Network Working Group
Request for Comments: 3264
Category: Standards Track
Obsoletes: 2543

Rosenberg/Schulzrinne
dynamicsoft/Columbia U.
June 2002

An Offer/Answer Model with the Session Description Protocol (SDP)

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This document defines a mechanism by which two entities can make use of the Session Description Protocol (SDP) to arrive at a common view of a multimedia session between them. In the model, one participant offers the other a description of the desired session from their perspective, and the other participant answers with the desired session from their perspective. This offer/answer model is most useful in unicast sessions where information from both participants is needed for the complete view of the session. The offer/answer model is used by protocols like the Session Initiation Protocol (SIP).

Contents

1 Introduction

The Session Description Protocol (SDP) [1] was originally conceived as a way to describe multicast sessions carried on the Mbone. The Session Announcement Protocol (SAP) [6] was devised as a multicast mechanism to carry SDP messages. Although the SDP specification allows for unicast operation, it is not complete. Unlike multicast, where there is a global view of the session that is used by all participants, unicast sessions involve two participants, and a complete view of the session requires information from both participants, and agreement on parameters between them.

As an example, a multicast session requires conveying a single multicast address for a particular media stream. However, for a unicast session, two addresses are needed - one for each participant. As another example, a multicast session requires an indication of which codecs will be used in the session. However, for unicast, the set of codecs needs to be determined by finding an overlap in the set supported by each participant.

As a result, even though SDP has the expressiveness to describe unicast sessions, it is missing the semantics and operational details of how it is actually done. In this document, we remedy that by defining a simple offer/answer model based on SDP. In this model, one participant in the session generates an SDP message that constitutes the offer - the set of media streams and codecs the offerer wishes to use, along with the IP addresses and ports the offerer would like to use to receive the media. The offer is conveyed to the other participant, called the answerer. The answerer generates an answer, which is an SDP message that responds to the offer provided by the offerer. The answer has a matching media stream for each stream in the offer, indicating whether the stream is accepted or not, along with the codecs that will be used and the IP addresses and ports that the answerer wants to use to receive media.

It is also possible for a multicast session to work similar to a unicast one; its parameters are negotiated between a pair of users as in the unicast case, but both sides send packets to the same multicast address, rather than unicast ones. This document also discusses the application of the offer/answer model to multicast streams.

We also define guidelines for how the offer/answer model is used to update a session after an initial offer/answer exchange.

The means by which the offers and answers are conveyed are outside the scope of this document. The offer/answer model defined here is the mandatory baseline mechanism used by the Session Initiation Protocol (SIP) [7].

2 Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL","SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [2] and indicate requirement levels for compliant implementations.

3 Definitions

The following terms are used throughout this document:

Agent: An agent is the protocol implementation involved in the offer/answer exchange. There are two agents involved in an offer/answer exchange.

Answer: An SDP message sent by an answerer in response to an offer received from an offerer.

Answerer: An agent which receives a session description from another agent describing aspects of desired media communication, and then responds to that with its own session description.

Media Stream: From RTSP [8], a media stream is a single media instance, e.g., an audio stream or a video stream as well as a single whiteboard or shared application group. In SDP, a media stream is described by an "m=" line and its associated attributes.

Offer: An SDP message sent by an offerer.

Offerer: An agent which generates a session description in order to create or modify a session.

4 Protocol Operation

The offer/answer exchange assumes the existence of a higher layer protocol (such as SIP) which is capable of exchanging SDP for the purposes of session establishment between agents.

Protocol operation begins when one agent sends an initial offer to another agent. An offer is initial if it is outside of any context that may have already been established through the higher layer protocol. It is assumed that the higher layer protocol provides maintenance of some kind of context which allows the various SDP exchanges to be associated together.

The agent receiving the offer MAY generate an answer, or it MAY reject the offer. The means for rejecting an offer are dependent on the higher layer protocol. The offer/answer exchange is atomic; if the answer is rejected, the session reverts to the state prior to the offer (which may be absence of a session).

