (optional time zone offset line)

      Media description, if present
         m=  (media name and transport address)
         i=* (media title)
         c=* (connection information -- optional if included at
              session level)
         b=* (zero or more bandwidth information lines)
         k=* (obsolete)
         a=* (zero or more media attribute lines)
```

The set of type letters is deliberately small and not intended to be extensible -- an SDP parser MUST completely ignore or reject any session description that contains a type letter that it does not understand. The attribute mechanism ("a=" described below) is the primary means for extending SDP and tailoring it to particular applications or media. Some attributes (the ones listed in Section 6 of this memo) have a defined meaning, but others may be added on a media- or session-specific basis. (Attribute scopes in addition to media-specific and session-specific may also be defined in extensions to this document. E.g., [RFC5576], [I-D.ietf-mmusic-data-channel-sdpneg].) An SDP parser MUST ignore any attribute it doesn't understand.

An SDP description may contain URIs that reference external content in the "u=", "k=", and "a=" lines. These URIs may be dereferenced in some cases, making the session description non-self-contained.

The connection ("c=") information in the session-level section applies to all the media descriptions of that session unless overridden by connection information in the media description. For instance, in the example below, each audio media description behaves as if it were given a "c=IN IP4 198.51.100.1".

An example SDP description is:

```
v=0
o=jdoe 3724394400 3724394405 IN IP4 198.51.100.1
s=Call to John Smith
i=SDP Offer #1
u=http://www.jdoe.example.com/home.html
e=Jane Doe <jane@jdoe.example.com>
p=+1 617 555-6011
c=IN IP4 198.51.100.1
t=0 0
m=audio 49170 RTP/AVP 0
m=audio 49180 RTP/AVP 0
m=video 51372 RTP/AVP 99
c=IN IP6 2001:db8::2
a=rtpmap:99 h263-1998/90000
```

Text-containing fields such as the session-name-field and information-field are octet strings that may contain any octet with the exceptions of 0x00 (Nul), 0x0a (ASCII newline), and 0x0d (ASCII carriage return). The sequence CRLF (0x0d0a) is used to end a line, although parsers SHOULD be tolerant and also accept lines terminated with a single newline character. If the "a=charset" attribute is not present, these octet strings MUST be interpreted as containing ISO-10646 characters in UTF-8 encoding. When the "a=charset" attribute is present the session-name-field, information-field, and some attribute fields are interpreted according to the selected character set.

A session description can contain domain names in the "o=", "u=", "e=", "c=", and "a=" lines. Any domain name used in SDP MUST comply with [RFC1034] and [RFC1035]. Internationalized domain names (IDNs) MUST be represented using the ASCII Compatible Encoding (ACE) form defined in [RFC5890] and MUST NOT be directly represented in UTF-8 or any other encoding (this requirement is for compatibility with [RFC2327] and other early SDP-related standards, which predate the development of internationalized domain names).

5.1. Protocol Version ("v=")

```
v=0
```

The "v=" line (version-field) gives the version of the Session Description Protocol. This memo defines version 0. There is no minor version number.

5.2. Origin ("o=")

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

The "o=" line (origin-field) gives the originator of the session (her username and the address of the user's host) plus a session identifier and version number:

<username> is the user's login on the originating host, or it is "-" if the originating host does not support the concept of user IDs. The <username> MUST NOT contain spaces.

<sess-id> is a numeric string such that the tuple of <username>, <sess-id>, <nettype>, <addrtype>, and <unicast-address> forms a globally unique identifier for the session. The method of <sess-id> allocation is up to the creating tool, but a timestamp, in seconds since January 1, 1900 UTC, is recommended to ensure uniqueness.

<sess-version> is a version number for this session description. Its usage is up to the creating tool, so long as <sess-version> is increased when a modification is made to the session description. Again, as with <sess-id> it is RECOMMENDED that a timestamp be used.

<nettype> is a text string giving the type of network. Initially "IN" is defined to have the meaning "Internet", but other values MAY be registered in the future (see Section 8).

<addrtype> is a text string giving the type of the address that follows. Initially "IP4" and "IP6" are defined, but other values MAY be registered in the future (see Section 8).

<unicast-address> is an address of the machine from which the session was created. For an address type of IP4, this is either a fully qualified domain name of the machine or the dotted-decimal representation of an IP version 4 address of the machine. For an address type of IP6, this is either a fully qualified domain name of the machine or the address of the machine represented as

specified in Section 4 of [RFC5952]. For both IP4 and IP6, the fully qualified domain name is the form that SHOULD be given unless this is unavailable, in which case a globally unique address MAY be substituted.

In general, the "o=" line serves as a globally unique identifier for this version of the session description, and the sub-fields excepting the version, taken together identify the session irrespective of any modifications.

For privacy reasons, it is sometimes desirable to obfuscate the username and IP address of the session originator. If this is a concern, an arbitrary <username> and private <unicast-address> MAY be chosen to populate the "o=" line, provided that these are selected in a manner that does not affect the global uniqueness of the field.

5.3. Session Name ("s=")

s=<session name>

The "s=" line (session-name-field) is the textual session name. There MUST be one and only one "s=" line per session description. The "s=" line MUST NOT be empty. If a session has no meaningful name, then "s= " or "s=-" (i.e., a single space or dash as the session name) is RECOMMENDED. If a session-level "a=charset" attribute is present, it specifies the character set used in the "s=" field. If a session-level "a=charset" attribute is not present, the "s=" field MUST contain ISO 10646 characters in UTF-8 encoding.

5.4. Session Information ("i=")

i=<session information>

The "i=" line (information-field) provides textual information about the session. There can be at most one session-level "i=" line per session description, and at most one "i=" line in each media description. Unless a media-level "i=" line is provided, the session-level "i=" line applies to that media description. If the "a=charset" attribute is present, it specifies the character set used in the "i=" line. If the "a=charset" attribute is not present, the "i=" line MUST contain ISO 10646 characters in UTF-8 encoding.

At most one "i=" line can be used for each media description. In media definitions, "i=" lines are primarily intended for labelling media streams. As such, they are most likely to be useful when a single session has more than one distinct media stream of the same media type. An example would be two different whiteboards, one for slides and one for feedback and questions.

The "i=" line is intended to provide a free-form human-readable description of the session or the purpose of a media stream. It is not suitable for parsing by automata.

5.5. URI ("u=")

```
u=<uri>
```

The "u=" line (uri-field) provides a URI (Uniform Resource Identifier) [RFC3986]. The URI should be a pointer to additional human readable information about the session. This line is OPTIONAL. No more than one "u=" line is allowed per session description.

5.6. Email Address and Phone Number ("e=" and "p=")

```
e=<email-address>
p=<phone-number>
```

The "e=" line (email-field) and "p=" line (phone-field) specify contact information for the person responsible for the session. This is not necessarily the same person that created the session description.

Inclusion of an email address or phone number is OPTIONAL.

If an email address or phone number is present, it MUST be specified before the first media description. More than one email or phone line can be given for a session description.

Phone numbers SHOULD be given in the form of an international public telecommunication number (see ITU-T Recommendation E.164 [E164]) preceded by a "+". Spaces and hyphens may be used to split up a phone-field to aid readability if desired. For example:

```
p=+1 617 555-6011
```

Both email addresses and phone numbers can have an OPTIONAL free text string associated with them, normally giving the name of the person who may be contacted. This MUST be enclosed in parentheses if it is present. For example:

```
e=j.doe@example.com (Jane Doe)
```

The alternative [RFC5322] name quoting convention is also allowed for both email addresses and phone numbers. For example:

```
e=Jane Doe <j.doe@example.com>
```

The free text string SHOULD be in the ISO-10646 character set with UTF-8 encoding, or alternatively in ISO-8859-1 or other encodings if the appropriate session-level "a=charset" attribute is set.

5.7. Connection Information ("c=")

c=<nettype> <addrtype> <connection-address>

The "c=" line (connection-field) contains information necessary to establish a network connection.

A session description MUST contain either at least one "c=" line in each media description or a single "c=" line at the session level. It MAY contain a single session-level "c=" line and additional media-level "c=" line(s) per-media-description, in which case the media-level values override the session-level settings for the respective media.

The first sub-field ("<nettype>") is the network type, which is a text string giving the type of network. Initially, "IN" is defined to have the meaning "Internet", but other values MAY be registered in the future (see Section 8).

The second sub-field ("<addrtype>") is the address type. This allows SDP to be used for sessions that are not IP based. This memo only defines IP4 and IP6, but other values MAY be registered in the future (see Section 8).

The third sub-field ("<connection-address>") is the connection address. Additional sub-fields MAY be added after the connection address depending on the value of the <addrtype> sub-field.

When the <addrtype> is IP4 or IP6, the connection address is defined as follows:

- If the session is multicast, the connection address will be an IP multicast group address. If the session is not multicast, then the connection address contains the unicast IP address of the expected data source, data relay or data sink as determined by additional attribute-fields. It is not expected that unicast addresses will be given in a session description that is communicated by a multicast announcement, though this is not prohibited.

- Sessions using an IP4 multicast connection address MUST also have a time to live (TTL) value present in addition to the multicast address. The TTL and the address together define the scope with which multicast packets sent in this session will be sent. TTL

values MUST be in the range 0-255. Although the TTL MUST be specified, its use to scope multicast traffic is deprecated; applications SHOULD use an administratively scoped address instead.

The TTL for the session is appended to the address using a slash as a separator. An example is:

```
c=IN IP4 233.252.0.1/127
```

IP6 multicast does not use TTL scoping, and hence the TTL value MUST NOT be present for IP6 multicast. It is expected that IPv6 scoped addresses will be used to limit the scope of multimedia conferences.

Hierarchical or layered encoding schemes are data streams where the encoding from a single media source is split into a number of layers. The receiver can choose the desired quality (and hence bandwidth) by only subscribing to a subset of these layers. Such layered encodings are normally transmitted in multiple multicast groups to allow multicast pruning. This technique keeps unwanted traffic from sites only requiring certain levels of the hierarchy. For applications requiring multiple multicast groups, we allow the following notation to be used for the connection address:

```
<base multicast address>[/<ttl>]/<number of addresses>
```

If the number of addresses is not given, it is assumed to be one. Multicast addresses so assigned are contiguously allocated above the base address, so that, for example:

```
c=IN IP4 233.252.0.1/127/3
```

would state that addresses 233.252.0.1, 233.252.0.2, and 233.252.0.3 are to be used with a TTL of 127. This is semantically identical to including multiple "c=" lines in a media description:

```
c=IN IP4 233.252.0.1/127
c=IN IP4 233.252.0.2/127
c=IN IP4 233.252.0.3/127
```

Similarly, an IPv6 example would be:

```
c=IN IP6 ff00::db8:0:101/3
```

which is semantically equivalent to:

```
c=IN IP6 ff00::db8:0:101
c=IN IP6 ff00::db8:0:102
c=IN IP6 ff00::db8:0:103
```

(remembering that the TTL sub-field is not present in IP6 multicast).

Multiple addresses or "c=" lines MAY be specified on a per media description basis only if they provide multicast addresses for different layers in a hierarchical or layered encoding scheme. Multiple addresses or "c=" lines MUST NOT be specified at session level.

The slash notation for multiple addresses described above MUST NOT be used for IP unicast addresses.

5.8. Bandwidth Information ("b=")

```
b=<bwtype>:<bandwidth>
```

The OPTIONAL "b=" line (bandwidth-field) denotes the proposed bandwidth to be used by the session or media description. The <bwtype> is an alphanumeric modifier giving the meaning of the <bandwidth> figure. Two values are defined in this specification, but other values MAY be registered in the future (see Section 8 and [RFC3556], [RFC3890]):

CT If the bandwidth of a session is different from the bandwidth implicit from the scope, a "b=CT:..." line SHOULD be supplied for the session giving the proposed upper limit to the bandwidth used (the "conference total" bandwidth). Similarly, if the bandwidth of bundled media streams [I-D.ietf-mmusic-sdp-bundle-negotiation] in an "m=" line is different from the implicit value from the scope, a "b=CT:..." line SHOULD be supplied in the media level. The primary purpose of this is to give an approximate idea as to whether two or more sessions (or bundled media streams) can coexist simultaneously. Note that CT gives a total bandwidth figure for all the media at all endpoints.

   The Mux Category for CT is NORMAL. This is discussed in [I-D.ietf-mmusic-sdp-mux-attributes].

AS The bandwidth is interpreted to be application specific (it will be the application's concept of maximum bandwidth). Normally, this will coincide with what is set on the application's "maximum bandwidth" control if applicable. For RTP-based applications, AS gives the RTP "session bandwidth" as defined in Section 6.2 of [RFC3550]. Note that AS gives a bandwidth figure for a single

media at a single endpoint, although there may be many endpoints sending simultaneously.

The Mux Category for AS is SUM. This is discussed in [I-D.ietf-mmusic-sdp-mux-attributes].

[RFC4566] defined an "X-" prefix for <bwtype> names. This was intended for experimental purposes only. For example:

b=X-YZ:128

Use of the "X-" prefix is NOT RECOMMENDED. Instead new (non "X-" prefix) <bwtype> names SHOULD be defined, and then MUST be registered with IANA in the standard namespace. SDP parsers MUST ignore bandwidth-fields with unknown <bwtype> names. The <bwtype> names MUST be alphanumeric and, although no length limit is given, it is recommended that they be short.

The <bandwidth> is interpreted as kilobits per second by default (including the transport and network-layer but not the link-layer overhead). The definition of a new <bwtype> modifier MAY specify that the bandwidth is to be interpreted in some alternative unit (the "CT" and "AS" modifiers defined in this memo use the default units).

5.9. Time Active ("t=")

t=<start-time> <stop-time>

A "t=" line (time-field) begins a time description that specifies the start and stop times for a session. Multiple time descriptions MAY be used if a session is active at multiple irregularly spaced times; each additional time description specifies additional periods of time for which the session will be active. If the session is active at regular repeat times, a repeat description, begun by an "r=" line (see below) can be included following the time-field -- in which case the time-field specifies the start and stop times of the entire repeat sequence.

The following example specifies two active intervals:

```
t=3724394400 3724398000 ; Mon 8-Jan-2018 10:00-11:00 UTC
t=3724484400 3724488000 ; Tue 9-Jan-2018 11:00-12:00 UTC
```

The first and second sub-fields of the time-field give the start and stop times, respectively, for the session. These are the decimal representation of time values in seconds since January 1, 1900 UTC. To convert these values to UNIX time (UTC), subtract decimal 2208988800.

Some time representations will wrap in the year 2036. Because SDP uses an arbitrary length decimal representation, it does not have this issue. Implementations of SDP need to be prepared to handle these larger values.

If the <stop-time> is set to zero, then the session is not bounded, though it will not become active until after the <start-time>. If the <start-time> is also zero, the session is regarded as permanent.

User interfaces SHOULD strongly discourage the creation of unbounded and permanent sessions as they give no information about when the session is actually going to terminate, and so make scheduling difficult.

The general assumption may be made, when displaying unbounded sessions that have not timed out to the user, that an unbounded session will only be active until half an hour from the current time or the session start time, whichever is the later. If behavior other than this is required, an end-time SHOULD be given and modified as appropriate when new information becomes available about when the session should really end.

Permanent sessions may be shown to the user as never being active unless there are associated repeat times that state precisely when the session will be active.

5.10. Repeat Times ("r=")

r=<repeat interval> <active duration> <offsets from start-time>

An"r=" line (repeat-field) specifies repeat times for a session. If needed to express complex schedules, multiple repeat-fields may be included. For example, if a session is active at 10am on Monday and 11am on Tuesday for one hour each week for three months, then the <start-time> in the corresponding "t=" line would be the representation of 10am on the first Monday, the <repeat interval> would be 1 week, the <active duration> would be 1 hour, and the offsets would be zero and 25 hours. The corresponding "t=" line stop time would be the representation of the end of the last session three months later. By default, all sub-fields are in seconds, so the "r=" and "t=" lines might be the following:

```
t=3724394400 3730536000 ; Mon 8-Jan-2018 10:00-11:00 UTC
                        ; Tues 20-Mar-2018 12:00 UTC
r=604800 3600 0 90000   ; 1 week, 1 hour, zero, 25 hours
```

To make the description more compact, times may also be given in units of days, hours, or minutes. The syntax for these is a number

immediately followed by a single case-sensitive character. Fractional units are not allowed -- a smaller unit should be used instead. The following unit specification characters are allowed:

d - days (86400 seconds)
h - hours (3600 seconds)
m - minutes (60 seconds)
s - seconds (allowed for completeness)

Thus, the above repeat-field could also have been written:

r=7d 1h 0 25h

Monthly and yearly repeats cannot be directly specified with a single SDP repeat time; instead, separate time-descriptions should be used to explicitly list the session times.

5.11. Time Zone Adjustment ("z=")

z=<adjustment time> <offset> <adjustment time> <offset> ....

A "z=" line (zone-field) is an optional modifier to the repeat-fields it immediately follows. It does not apply to any other fields.

To schedule a repeated session that spans a change from daylight saving time to standard time or vice versa, it is necessary to specify offsets from the base time. This is required because different time zones change time at different times of day, different countries change to or from daylight saving time on different dates, and some countries do not have daylight saving time at all.

Thus, in order to schedule a session that is at the same time winter and summer, it must be possible to specify unambiguously by whose time zone a session is scheduled. To simplify this task for receivers, we allow the sender to specify the time (represented as seconds since January 1, 1900 UTC) that a time zone adjustment happens and the offset from the time when the session was first scheduled. The "z=" line allows the sender to specify a list of these adjustment times and offsets from the base time.

An example might be the following:

```
t=3724394400 3754123200       ; Mon 8-Jan-2018 10:00 UTC
                              ; Tues 18-Dec-2018 12:00 UTC
r=604800 3600 0 90000         ; 1 week, 1 hour, zero, 25 hours
z=3730928400 -1h 3749680800 0 ; Sun 25-Mar-2018 1:00 UTC,
                              ; offset 1 hour,
                              ; Sun 28-Oct-2018 2:00 UTC,
                              ; no offset
```

This specifies that at time 3730928400 (Sun 25-Mar-2018 1:00 UTC, the onset of British Summer Time) the time base by which the session's repeat times are calculated is shifted back by 1 hour, and that at time 3749680800 (Sun 28-Oct-2018 2:00 UTC, the end of British Summer Time) the session's original time base is restored. Adjustments are always relative to the specified start time -- they are not cumulative.

If a session is likely to last several years, it is expected that the session description will be modified periodically rather than transmit several years' worth of adjustments in one session description.

5.12. Encryption Keys ("k=")

```
k=<method>
k=<method>:<encryption key>
```

The "k=" line (key-field) is obsolete and MUST NOT be used. It is included in this document for legacy reasons. One MUST NOT include a "k=" line in an SDP, and MUST discard it if it is received in an SDP.

5.13. Attributes ("a=")