At any time, either agent MAY generate a new offer that updates the session. However, it MUST NOT generate a new offer if it has received an offer which it has not yet answered or rejected. Furthermore, it MUST NOT generate a new offer if it has generated a prior offer for which it has not yet received an answer or a rejection. If an agent receives an offer after having sent one, but before receiving an answer to it, this is considered a “glare” condition.

The term glare was originally used in circuit switched telecommunications networks to describe the condition where two switches both attempt to seize the same available circuit on the same trunk at the same time. Here, it means both agents have attempted to send an updated offer at the same time.

The higher layer protocol needs to provide a means for resolving such conditions. The higher layer protocol will need to provide a means for ordering of messages in each direction. SIP meets these requirements [7].

5 Generating the Initial Offer

The offer (and answer) MUST be a valid SDP message, as defined by RFC 2327 [1], with one exception. RFC 2327 mandates that either an e or a p line is present in the SDP message. This specification relaxes that constraint; an SDP formulated for an offer/answer application MAY omit both the e and p lines. The numeric value of the session id and version in the o line MUST be representable with a 64 bit signed integer. The initial value of the version MUST be less than (2**62)-1, to avoid rollovers. Although the SDP specification allows for multiple session descriptions to be concatenated together into a large SDP message, an SDP message used in the offer/answer model MUST contain exactly one session description.

The SDP “s=“ line conveys the subject of the session, which is reasonably defined for multicast, but ill defined for unicast. For unicast sessions, it is RECOMMENDED that it consist of a single space character (0x20) or a dash (-).

Unfortunately, SDP does not allow the “s=“ line to be empty.

The SDP "t=" line conveys the time of the session. Generally, streams for unicast sessions are created and destroyed through external signaling means, such as SIP. In that case, the "t=" line SHOULD have a value of "0 0".

The offer will contain zero or more media streams (each media stream is described by an "m=" line and its associated attributes). Zero media streams implies that the offerer wishes to communicate, but that the streams for the session will be added at a later time through a modified offer. The streams MAY be for a mix of unicast and multicast; the latter obviously implies a multicast address in the relevant "c=" line(s).

Construction of each offered stream depends on whether the stream is multicast or unicast.

5.1 Unicast Streams

If the offerer wishes to only send media on a stream to its peer, it MUST mark the stream as sendonly with the "a=sendonly" attribute. We refer to a stream as being marked with a certain direction if a direction attribute was present as either a media stream attribute or a session attribute. If the offerer wishes to only receive media from its peer, it MUST mark the stream as recvonly. If the offerer wishes to communicate, but wishes to neither send nor receive media at this time, it MUST mark the stream with an "a=inactive" attribute. The inactive direction attribute is specified in RFC 3108 [3]. Note that in the case of the Real Time Transport Protocol (RTP) [4], RTCP is still sent and received for sendonly, recvonly, and inactive streams. That is, the directionality of the media stream has no impact on the RTCP usage. If the offerer wishes to both send and receive media with its peer, it MAY include an "a=sendrecv" attribute, or it MAY omit it, since sendrecv is the default.

For recvonly and sendrecv streams, the port number and address in the offer indicate where the offerer would like to receive the media stream. For sendonly RTP streams, the address and port number indirectly indicate where the offerer wants to receive RTCP reports. Unless there is an explicit indication otherwise, reports are sent to the port number one higher than the number indicated. The IP address and port present in the offer indicate nothing about the source IP address and source port of RTP and RTCP packets that will be sent by the offerer. A port number of zero in the offer indicates that the stream is offered but MUST NOT be used. This has no useful semantics in an initial offer, but is allowed for reasons of completeness, since the answer can contain a zero port indicating a rejected stream (Section 5). Furthermore, existing streams can be terminated by setting the port to zero (Section 8). In general, a port number of zero indicates that the media stream is not wanted.

The list of media formats for each media stream conveys two pieces of information, namely the set of formats (codecs and any parameters associated with the codec, in the case of RTP) that the offerer is capable of sending and/or receiving (depending on the direction attributes), and, in the case of RTP, the RTP payload type numbers used to identify those formats. If multiple formats are listed, it means that the offerer is capable of making use of any of those formats during the session. In other words, the answerer MAY change formats in the middle of the session, making use of any of the formats listed, without sending a new offer. For a sendonly stream, the offer SHOULD indicate those formats the offerer is willing to send for this stream. For a recvonly stream, the offer SHOULD indicate those formats the offerer is willing to receive for this stream. For a sendrecv stream, the offer SHOULD indicate those codecs that the offerer is willing to send and receive with.