```
a=<attribute>
a=<attribute>:<value>
```

Attributes are the primary means for extending SDP. Attributes may be defined to be used as "session-level" attributes, "media-level" attributes, or both. (Attribute scopes in addition to media- and session- level may also be defined in extensions to this document. E.g., [RFC5576], [I-D.ietf-mmusic-data-channel-sdpneg].)

A media description may contain any number of "a=" lines (attribute-fields) that are media description specific. These are referred to as "media-level" attributes and add information about the media description. Attribute-fields can also be added before the first media description; these "session-level" attributes convey additional information that applies to the session as a whole rather than to individual media descriptions.

Attribute-fields may be of two forms:

- A property attribute is simply of the form "a=<attribute>". These are binary attributes, and the presence of the attribute conveys that the attribute is a property of the session. An example might be "a=recvonly".

- A value attribute is of the form "a=<attribute>:<value>". For example, a whiteboard could have the value attribute "a=orient:landscape"

Attribute interpretation depends on the media tool being invoked. Thus receivers of session descriptions should be configurable in their interpretation of session descriptions in general and of attributes in particular.

Attribute names MUST use the US-ASCII subset of ISO-10646/UTF-8.

Attribute values are octet strings, and MAY use any octet value except 0x00 (Nul), 0x0A (LF), and 0x0D (CR). By default, attribute values are to be interpreted as in ISO-10646 character set with UTF-8 encoding. Unlike other text fields, attribute values are NOT normally affected by the "charset" attribute as this would make comparisons against known values problematic. However, when an attribute is defined, it can be defined to be charset dependent, in which case its value should be interpreted in the session charset rather than in ISO-10646.

Attributes MUST be registered with IANA (see Section 8). If an attribute is received that is not understood, it MUST be ignored by the receiver.

5.14. Media Descriptions ("m=")

```
m=<media> <port> <proto> <fmt> ...
```

A session description may contain a number of media descriptions. Each media description starts with an "m=" line (media-field) and is terminated by either the next "m=" line or by the end of the session description. A media-field has several sub-fields:

<media> is the media type. This document defines the values "audio", "video", "text", "application", and "message". This list is extended by other memos and may be further extended by additional memos registering media types in the future (see Section 8). For example, [RFC6466] defined the "image" media type.

<port> is the transport port to which the media stream is sent. The meaning of the transport port depends on the network being used as specified in the relevant "c=" line, and on the transport protocol defined in the <proto> sub-field of the media-field. Other ports used by the media application (such as the RTP Control Protocol (RTCP) port [RFC3550]) MAY be derived algorithmically from the base media port or MAY be specified in a separate attribute (for example, "a=rtcp:" as defined in [RFC3605]).

If non-contiguous ports are used or if they don't follow the parity rule of even RTP ports and odd RTCP ports, the "a=rtcp:" attribute MUST be used. Applications that are requested to send media to a <port> that is odd and where the "a=rtcp:" is present MUST NOT subtract 1 from the RTP port: that is, they MUST send the RTP to the port indicated in <port> and send the RTCP to the port indicated in the "a=rtcp" attribute.

For applications where hierarchically encoded streams are being sent to a unicast address, it may be necessary to specify multiple transport ports. This is done using a similar notation to that used for IP multicast addresses in the "c=" line:

```
m=<media> <port>/<number of ports> <proto> <fmt> ...
```

In such a case, the ports used depend on the transport protocol. For RTP, the default is that only the even-numbered ports are used for data with the corresponding one-higher odd ports used for the RTCP belonging to the RTP session, and the <number of ports> denoting the number of RTP sessions. For example:

```
m=video 49170/2 RTP/AVP 31
```

would specify that ports 49170 and 49171 form one RTP/RTCP pair and 49172 and 49173 form the second RTP/RTCP pair. RTP/AVP is the transport protocol and 31 is the format (see below).

This document does not include a mechanism for declaring hierarchically encoded streams using non-contiguous ports. (There is currently no attribute defined that can accomplish this. The "a=rtcp:" defined in [RFC3605] does not handle hierarchical encoding.) If a need arises to declare non-contiguous ports then it will be necessary to define a new attribute to do so.

If multiple addresses are specified in the "c=" line and multiple ports are specified in the "m=" line, a one-to-one mapping from port to the corresponding address is implied. For example:

```
m=video 49170/2 RTP/AVP 31
c=IN IP4 233.252.0.1/127/2
```

would imply that address 233.252.0.1 is used with ports 49170 and 49171, and address 233.252.0.2 is used with ports 49172 and 49173.

The mapping is similar if multiple addresses are specified using multiple "c=" lines. For example:

```
m=video 49170/2 RTP/AVP 31
c=IN IP6 ff00::db8:0:101
c=IN IP6 ff00::db8:0:102
```

would imply that address ff00::db8:0:101 is used with ports 49170 and 49171, and address ff00::db8:0:102 is used with ports 49172 and 49173.

This document gives no meaning to assigning the same media address to multiple media-descriptions. Doing so does not implicitly group those media-descriptions in any way. An explicit grouping framework (for example, [RFC5888]) should instead be used to express the intended semantics. For instance, see [I-D.ietf-mmusic-sdp-bundle-negotiation].

<proto> is the transport protocol. The meaning of the transport protocol is dependent on the address type sub-field in the relevant "c=" line. Thus a "c=" line with an address type of IP4 indicates that the transport protocol runs over IPv4. The following transport protocols are defined, but may be extended through registration of new protocols with IANA (see Section 8):

* udp: denotes that the data is transported directly in UDP with no additional framing.

* RTP/AVP: denotes RTP [RFC3550] used under the RTP Profile for Audio and Video Conferences with Minimal Control [RFC3551] running over UDP.

* RTP/SAVP: denotes the Secure Real-time Transport Protocol [RFC3711] running over UDP.

The main reason to specify the transport protocol in addition to the media format is that the same standard media formats may be carried over different transport protocols even when the network protocol is the same -- a historical example is VAT (MBone's popular multimedia audio tool) Pulse Code Modulation (PCM) audio and RTP PCM audio; another might be TCP/RTP PCM audio. In

addition, relays and monitoring tools that are transport-protocol-specific but format-independent are possible.

<fmt> is a media format description. The fourth and any subsequent sub-fields describe the format of the media. The interpretation of the media format depends on the value of the <proto> sub-field.

If the <proto> sub-field is "RTP/AVP" or "RTP/SAVP" the <fmt> sub-fields contain RTP payload type numbers. When a list of payload type numbers is given, this implies that all of these payload formats MAY be used in the session, but the first of these formats SHOULD be used as the default format for the session. For dynamic payload type assignments the "a=rtpmap:" attribute (see Section 6) SHOULD be used to map from an RTP payload type number to a media encoding name that identifies the payload format. The "a=fmtp:" attribute MAY be used to specify format parameters (see Section 6).

If the <proto> sub-field is "udp" the <fmt> sub-fields MUST reference a media type describing the format under the "audio", "video", "text", "application", or "message" top-level media types. The media type registration SHOULD define the packet format for use with UDP transport.

For media using other transport protocols, the <fmt> sub-field is protocol specific. Rules for interpretation of the <fmt> sub-field MUST be defined when registering new protocols (see Section 8.2.2).

Section 3 of [RFC4855] states that the payload format (encoding) names defined in the RTP Profile are commonly shown in upper case, while media subtype names are commonly shown in lower case. It also states that both of these names are case-insensitive in both places, similar to parameter names which are case-insensitive both in media type strings and in the default mapping to the SDP a=fmtp attribute.

6. SDP Attributes

The following attributes are defined. Since application writers may add new attributes as they are required, this list is not exhaustive. Registration procedures for new attributes are defined in Section 8.2.4. Syntax is provided using ABNF [RFC7405] with some of the rules defined further in Section 9.

6.1. cat (category)

Name: cat

Value: cat-value

Usage Level: session

Charset Dependent: no

Syntax:

```
cat-value = category
category = non-ws-string
```

Example:

```
a=cat:foo.bar
```

This attribute gives the dot-separated hierarchical category of the session. This is to enable a receiver to filter unwanted sessions by category. There is no central registry of categories. This attribute is obsolete and SHOULD NOT be used. It SHOULD be ignored if received.

6.2. keywds (keywords)

Name: keywds

Value: keywds-value

Usage Level: session

Charset Dependent: yes

Syntax:

```
keywds-value = keywords
keywords = text
```

Example:

```
a=keywds:SDP session description protocol
```

Like the cat attribute, this was intended to assist identifying wanted sessions at the receiver. This allows a receiver to select interesting sessions based on keywords describing the purpose of the session; there is no central registry of keywords. Its value should

be interpreted in the charset specified for the session description if one is specified, or by default in ISO 10646/UTF-8. This attribute is obsolete and SHOULD NOT be used. It SHOULD be ignored if received.

6.3. tool

Name: tool

Value: tool-value

Usage Level: session

Charset Dependent: no

Syntax:

```
tool-value = tool-name-and-version
tool-name-and-version = text
```

Example:

```
a=tool:foobar V3.2
```

This gives the name and version number of the tool used to create the session description.

6.4. ptime (packet time)

Name: ptime

Value: ptime-value

Usage Level: media

Charset Dependent: no

Syntax:

```
ptime-value = non-zero-int-or-real
```

Example:

```
a=ptime:20
```

This gives the length of time in milliseconds represented by the media in a packet. This is probably only meaningful for audio data, but may be used with other media types if it makes sense. It should

not be necessary to know ptime to decode RTP or vat audio, and it is intended as a recommendation for the encoding/packetization of audio.

6.5. maxptime (maximum packet time)

Name: maxptime

Value: maxptime-value

Usage Level: media

Charset Dependent: no

Syntax:

```
maxptime-value = non-zero-int-or-real
```

Example:

```
a=maxptime:20
```

This gives the maximum amount of media that can be encapsulated in each packet, expressed as time in milliseconds. The time SHALL be calculated as the sum of the time the media present in the packet represents. For frame-based codecs, the time SHOULD be an integer multiple of the frame size. This attribute is probably only meaningful for audio data, but may be used with other media types if it makes sense. Note that this attribute was introduced after [RFC2327], and non-updated implementations will ignore this attribute.

6.6. rtpmap

Name: rtpmap

Value: rtpmap-value

Usage Level: media

Charset Dependent: no

Syntax:

```
rtpmap-value = payload-type SP encoding-name
  "/" clock-rate [ "/" encoding-params ]
payload-type = zero-based-integer
encoding-name = token
clock-rate = integer
encoding-params = channels
channels = integer
```

This attribute maps from an RTP payload type number (as used in an "m=" line) to an encoding name denoting the payload format to be used. It also provides information on the clock rate and encoding parameters. Note that the payload type number is indicated in a 7-bit field, limiting the values to inclusively between 0 and 127.

Although an RTP profile can make static assignments of payload type numbers to payload formats, it is more common for that assignment to be done dynamically using "a=rtpmap:" attributes. As an example of a static payload type, consider u-law PCM coded single-channel audio sampled at 8 kHz. This is completely defined in the RTP Audio/Video profile as payload type 0, so there is no need for an "a=rtpmap:" attribute, and the media for such a stream sent to UDP port 49232 can be specified as:

```
m=audio 49232 RTP/AVP 0
```

An example of a dynamic payload type is 16-bit linear encoded stereo audio sampled at 16 kHz. If we wish to use the dynamic RTP/AVP payload type 98 for this stream, additional information is required to decode it:

```
m=audio 49232 RTP/AVP 98
a=rtpmap:98 L16/16000/2
```

Up to one rtpmap attribute can be defined for each media format specified. Thus, we might have the following:

```
m=audio 49230 RTP/AVP 96 97 98
a=rtpmap:96 L8/8000
a=rtpmap:97 L16/8000
a=rtpmap:98 L16/11025/2
```

RTP profiles that specify the use of dynamic payload types MUST define the set of valid encoding names and/or a means to register encoding names if that profile is to be used with SDP. The "RTP/AVP" and "RTP/SAVP" profiles use media subtypes for encoding names, under

the top-level media type denoted in the "m=" line. In the example above, the media types are "audio/L8" and "audio/L16".

For audio streams, encoding-params indicates the number of audio channels. This parameter is OPTIONAL and may be omitted if the number of channels is one, provided that no additional parameters are needed.

For video streams, no encoding parameters are currently specified.

Additional encoding parameters MAY be defined in the future, but codec-specific parameters SHOULD NOT be added. Parameters added to an "a=rtpmap:" attribute SHOULD only be those required for a session directory to make the choice of appropriate media to participate in a session. Codec-specific parameters should be added in other attributes (for example, "a=fmtp:").

Note: RTP audio formats typically do not include information about the number of samples per packet. If a non-default (as defined in the RTP Audio/Video Profile [RFC3551]) packetization is required, the "ptime" attribute is used as given above.

6.7. Media Direction Attributes

At most one occurrence of recvonly, sendrecv, sendonly, or inactive MAY appear at session level, and at most one MAY appear in each media description.

If any one of these appears in a media description then it applies for that media description. If none appear in a media description then the one from session level, if any, applies to that media description.

If none of the media direction attributes is present at either session level or media level, "sendrecv" SHOULD be assumed as the default.

Within the following SDP example, the "sendrecv" attribute applies to the first audio media and the "inactive" attribute applies to the others.

```
v=0
o=jdoe 3724395000 3724395001 IN IP6 2001:db8::1
s=-
c=IN IP6 2001:db8::1
t=0 0
a=inactive
m=audio 49170 RTP/AVP 0
a=sendrecv
m=audio 49180 RTP/AVP 0
m=video 51372 RTP/AVP 99
a=rtpmap:99 h263-1998/90000
```

6.7.1. recvonly (receive-only)

Name: recvonly

Value:

Usage Level: session, media

Charset Dependent: no

Example:

```
a=recvonly
```

This specifies that the tools should be started in receive-only mode where applicable. Note that recvonly applies to the media only, not to any associated control protocol. An RTP-based system in recvonly mode MUST still send RTCP packets as described in [RFC3550] Section 6.

6.7.2. sendrecv (send-receive)

Name: sendrecv

Value:

Usage Level: session, media

Charset Dependent: no

Example:

```
a=sendrecv
```

This specifies that the tools should be started in send and receive mode. This is necessary for interactive multimedia conferences with tools that default to receive-only mode.

6.7.3. sendonly (send-only)

Name: sendonly

Value:

Usage Level: session, media

Charset Dependent: no

Example:

```
a=sendonly
```

This specifies that the tools should be started in send-only mode. An example may be where a different unicast address is to be used for a traffic destination than for a traffic source. In such a case, two media descriptions may be used, one sendonly and one recvonly. Note that sendonly applies only to the media, and any associated control protocol (e.g., RTCP) SHOULD still be received and processed as normal.

6.7.4. inactive

Name: inactive

Value:

Usage Level: session, media

Charset Dependent: no

Example:

```
a=inactive
```

This specifies that the tools should be started in inactive mode. This is necessary for interactive multimedia conferences where users can put other users on hold. No media is sent over an inactive media stream. Note that an RTP-based system MUST still send RTCP (if RTCP is used), even if started inactive.

6.8. orient (orientation)

Name: orient

Value: orient-value

Usage Level: media

Charset Dependent: no

Syntax:

```
orient-value = portrait / landscape / seascape
portrait  = %s"portrait"
landscape = %s"landscape"
seascape  = %s"seascape"
   ; NOTE: These names are case-sensitive.
```

Example:

```
a=orient:portrait
```

Normally this is only used for a whiteboard or presentation tool. It specifies the orientation of the workspace on the screen. Permitted values are "portrait", "landscape", and "seascape" (upside-down landscape).

6.9. type (conference type)

Name: type

Value: type-value

Usage Level: session

Charset Dependent: no

Syntax:

```
type-value = conference-type
conference-type = broadcast / meeting / moderated / test /
                  H332
broadcast = %s"broadcast"
meeting   = %s"meeting"
moderated = %s"moderated"
test      = %s"test"
H332      = %s"H332"
   ; NOTE: These names are case-sensitive.
```

Example:

```
a=type:moderated
```

This specifies the type of the multimedia conference. Allowed values are "broadcast", "meeting", "moderated", "test", and "H332". These values have implications for other options that are likely to be appropriate:

- When "a=type:broadcast" is specified, "a=recvonly" is probably appropriate for those connecting.
- When "a=type:meeting" is specified, "a=sendrecv" is likely to be appropriate.
- "a=type:moderated" suggests the use of a floor control tool and that the media tools be started so as to mute new sites joining the multimedia conference.
- Specifying "a=type:H332" indicates that this loosely coupled session is part of an H.332 session as defined in the ITU H.332 specification [ITU.H332.1998]. Media tools should be started using "a=recvonly".
- Specifying "a=type:test" is suggested as a hint that, unless explicitly requested otherwise, receivers can safely avoid displaying this session description to users.

6.10. charset (character set)

Name: charset

Value: charset-value

Usage Level: session

Charset Dependent: no

Syntax:

```
charset-value = <defined in [RFC2978]>
```

This specifies the character set to be used to display the session name and information data. By default, the ISO-10646 character set in UTF-8 encoding is used. If a more compact representation is required, other character sets may be used. For example, the ISO 8859-1 is specified with the following SDP attribute:

```
a=charset:ISO-8859-1
```

The charset specified MUST be one of those registered in the IANA Character Sets registry (http://www.iana.org/assignments/character-sets), such as ISO-8859-1. The character set identifier is a string that MUST be compared against identifiers from the "Name" or "Preferred MIME Name" field of the registry using a case-insensitive comparison. If the identifier is not recognized or not supported, all strings that are affected by it SHOULD be regarded as octet strings.

Charset-dependent fields MUST contain only sequences of bytes that are valid according to the definition of the selected character set. Furthermore, charset-dependent fields MUST NOT contain the bytes 0x00 (Nul), 0x0A (LF), and 0x0d (CR).

6.11. sdplang (SDP language)

Name: sdplang

Value: sdplang-value

Usage Level: session, media

Charset Dependent: no

Syntax:

```
sdplang-value = Language-Tag
; Language-Tag defined in RFC5646
```

Example:

```
a=sdplang:fr
```

Multiple sdplang attributes can be provided either at session or media level if the session description or media use multiple languages.

As a session-level attribute, it specifies the language for the session description (not the language of the media). As a media-level attribute, it specifies the language for any media-level SDP information-field associated with that media (again not the language of the media), overriding any sdplang attributes specified at session level.

In general, sending session descriptions consisting of multiple languages is discouraged. Instead, multiple sesssion descriptions

SHOULD be sent describing the session, one in each language. However, this is not possible with all transport mechanisms, and so multiple sdplang attributes are allowed although NOT RECOMMENDED.

The "sdplang" attribute value must be a single [RFC5646] language tag. An "sdplang" attribute SHOULD be specified when a session is distributed with sufficient scope to cross geographic boundaries, where the language of recipients cannot be assumed, or where the session is in a different language from the locally assumed norm.

6.12. lang (language)

Name: lang

Value: lang-value

Usage Level: session, media

Charset Dependent: no

Syntax:

```
lang-value = Language-Tag
; Language-Tag defined in RFC5646
```

Example:

```
a=lang:de
```

Multiple lang attributes can be provided either at session or media level if the session or media has capabilities in more than one language, in which case the order of the attributes indicates the order of preference of the various languages in the session or media, from most preferred to least preferred.

As a session-level attribute, lang specifies a language capability for the session being described. As a media-level attribute, it specifies a language capability for that media, overriding any session-level language(s) specified.

The "lang" attribute value must be a single [RFC5646] language tag. A "lang" attribute SHOULD be specified when a session is of sufficient scope to cross geographic boundaries where the language of participants cannot be assumed, or where the session has capabilities in languages different from the locally assumed norm.

The "lang" attribute is supposed to be used for setting the initial language(s) used in the session. Events during the session may

influence which language(s) are used, and the participants are not strictly bound to only use the declared languages.

Most real-time use cases start with just one language used, while other cases involve a range of languages, e.g. an interpreted or subtitled session. When more than one 'lang' attribute is specified, the "lang" attribute itself does not provide any information about multiple languages being intended to be used during the session, or if the intention is to only select one of the languages. If needed, a new attribute can be defined and used to indicate such intentions. Without such semantics, it is assumed that for a negotiated session one of the declared languages will be selected and used.

6.13. framerate (frame rate)

Name: framerate

Value: framerate-value

Usage Level: media

Charset Dependent: no

Syntax:

```
framerate-value = non-zero-int-or-real
```

Example:

```
a=framerate:60
```

This gives the maximum video frame rate in frames/sec. It is intended as a recommendation for the encoding of video data. Decimal representations of fractional values are allowed. It is defined only for video media.

6.14. quality

Name: quality

Value: quality-value

Usage Level: media

Charset Dependent: no

Syntax:

```
quality-value = zero-based-integer
```

Example:

```
a=quality:10
```

This gives a suggestion for the quality of the encoding as an integer value. The intention of the quality attribute for video is to specify a non-default trade-off between frame-rate and still-image quality. For video, the value is in the range 0 to 10, with the following suggested meaning:

```
10 - the best still-image quality the compression scheme
     can give.
5  - the default behavior given no quality suggestion.
0  - the worst still-image quality the codec designer
     thinks is still usable.
```

6.15. fmtp (format parameters)

Name: fmtp

Value: fmtp-value

Usage Level: media

Charset Dependent: no

Syntax:

```
fmtp-value = fmt SP format-specific-params
format-specific-params = byte-string
  ; Notes:
  ; - The format parameters are media type parameters and
  ;   need to reflect their syntax.
```

Example:

```
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
```

This attribute allows parameters that are specific to a particular format to be conveyed in a way that SDP does not have to understand them. The format must be one of the formats specified for the media. Format-specific parameters, semicolon separated, may be any set of parameters required to be conveyed by SDP and given unchanged to the

media tool that will use this format. At most one instance of this attribute is allowed for each format.

The fmtp attribute may be used to specify parameters for any protocol and format that defines use of such parameters.

7. Security Considerations

SDP is frequently used with the Session Initiation Protocol [RFC3261] using the offer/answer model [RFC3264] to agree on parameters for unicast sessions. When used in this manner, the security considerations of those protocols apply.

SDP is a session description format that describes multimedia sessions. Entities receiving and acting upon an SDP message SHOULD be aware that a session description cannot be trusted unless it has been obtained by an authenticated and integrity-protected transport protocol from a known and trusted source. Many different transport protocols may be used to distribute session descriptions, and the nature of the authentication and integrity-protection will differ from transport to transport. For some transports, security features are often not deployed. In case a session description has not been obtained in a trusted manner, the endpoint SHOULD exercise care because, among other attacks, the media sessions received may not be the intended ones, the destination where media is sent to may not be the expected one, any of the parameters of the session may be incorrect, or the media security may be compromised. It is up to the endpoint to make a sensible decision taking into account the security risks of the application and the user preferences - the endpoint may decide to ask the user whether or not to accept the session.

On receiving a session description over an unauthenticated transport mechanism or from an untrusted party, software parsing the session description should take a few precautions. Similar concerns apply if integrity protection is not in place. Session descriptions contain information required to start software on the receiver's system. Software that parses a session description MUST NOT be able to start other software except that which is specifically configured as appropriate software to participate in multimedia sessions. It is normally considered inappropriate for software parsing a session description to start, on a user's system, software that is appropriate to participate in multimedia sessions, without the user first being informed that such software will be started and giving the user's consent. Thus, a session description arriving by session announcement, email, session invitation, or WWW page MUST NOT deliver the user into an interactive multimedia session unless the user has explicitly pre-authorized such action. As it is not always simple to tell whether or not a session is interactive, applications that are

unsure should assume sessions are interactive. Software processing URLs contained in session descriptions should also heed the security considerations identified in [RFC3986].

In this specification, there are no attributes that would allow the recipient of a session description to be informed to start multimedia tools in a mode where they default to transmitting. Under some circumstances it might be appropriate to define such attributes. If this is done, an application parsing a session description containing such attributes SHOULD either ignore them or inform the user that joining this session will result in the automatic transmission of multimedia data. The default behavior for an unknown attribute is to ignore it.

In certain environments, it has become common for intermediary systems to intercept and analyze session descriptions contained within other signaling protocols. This is done for a range of purposes, including but not limited to opening holes in firewalls to allow media streams to pass, or to mark, prioritize, or block traffic selectively. In some cases, such intermediary systems may modify the session description, for example, to have the contents of the session description match NAT bindings dynamically created. These behaviors are NOT RECOMMENDED unless the session description is conveyed in such a manner that allows the intermediary system to conduct proper checks to establish the authenticity of the session description, and the authority of its source to establish such communication sessions. SDP by itself does not include sufficient information to enable these checks: they depend on the encapsulating protocol (e.g., SIP or RTSP). Use of some procedures and SDP extensions (e.g., ICE [RFC8445] and ICE-SIP-SDP [I-D.ietf-mmusic-ice-sip-sdp]) may avoid the need for intermediaries to modify SDP.

SDP MUST NOT be used to convey keying material (e.g., using "a=crypto" [RFC4568]) unless it can be guaranteed that the channel over which the SDP is delivered is both private and authenticated.

8. IANA Considerations

8.1. The "application/sdp" Media Type

One media type registration from [RFC4566] is to be updated, as defined below.

To: ietf-types@iana.org
Subject: Registration of media type "application/sdp"

Type name: application

Subtype name: sdp

Required parameters: None.

Optional parameters: None.

Encoding considerations: 8-bit text.
SDP files are primarily UTF-8 format text. The "a=charset:" attribute may be used to signal the presence of other character sets in certain parts of an SDP file (see Section 6 of RFC XXXX). Arbitrary binary content cannot be directly represented in SDP.

Security considerations:
See Section 7 of RFC XXXX.

Interoperability considerations:
See RFC XXXX.

Published specification:
See RFC XXXX.

Applications which use this media type:
Voice over IP, video teleconferencing, streaming media, instant messaging, among others. See also Section 3 of RFC XXXX.

Fragment identifier considerations: None

Additional information:

Deprecated alias names for this type: N/A
Magic number(s): None.
File extension(s): The extension ".sdp" is commonly used.
Macintosh File Type Code(s): "sdp "

Person & email address to contact for further information:
IETF MMUSIC working group <mmusic@ietf.org>

Intended usage: COMMON

Restrictions on usage: None

Author/Change controller:
Authors of RFC XXXX
IETF MMUSIC working group delegated from the IESG

8.2. Registration of SDP Parameters with IANA

This document specifies IANA parameter registries for six named SDP sub-fields. Using the terminology in the SDP specification Augmented Backus-Naur Form (ABNF), they are "media", "proto", "att-field", "bwtype", "nettype", and "addrtype".

This document also replaces and updates the definitions of all those parameters previously defined by [RFC4566].

IANA: Please change all references to RFC4566 in these registries to instead refer to this document.

The contact name and email address for all parameters registered in this document is:

> The IETF MMUSIC working group <mmusic@ietf.org> or its successor as designated by the IESG.

All of these registries have a common format:

Type	SDP Name	[other fields]	Reference

8.2.1. Registration Procedure

A specification document that defines values for SDP "media", "proto", "att-field", "bwtype", "nettype", and "addrtype" parameters MUST include the following information:

- contact name;
- contact email address;
- name being defined (as it will appear in SDP);
- type of name ("media", "proto", "bwtype", "nettype", or "addrtype");
- a description of the purpose of the defined name;
- a stable reference to the document containing this information and the definition of the value. (This will typically be an RFC number.)

The subsections below specify what other information (if any) must be specified for particular parameters, and what other fields are to be included in the registry.

8.2.2. Media Types ("media")

The set of media types is intended to be small and SHOULD NOT be extended except under rare circumstances. The same rules should apply for media names as for top-level media types, and where possible the same name should be registered for SDP as for MIME. For media other than existing top-level media types, a Standards Track RFC MUST be produced for a new top-level media type to be registered, and the registration MUST provide good justification why no existing media name is appropriate (the "Standards Action" policy of [RFC8126]).

This memo registers the media types "audio", "video", "text", "application", and "message".

Note: The media types "control" and "data" were listed as valid in an early version of this specification (RFC 2327); however, their semantics were never fully specified and they are not widely used. These media types have been removed in this specification, although they still remain valid media type capabilities for a SIP user agent as defined in [RFC3840]. If these media types are considered useful in the future, a Standards Track RFC MUST be produced to document their use. Until that is done, applications SHOULD NOT use these types and SHOULD NOT declare support for them in SIP capabilities declarations (even though they exist in the registry created by [RFC3840]). Also note that [RFC6466] defined the "image" media type.

8.2.3. Transport Protocols ("proto")

The "proto" sub-field describes the transport protocol used. The registration procedure for this registry is "RFC Required".

This document registers two values:

- "RTP/AVP" is a reference to [RFC3550] used under the RTP Profile for Audio and Video Conferences with Minimal Control [RFC3551] running over UDP/IP,

- "UDP" indicates direct use of the UDP protocol.

New transport protocols MAY be defined, and MUST be registered with IANA. Registrations MUST reference an RFC describing the protocol. Such an RFC MAY be Experimental or Informational, although it is preferable that it be Standards Track. The RFC defining a new

protocol MUST define the rules by which the "fmt" (see below) namespace is managed.

"proto" names starting with "RTP/" MUST only be used for defining transport protocols that are profiles of the RTP protocol. For example, a profile whose short name is "XYZ" would be denoted by a "proto" sub-field of "RTP/XYZ".

Each transport protocol, defined by the "proto" sub-field, has an associated "fmt" namespace that describes the media formats that may be conveyed by that protocol. Formats cover all the possible encodings that could be transported in a multimedia session.

RTP payload formats under the "RTP/AVP" and other "RTP/*" profiles MUST use the payload type number as their "fmt" value. If the payload type number is dynamically assigned by this session description, an additional "rtpmap" attribute MUST be included to specify the format name and parameters as defined by the media type registration for the payload format. It is RECOMMENDED that other RTP profiles that are registered (in combination with RTP) as SDP transport protocols specify the same rules for the "fmt" namespace.

For the "UDP" protocol, allowed "fmt" values are media subtypes from the IANA Media Types registry. The media type and subtype combination <media>/<fmt> specifies the format of the body of UDP packets. Use of an existing media subtype for the format is encouraged. If no suitable media subtype exists, it is RECOMMENDED that a new one be registered through the IETF process [RFC6838] by production of, or reference to, a standards-track RFC that defines the format.

For other protocols, formats MAY be registered according to the rules of the associated "proto" specification.

Registrations of new formats MUST specify which transport protocols they apply to.

8.2.4. Attribute Names ("att-field")

Attribute-field names ("att-field") MUST be registered with IANA and documented, to avoid any issues due to conflicting attribute definitions under the same name. (While unknown attributes in SDP are simply ignored, conflicting ones that fragment the protocol are a serious problem.)

The format of the attribute registry is:

Type	SDP Name	Usage Level	Mux Category	Reference

For example, the attribute "setup" which is defined for both session and media level, will be listed in the new registry as follows:

Type	SDP Name	Usage Level	Mux Category	Reference
attribute	setup	session,media, dcsa,dcsa(msrp)	IDENTICAL	[RFC4145] [RFC6135] [I-D.mmusic-msrp-usage-data-channel]

This one registry combines all of the previous usage-level-specific "att-field" registries, including updates made by [I-D.ietf-mmusic-sdp-mux-attributes]. IANA is requested to do the necessary reformatting.

Section 6 of this document replaces the initial set of attribute definitions made by [RFC4566]. IANA is requested to update the registry accordingly.

Documents can define new attributes and can also extend the definitions of previously defined attributes:

8.2.4.1. New Attributes

New attribute registrations are accepted according to the "Specification Required" policy of [RFC8126], provided that the specification includes the following information:

- Contact Name.
- Contact Email Address.
- Attribute Name: The name of the attribute that will appear in SDP). This MUST conform to the definition of <att-field>.
- Attribute Syntax: For a value attribute (see clause 5.13), an ABNF definition of the attribute value <att-value> syntax (see Section 9) MUST be provided. The syntax MUST follow the rule form as per Section 2.2 of [RFC5234] and [RFC7405]. This SHALL define

the allowable values that the attribute might take. It MAY also define an extension method for the addition of future values. For a property attribute, the ABNF definition is omitted as the property attribute takes no values.

- Attribute Semantics: For a value attribute, a semantic description of the values that the attribute might take MUST be provided. The usage of a property attribute is described under purpose below.

- Attribute Value: The name of an ABNF syntax rule defining the syntax of the value. Absence of a rule name indicates that the attribute takes no values. Enclosing the rule name in "[" and "]" indicates that a value is optional.

- Usage Level: Usage level(s) of the attribute. This MUST be one or more of the following: session, media, source, dcsa and dcsa(subprotocol). For a definition of source level attributes, see [RFC5576]. For a definition of dcsa attributes see: [I-D.ietf-mmusic-data-channel-sdpneg].

- Charset Dependent: This MUST be "Yes" or "No" depending on whether the attribute value is subject to the charset attribute.

- Purpose: An explanation of the purpose and usage of the attribute.

- O/A Procedures: Offer/Answer procedures as explained in [RFC3264].

- Mux Category: This MUST indicate one of the following categories: NORMAL, NOT RECOMMENDED, IDENTICAL, SUM, TRANSPORT, INHERIT, IDENTICAL-PER-PT, SPECIAL or TBD as defined by [I-D.ietf-mmusic-sdp-mux-attributes].

- Reference: A reference to the specification defining the attribute.

The above is the minimum that IANA will accept. Attributes that are expected to see widespread use and interoperability SHOULD be documented with a standards-track RFC that specifies the attribute more precisely.

Submitters of registrations should ensure that the specification is in the spirit of SDP attributes, most notably that the attribute is platform independent in the sense that it makes no implicit assumptions about operating systems and does not name specific pieces of software in a manner that might inhibit interoperability.

Submitters of registrations should also carefully choose the attribute usage level. They should not choose only "session" when

the attribute can have different values when media is disaggregated, i.e., when each m= section has its own IP address on a different endpoint. In that case the attribute type chosen should be "session, media" or "media" (depending on desired semantics). The default rule is that for all new SDP attributes that can occur both in session and media level, the media level overrides the session level. When this is not the case for a new SDP attribute, it MUST be explicitly stated.

IANA has registered the initial set of attribute names ("att-field" values) with definitions as in Section 6 of this memo (these definitions replace those in [RFC4566]).

8.2.4.2. Updates to Existing Attributes

Updated attribute registrations are accepted according to the "Specification Required" policy of [RFC8126].

The Designated Expert reviewing the update is requested to evaluate whether the update is compatible with the prior intent and use of the attribute, and whether the new document is of sufficient maturity and authority in relation to the prior document.

The specification updating the attribute (for example, by adding a new value) MUST update registration information items from Section 8.2.4.1 according to the following constraints:

- Contact Name: A name for an entity responsible for the update MUST be provided.
- Contact Email Address: An email address for an entity responsible for the update MUST be provided.
- Attribute Name: MUST be provided and MUST NOT be changed. Otherwise it is a new attribute.
- Attribute Syntax: The existing rule syntax with the syntax extensions MUST be provided if there is a change to the syntax. A revision to an existing attribute usage MAY extend the syntax of an attribute, but MUST be backward compatible.
- Attribute Semantics: A semantic description of new additional attribute values or a semantic extension of existing values. Existing attribute values semantics MUST only be extended in a backward compatible manner.
- Usage Level: Updates MAY only add additional levels.

- Charset Dependent: MUST NOT be changed.

- Purpose: MAY be extended according to the updated usage.

- O/A Procedures: MAY be updated in a backward compatible manner and/or it applies to a new usage level only.

- Mux Category: No change unless from "TBD" to another value (see [I-D.ietf-mmusic-sdp-mux-attributes]. It MAY also change if 'media' level is being added to the definition of an attribute that previously did not include it.

- Reference: A new (additional or replacement) reference MUST be provided.

Items SHOULD be omitted if there is no impact to them as a result of the attribute update.

8.2.5. Bandwidth Specifiers ("bwtype")

A proliferation of bandwidth specifiers is strongly discouraged.

New bandwidth specifiers (<bwtype> sub-field values) MUST be registered with IANA. The submission MUST reference a standards-track RFC specifying the semantics of the bandwidth specifier precisely, and indicating when it should be used, and why the existing registered bandwidth specifiers do not suffice.

The RFC MUST specify the Mux Category for this value as defined by [I-D.ietf-mmusic-sdp-mux-attributes].

The format of the "bwtype" registry is:

Type	SDP Name	Mux Category	Reference

IANA is requested to update the "bwtype" registry entries for the bandwidth specifiers "CT" and "AS" with the definitions in Section 5.8 of this memo (these definitions replace those in [RFC4566]).

8.2.6. Network Types ("nettype")

Network type "IN", representing the Internet, is defined in Section 5.2 and Section 5.7 of this memo. (This definition replaces that in [RFC4566].)

To enable SDP to reference a new non-Internet environment a new network type (<nettype> sub-field value) MUST be registered with IANA. The registration is subject to the "RFC Required" policy of [RFC8126]. Although non-Internet environments are not normally the preserve of IANA, there may be circumstances when an Internet application needs to interoperate with a non-Internet application, such as when gatewaying an Internet telephone call into the Public Switched Telephone Network (PSTN). The number of network types should be small and should be rarely extended. A new network type registration MUST reference an RFC that gives details of the network type and the address type(s) that may be used with it.

The format of the "nettype" registry is:

Type	SDP Name	Usable addrtype Values	Reference

IANA is requested to update the "nettype" registry to this new format. The following is the updated content of th registry:

Type	SDP Name	Usable addrtype Values	Reference
nettype	IN	IP4, IP6	[RFCXXXX]
nettype	TN	RFC2543	[RFC2848]
nettype	ATM	NSAP, GWID, E164	[RFC3108]
nettype	PSTN	E164	[RFC7195]

Note that both [RFC7195] and [RFC3108] registered "E164" as an address type, although [RFC7195] mentions that the "E164" address type has a different context for ATM and PSTN networks.

8.2.7. Address Types ("addrtype")

New address types ("addrtype") MUST be registered with IANA. The registration is subject to the "RFC Required" policy of [RFC8126]. A new address type registration MUST reference an RFC giving details of the syntax of the address type. Address types are not expected to be registered frequently.

Section 5.7 of this document gives new definitions of address types "IP4" and "IP6".

8.3. Encryption Key Access Methods (OBSOLETE)

The IANA previously maintained a table of SDP encryption key access method ("enckey") names. This table is obsolete, since the "k=" line is not extensible. New registrations MUST NOT be accepted.

9. SDP Grammar

This section provides an Augmented BNF grammar for SDP. ABNF is defined in [RFC5234] and [RFC7405].

```
; SDP Syntax
session-description = version-field
                      origin-field
                      session-name-field
                      [information-field]
                      [uri-field]
                      *email-field
                      *phone-field
                      [connection-field]
                      *bandwidth-field
                      1*time-description
                      [key-field]
                      *attribute-field
                      *media-description

version-field =       %s"v" "=" 1*DIGIT CRLF
                          ;this memo describes version 0

origin-field =       %s"o" "=" username SP sess-id SP sess-version SP
                         nettype SP addrtype SP unicast-address CRLF

session-name-field =  %s"s" "=" text CRLF

information-field =   %s"i" "=" text CRLF

uri-field =           %s"u" "=" uri CRLF

email-field =         %s"e" "=" email-address CRLF

phone-field =         %s"p" "=" phone-number CRLF

connection-field =    %s"c" "=" nettype SP addrtype SP
                          connection-address CRLF
                          ;a connection field must be present
                          ;in every media description or at the
                          ;session level
```

```
   bandwidth-field =     %s"b" "=" bwtype ":" bandwidth CRLF

   time-description =    time-field
                             [repeat-description]

   repeat-description =  1*repeat-field
                             [zone-field]

   time-field =          %s"t" "=" start-time SP stop-time CRLF

   repeat-field =        %s"r" "=" repeat-interval SP typed-time
                             1*(SP typed-time) CRLF

   zone-field =          %s"z" "=" time SP ["-"] typed-time
                             *(SP time SP ["-"] typed-time) CRLF

   key-field =           %s"k" "=" key-type CRLF

   attribute-field =     %s"a" "=" attribute CRLF

   media-description =   media-field
                         [information-field]
                         *connection-field
                         *bandwidth-field
                         [key-field]
                         *attribute-field

   media-field =         %s"m" "=" media SP port ["/" integer]
                             SP proto 1*(SP fmt) CRLF

   ; sub-rules of 'o='
   username =            non-ws-string
                         ;pretty wide definition, but doesn't
                         ;include space

   sess-id =             1*DIGIT
                         ;should be unique for this username/host

   sess-version =        1*DIGIT

   nettype =             token
                         ;typically "IN"

   addrtype =            token
                         ;typically "IP4" or "IP6"

   ; sub-rules of 'u='
   uri =                 URI-reference
```

```
                        ; see RFC 3986

   ; sub-rules of 'e=', see RFC 5322 for definitions
   email-address        = address-and-comment / dispname-and-address
                          / addr-spec
   address-and-comment  = addr-spec 1*SP "(" 1*email-safe ")"
   dispname-and-address = 1*email-safe 1*SP "<" addr-spec ">"

   ; sub-rules of 'p='
   phone-number =       phone *SP "(" 1*email-safe ")" /
                        1*email-safe "<" phone ">" /
                        phone

   phone =              ["+"] DIGIT 1*(SP / "-" / DIGIT)

   ; sub-rules of 'c='
   connection-address =  multicast-address / unicast-address

   ; sub-rules of 'b='
   bwtype =             token

   bandwidth =          1*DIGIT

   ; sub-rules of 't='
   start-time =         time / "0"

   stop-time =          time / "0"

   time =               POS-DIGIT 9*DIGIT
                        ; Decimal representation of time in
                        ; seconds since January 1, 1900 UTC.
                        ; The representation is an unbounded
                        ; length field containing at least
                        ; 10 digits. Unlike some representations
                        ; used elsewhere, time in SDP does not
                        ; wrap in the year 2036.

   ; sub-rules of 'r=' and 'z='
   repeat-interval =    POS-DIGIT *DIGIT [fixed-len-time-unit]

   typed-time =         1*DIGIT [fixed-len-time-unit]

   fixed-len-time-unit = %s"d" / %s"h" / %s"m" / %s"s"
   ; NOTE: These units are case-sensitive.

   ; sub-rules of 'k='
   key-type =           %s"prompt" /
                        %s"clear:" text /
```

```
                        %s"base64:" base64 /
                        %s"uri:" uri
                        ; NOTE: These names are case-sensitive.

   base64      =        *base64-unit [base64-pad]
   base64-unit =        4base64-char
   base64-pad  =        2base64-char "==" / 3base64-char "="
   base64-char =        ALPHA / DIGIT / "+" / "/"

   ; sub-rules of 'a='
   attribute =          (att-field ":" att-value) / att-field

   att-field =          token

   att-value =          byte-string

   ; sub-rules of 'm='
   media =              token
                        ;typically "audio", "video", "text", "image"
                        ;or "application"

   fmt =                token
                        ;typically an RTP payload type for audio
                        ;and video media

   proto  =             token *("/" token)
                        ;typically "RTP/AVP" or "udp"

   port =               1*DIGIT

   ; generic sub-rules: addressing
   unicast-address =    IP4-address / IP6-address / FQDN / extn-addr

   multicast-address =  IP4-multicast / IP6-multicast / FQDN
                        / extn-addr

   IP4-multicast =      m1 3( "." decimal-uchar )
                        "/" ttl [ "/" numaddr ]
                        ; IP4 multicast addresses may be in the
                        ; range 224.0.0.0 to 239.255.255.255

   m1 =                 ("22" ("4"/"5"/"6"/"7"/"8"/"9")) /
                        ("23" DIGIT )

   IP6-multicast =      IP6-address [ "/" numaddr ]
                        ; IP6 address starting with FF

   numaddr =            integer
```

```
   ttl =                (POS-DIGIT *2DIGIT) / "0"

   FQDN =               4*(alpha-numeric / "-" / ".")
                        ; fully qualified domain name as specified
                        ; in RFC 1035 (and updates)

   IP4-address =        b1 3("." decimal-uchar)

   b1 =                 decimal-uchar
                        ; less than "224"

   IP6-address =                                    6( h16 ":" ) ls32
                        /                      "::" 5( h16 ":" ) ls32
                        / [               h16 ] "::" 4( h16 ":" ) ls32
                        / [ *1( h16 ":" ) h16 ] "::" 3( h16 ":" ) ls32
                        / [ *2( h16 ":" ) h16 ] "::" 2( h16 ":" ) ls32
                        / [ *3( h16 ":" ) h16 ] "::"    h16 ":"   ls32
                        / [ *4( h16 ":" ) h16 ] "::"              ls32
                        / [ *5( h16 ":" ) h16 ] "::"              h16
                        / [ *6( h16 ":" ) h16 ] "::"

   h16 =                1*4HEXDIG

   ls32 =               ( h16 ":" h16 ) / IP4-address

   ; Generic for other address families
   extn-addr =      non-ws-string

   ; generic sub-rules: datatypes
   text =               byte-string
                        ;default is to interpret this as UTF8 text.
                        ;ISO 8859-1 requires "a=charset:ISO-8859-1"
                        ;session-level attribute to be used

   byte-string =        1*(%x01-09/%x0B-0C/%x0E-FF)
                        ;any byte except NUL, CR, or LF

   non-ws-string =      1*(VCHAR/%x80-FF)
                        ;string of visible characters

   token-char =         ALPHA / DIGIT
                               / "!" / "#" / "$" / "%" / "&"
                               / "'" ; (single quote)
                               / "*" / "+" / "-" / "." / "^" / "_"
                               / "`" ; (Grave accent)
                               / "{" / "|" / "}" / "~"

   token =              1*(token-char)
```

```
email-safe =          %x01-09/%x0B-0C/%x0E-27/%x2A-3B/%x3D/%x3F-FF
                      ;any byte except NUL, CR, LF, or the quoting
                      ;characters ()<>

integer =             POS-DIGIT *DIGIT

zero-based-integer = "0" / integer

non-zero-int-or-real = integer / non-zero-real

non-zero-real = zero-based-integer "." *DIGIT POS-DIGIT

; generic sub-rules: primitives
alpha-numeric =       ALPHA / DIGIT

POS-DIGIT =           %x31-39 ; 1 - 9

decimal-uchar =       DIGIT
                      / POS-DIGIT DIGIT
                      / ("1" 2(DIGIT))
                      / ("2" ("0"/"1"/"2"/"3"/"4") DIGIT)
                      / ("2" "5" ("0"/"1"/"2"/"3"/"4"/"5"))