For recvonly RTP streams, the payload type numbers indicate the value of the payload type field in RTP packets the offerer is expecting to receive for that codec. For sendonly RTP streams, the payload type numbers indicate the value of the payload type field in RTP packets the offerer is planning to send for that codec. For sendrecv RTP streams, the payload type numbers indicate the value of the payload type field the offerer expects to receive, and would prefer to send. However, for sendonly and sendrecv streams, the answer might indicate different payload type numbers for the same codecs, in which case, the offerer MUST send with the payload type numbers from the answer.

Different payload type numbers may be needed in each direction because of interoperability concerns with H.323.

As per RFC 2327, fmtp parameters MAY be present to provide additional parameters of the media format.

In the case of RTP streams, all media descriptions SHOULD contain "a=rtpmap" mappings from RTP payload types to encodings. If there is no "a=rtpmap", the default payload type mapping, as defined by the current profile in use (for example, RFC 1890 [5]) is to be used.

This allows easier migration away from static payload types.

In all cases, the formats in the "m=" line MUST be listed in order of preference, with the first format listed being preferred. In this case, preferred means that the recipient of the offer SHOULD use the format with the highest preference that is acceptable to it.

If the ptime attribute is present for a stream, it indicates the desired packetization interval that the offerer would like to receive. The ptime attribute MUST be greater than zero.

If the bandwidth attribute is present for a stream, it indicates the desired bandwidth that the offerer would like to receive. A value of zero is allowed, but discouraged. It indicates that no media should be sent. In the case of RTP, it would also disable all RTCP.

If multiple media streams of different types are present, it means that the offerer wishes to use those streams at the same time. A typical case is an audio and a video stream as part of a videoconference.

If multiple media streams of the same type are present in an offer, it means that the offerer wishes to send (and/or receive) multiple streams of that type at the same time. When sending multiple streams of the same type, it is a matter of local policy as to how each media source of that type (for example, a video camera and VCR in the case of video) is mapped to each stream. When a user has a single source for a particular media type, only one policy makes sense: the source is sent to each stream of the same type. Each stream MAY use different encodings. When receiving multiple streams of the same type, it is a matter of local policy as to how each stream is mapped to the various media sinks for that particular type (for example, speakers or a recording device in the case of audio). There are a few constraints on the policies, however. First, when receiving multiple streams of the same type, each stream MUST be mapped to at least one sink for the purpose of presentation to the user. In other words, the intent of receiving multiple streams of the same type is that they should all be presented in parallel, rather than choosing just one. Another constraint is that when multiple streams are received and sent to the same sink, they MUST be combined in some media specific way. For example, in the case of two audio streams, the received media from each might be mapped to the speakers. In that case, the combining operation would be to mix them. In the case of multiple instant messaging streams, where the sink is the screen, the combining operation would be to present all of them to the user interface. The third constraint is that if multiple sources are mapped to the same stream, those sources MUST be combined in some media specific way before they are sent on the stream. Although policies beyond these constraints are flexible, an agent won't generally want a policy that will copy media from its sinks to its sources unless it is a conference server (i.e., don't copy received media on one stream to another stream).

A typical usage example for multiple media streams of the same type is a pre-paid calling card application, where the user can press and hold the pound ("#") key at any time during a call to hangup and make a new call on the same card. This requires media from the user to two destinations - the remote gateway, and the DTMF processing application which looks for the pound. This could be accomplished with two media streams, one sendrecv to the gateway, and the other sendonly (from the perspective of the user) to the DTMF application.

Once the offerer has sent the offer, it MUST be prepared to receive media for any recvonly streams described by that offer. It MUST be prepared to send and receive media for any sendrecv streams in the offer, and send media for any sendonly streams in the offer (of course, it cannot actually send until the peer provides an answer with the needed address and port information). In the case of RTP, even though it may receive media before the answer arrives, it will not be able to send RTCP receiver reports until the answer arrives.