; external references:
ALPHA =               <ALPHA definition from RFC5234>
DIGIT =               <DIGIT definition from RFC5234>
CRLF =                <CRLF definition from RFC5234>
HEXDIG =              <HEXDIG definition from RFC5234>
SP =                  <SP definition from RFC5234>
VCHAR =               <VCHAR definition from RFC5234>
URI-reference =       <URI-reference definition from RFC3986>
addr-spec =           <addr-spec definition from RFC5322>
```

10. Summary of Changes from RFC 4566

- Generally clarified and refined terminology.
- Identified now-obsolete items: "a=cat", "a=keywds", "k=".
- Updated normative and informative references, and added references to additional relevant related RFCs.
- Reformatted the SDP Attributes section for readability. The syntax of attribute values is now given in ABNF.
- Made mandatory the sending of RTCP with inactive media streams.

- Removed the section "Private Sessions". That section dates back to a time when the primary use of SDP was with SAP (Session Announcement Protocol). That has fallen out of use. Now the vast majority of uses of SDP is for establishment of private sessions. The considerations for that are covered in Section 7.

- Expanded and clarified the specification of the "lang" and "sdplang" attributes.

- Removed some references to SAP because it is no longer in widespread use.

- Changed the way <fmt> values for UDP transport are registered.

- Changed the mechanism and documentation required for registering new attributes.

- Tightened up IANA registration procedures for extensions. Removed phone number and long-form name.

- Expanded the IANA nettype registry to identify valid addrtypes.

- Reorganized the several IANA att-type registries into a single registry

- Revised ABNF syntax for clarity. Backward compatibility is maintained with a few exceptions:

  * Revised the syntax of time descriptions ("t=", "r=", "z=") to remove ambiguities. Clarified that "z=" only modifies the immediately preceding "r=" lines. Made "z=" without a preceding "r=" a syntax error. (This is incompatible with certain aberrant usage.)

  * Updated the "IP6-address" and "IP6-multicast" rules, consistent with the syntax in RFC3986. (This mirrors a bug fix made to RFC3261 by RFC5964.) Removed rules that were unused as a result of this change.

- Revised normative statements that were redundant with ABNF syntax, making the text non-normative.

- Revised IPv4 unicast and multicast addresses in the example SDP descriptions per RFCs 5735 and 5771.

- Changed some examples to use IPv6 addresses, and added additional examples using IPv6.

- Incorporated case-insensitivity rules from RFC 4855.
- Revised sections that incorrectly referenced NTP.
- Clarified the explanation of the impact and use of a=charset.
- Revised the description of a=type to remove implication that it sometimes changes the default media direction to something other than sendrecv.

11. Acknowledgements

Many people in the IETF Multiparty Multimedia Session Control (MMUSIC) working group have made comments and suggestions contributing to this document.

In particular, we would like to thank the following people who contributed to the creation of this document or one of its predecessor documents: Adam Roach, Allison Mankin, Bernie Hoeneisen, Bill Fenner, Carsten Bormann, Eve Schooler, Flemming Andreasen, Gonzalo Camarillo, Joerg Ott, John Elwell, Jon Peterson, Jonathan Lennox, Jonathan Rosenberg, Keith Drage, Peter Parnes, Rob Lanphier, Ross Finlayson, Sean Olson, Spencer Dawkins, Steve Casner, Steve Hanna, Van Jacobson.

12. References

12.1. Normative References

[E164] International Telecommunication Union, "E.164 : The international public telecommunication numbering plan", ITU Recommendation E.164, November 2010.

[I-D.ietf-mmusic-data-channel-sdpneg] Drage, K., Makaraju, M., Ejzak, R., Marcon, J., and R. Even, "SDP-based Data Channel Negotiation", draft-ietf-mmusic-data-channel-sdpneg-28 (work in progress), May 2019.

[I-D.ietf-mmusic-sdp-mux-attributes] Nandakumar, S., "A Framework for SDP Attributes when Multiplexing", draft-ietf-mmusic-sdp-mux-attributes-17 (work in progress), February 2018.

[ISO.8859-1.1998]
International Organization for Standardization, "Information technology - 8-bit single byte coded graphic - character sets - Part 1: Latin alphabet No. 1, JTC1/SC2", ISO/IEC Standard 8859-1, 1998.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2848] Petrack, S. and L. Conroy, "The PINT Service Protocol: Extensions to SIP and SDP for IP Access to Telephone Call Services", RFC 2848, DOI 10.17487/RFC2848, June 2000, <https://www.rfc-editor.org/info/rfc2848>.

[RFC2978] Freed, N. and J. Postel, "IANA Charset Registration Procedures", BCP 19, RFC 2978, DOI 10.17487/RFC2978, October 2000, <https://www.rfc-editor.org/info/rfc2978>.

[RFC3108] Kumar, R. and M. Mostafa, "Conventions for the use of the Session Description Protocol (SDP) for ATM Bearer Connections", RFC 3108, DOI 10.17487/RFC3108, May 2001, <https://www.rfc-editor.org/info/rfc3108>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/info/rfc3629>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4145] Yon, D. and G. Camarillo, "TCP-Based Media Transport in the Session Description Protocol (SDP)", RFC 4145, DOI 10.17487/RFC4145, September 2005, <https://www.rfc-editor.org/info/rfc4145>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <https://www.rfc-editor.org/info/rfc4566>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, DOI 10.17487/RFC5576, June 2009, <https://www.rfc-editor.org/info/rfc5576>.

[RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <https://www.rfc-editor.org/info/rfc5646>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <https://www.rfc-editor.org/info/rfc5890>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/info/rfc5952>.

[RFC6135] Holmberg, C. and S. Blau, "An Alternative Connection Model for the Message Session Relay Protocol (MSRP)", RFC 6135, DOI 10.17487/RFC6135, February 2011, <https://www.rfc-editor.org/info/rfc6135>.

[RFC7195] Garcia-Martin, M. and S. Veikkolainen, "Session Description Protocol (SDP) Extension for Setting Audio and Video Media Streams over Circuit-Switched Bearers in the Public Switched Telephone Network (PSTN)", RFC 7195, DOI 10.17487/RFC7195, May 2014, <https://www.rfc-editor.org/info/rfc7195>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

12.2. Informative References

[I-D.ietf-mmusic-ice-sip-sdp]
Petit-Huguenin, M., Nandakumar, S., Keranen, A., Shpount, R., and C. Holmberg, "Session Description Protocol (SDP) Offer/Answer procedures for Interactive Connectivity Establishment (ICE)", draft-ietf-mmusic-ice-sip-sdp-38 (work in progress), August 2019.

[I-D.ietf-mmusic-sdp-bundle-negotiation]
Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-negotiation-54 (work in progress), December 2018.

[ITU.H332.1998]
International Telecommunication Union, "H.323 extended for loosely coupled conferences", ITU Recommendation H.332, September 1998.

[RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, DOI 10.17487/RFC2045, November 1996, <https://www.rfc-editor.org/info/rfc2045>.

[RFC2327] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, DOI 10.17487/RFC2327, April 1998, <https://www.rfc-editor.org/info/rfc2327>.

[RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, DOI 10.17487/RFC2974, October 2000, <https://www.rfc-editor.org/info/rfc2974>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <https://www.rfc-editor.org/info/rfc3264>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <https://www.rfc-editor.org/info/rfc3550>.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <https://www.rfc-editor.org/info/rfc3551>.

[RFC3556] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, DOI 10.17487/RFC3556, July 2003, <https://www.rfc-editor.org/info/rfc3556>.

[RFC3605] Huitema, C., "Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)", RFC 3605, DOI 10.17487/RFC3605, October 2003, <https://www.rfc-editor.org/info/rfc3605>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <https://www.rfc-editor.org/info/rfc3711>.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, DOI 10.17487/RFC3840, August 2004, <https://www.rfc-editor.org/info/rfc3840>.

[RFC3890] Westerlund, M., "A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP)", RFC 3890, DOI 10.17487/RFC3890, September 2004, <https://www.rfc-editor.org/info/rfc3890>.

[RFC4568] Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, DOI 10.17487/RFC4568, July 2006, <https://www.rfc-editor.org/info/rfc4568>.

[RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, DOI 10.17487/RFC4855, February 2007, <https://www.rfc-editor.org/info/rfc4855>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description Protocol (SDP) Grouping Framework", RFC 5888, DOI 10.17487/RFC5888, June 2010, <https://www.rfc-editor.org/info/rfc5888>.

[RFC6466] Salgueiro, G., "IANA Registration of the 'image' Media Type for the Session Description Protocol (SDP)", RFC 6466, DOI 10.17487/RFC6466, December 2011, <https://www.rfc-editor.org/info/rfc6466>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <https://www.rfc-editor.org/info/rfc6838>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <https://www.rfc-editor.org/info/rfc7405>.

[RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", RFC 7656, DOI 10.17487/RFC7656, November 2015, <https://www.rfc-editor.org/info/rfc7656>.

[RFC7826] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, Ed., "Real-Time Streaming Protocol Version 2.0", RFC 7826, DOI 10.17487/RFC7826, December 2016, <https://www.rfc-editor.org/info/rfc7826>.

[RFC8445] Keranen, A., Holmberg, C., and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal", RFC 8445, DOI 10.17487/RFC8445, July 2018, <https://www.rfc-editor.org/info/rfc8445>.

Authors' Addresses

Ali Begen
Networked Media
Konya
Turkey

EMail: ali.begen@networked.media

Paul Kyzivat
USA

EMail: pkyzivat@alum.mit.edu

Colin Perkins
University of Glasgow
School of Computing Science
University of Glasgow
Glasgow G12 8QQ
UK

EMail: csp@csperkins.org

Mark Handley
University College London
Department of Computer Science
London WC1E 6BT
UK

EMail: M.Handley@cs.ucl.ac.uk