5.2 Multicast Streams

If a session description contains a multicast media stream which is listed as receive (send) only, it means that the participants, including the offerer and answerer, can only receive (send) on that stream. This differs from the unicast view, where the directionality refers to the flow of media between offerer and answerer.

Beyond that clarification, the semantics of an offered multicast stream are exactly as described in RFC 2327 [1].

6 Generating the Answer

The answer to an offered session description is based on the offered session description. If the answer is different from the offer in any way (different IP addresses, ports, etc.), the origin line MUST be different in the answer, since the answer is generated by a different entity. In that case, the version number in the "o=" line of the answer is unrelated to the version number in the o line of the offer.

For each "m=" line in the offer, there MUST be a corresponding "m=" line in the answer. The answer MUST contain exactly the same number of "m=" lines as the offer. This allows for streams to be matched up based on their order. This implies that if the offer contained zero "m=" lines, the answer MUST contain zero "m=" lines.

The "t=" line in the answer MUST equal that of the offer. The time of the session cannot be negotiated.

An offered stream MAY be rejected in the answer, for any reason. If a stream is rejected, the offerer and answerer MUST NOT generate media (or RTCP packets) for that stream. To reject an offered stream, the port number in the corresponding stream in the answer MUST be set to zero. Any media formats listed are ignored. At least one MUST be present, as specified by SDP.

Constructing an answer for each offered stream differs for unicast and multicast.

6.1 Unicast Streams

If a stream is offered with a unicast address, the answer for that stream MUST contain a unicast address. The media type of the stream in the answer MUST match that of the offer.

If a stream is offered as sendonly, the corresponding stream MUST be marked as recvonly or inactive in the answer. If a media stream is listed as recvonly in the offer, the answer MUST be marked as sendonly or inactive in the answer. If an offered media stream is listed as sendrecv (or if there is no direction attribute at the media or session level, in which case the stream is sendrecv by default), the corresponding stream in the answer MAY be marked as sendonly, recvonly, sendrecv, or inactive. If an offered media stream is listed as inactive, it MUST be marked as inactive in the answer.

For streams marked as recvonly in the answer, the “m=“ line MUST contain at least one media format the answerer is willing to receive with from amongst those listed in the offer. The stream MAY indicate additional media formats, not listed in the corresponding stream in the offer, that the answerer is willing to receive. For streams marked as sendonly in the answer, the “m=“ line MUST contain at least one media format the answerer is willing to send from amongst those listed in the offer. For streams marked as sendrecv in the answer, the “m=“ line MUST contain at least one codec the answerer is willing to both send and receive, from amongst those listed in the offer. The stream MAY indicate additional media formats, not listed in the corresponding stream in the offer, that the answerer is willing to send or receive (of course, it will not be able to send them at this time, since it was not listed in the offer). For streams marked as inactive in the answer, the list of media formats is constructed based on the offer. If the offer was sendonly, the list is constructed as if the answer were recvonly. Similarly, if the offer was recvonly, the list is constructed as if the answer were sendonly, and if the offer was sendrecv, the list is constructed as if the answer were sendrecv. If the offer was inactive, the list is constructed as if the offer were actually sendrecv and the answer were sendrecv.

The connection address and port in the answer indicate the address where the answerer wishes to receive media (in the case of RTP, RTCP will be received on the port which is one higher unless there is an explicit indication otherwise). This address and port MUST be present even for sendonly streams; in the case of RTP, the port one higher is still used to receive RTCP.

In the case of RTP, if a particular codec was referenced with a specific payload type number in the offer, that same payload type number SHOULD be used for that codec in the answer. Even if the same payload type number is used, the answer MUST contain rtpmap attributes to define the payload type mappings for dynamic payload types, and SHOULD contain mappings for static payload types. The media formats in the “m=“ line MUST be listed in order of preference, with the first format listed being preferred. In this case, preferred means that the offerer SHOULD use the format with the highest preference from the answer.

Although the answerer MAY list the formats in their desired order of preference, it is RECOMMENDED that unless there is a specific reason, the answerer list formats in the same relative order they were present in the offer. In other words, if a stream in the offer lists audio codecs 8, 22 and 48, in that order, and the answerer only supports codecs 8 and 48, it is RECOMMENDED that, if the answerer has no reason to change it, the ordering of codecs in the answer be 8, 48, and not 48, 8. This helps assure that the same codec is used in both directions.

The interpretation of fmtp parameters in an offer depends on the parameters. In many cases, those parameters describe specific configurations of the media format, and should therefore be processed as the media format value itself would be. This means that the same fmtp parameters with the same values MUST be present in the answer if the media format they describe is present in the answer. Other fmtp parameters are more like parameters, for which it is perfectly acceptable for each agent to use different values. In that case, the answer MAY contain fmtp parameters, and those MAY have the same values as those in the offer, or they MAY be different. SDP extensions that define new parameters SHOULD specify the proper interpretation in offer/answer.

The answerer MAY include a non-zero ptime attribute for any media stream; this indicates the packetization interval that the answerer would like to receive. There is no requirement that the packetization interval be the same in each direction for a particular stream.

The answerer MAY include a bandwidth attribute for any media stream; this indicates the bandwidth that the answerer would like the offerer to use when sending media. The value of zero is allowed, interpreted as described in Section 5.

If the answerer has no media formats in common for a particular offered stream, the answerer MUST reject that media stream by setting the port to zero.

If there are no media formats in common for all streams, the entire offered session is rejected.

Once the answerer has sent the answer, it MUST be prepared to receive media for any recvonly streams described by that answer. It MUST be prepared to send and receive media for any sendrecv streams in the answer, and it MAY send media immediately. The answerer MUST be prepared to receive media for recvonly or sendrecv streams using any media formats listed for those streams in the answer, and it MAY send media immediately. When sending media, it SHOULD use a packetization interval equal to the value of the ptime attribute in the offer, if any was present. It SHOULD send media using a bandwidth no higher than the value of the bandwidth attribute in the offer, if any was present. The answerer MUST send using a media format in the offer that is also listed in the answer, and SHOULD send using the most preferred media format in the offer that is also listed in the answer. In the case of RTP, it MUST use the payload type numbers from the offer, even if they differ from those in the answer.

6.2 Multicast Streams

Unlike unicast, where there is a two-sided view of the stream, there is only a single view of the stream for multicast. As such, generating an answer to a multicast offer generally involves modifying a limited set of aspects of the stream.

If a multicast stream is accepted, the address and port information in the answer MUST match that of the offer. Similarly, the directionality information in the answer (sendonly, recvonly, or sendrecv) MUST equal that of the offer. This is because all participants in a multicast session need to have equivalent views of the parameters of the session, an underlying assumption of the multicast bias of RFC 2327.

The set of media formats in the answer MUST be equal to or be a subset of those in the offer. Removing a format is a way for the answerer to indicate that the format is not supported.

The ptime and bandwidth attributes in the answer MUST equal the ones in the offer, if present. If not present, a non-zero ptime MAY be added to the answer.

7 Offerer Processing of the Answer

When the offerer receives the answer, it MAY send media on the accepted stream(s) (assuming it is listed as sendrecv or recvonly in the answer). It MUST send using a media format listed in the answer, and it SHOULD use the first media format listed in the answer when it does send.

> The reason this is a SHOULD,and not a MUST(its also a SHOULD,and not a MUST,for the answerer), is because there will oftentimes be a need to change codecs on the fly. For example, during silence periods, an agent might like to switch to a comfort noise codec. Or, if the user presses a number on the keypad, the agent might like to send that using RFC 2833 [9]. Congestion control might necessitate changing to a lower rate codec based on feedback.

The offerer SHOULD send media according to the value of any ptime and bandwidth attribute in the answer.

The offerer MAY immediately cease listening for media formats that were listed in the initial offer, but not present in the answer.

8 Modifying the Session

At any point during the session, either participant MAY issue a new offer to modify characteristics of the session. It is fundamental to the operation of the offer/answer model that the exact same offer/answer procedure defined above is used for modifying parameters of an existing session.

The offer MAY be identical to the last SDP provided to the other party (which may have been provided in an offer or an answer), or it MAY be different. We refer to the last SDP provided as the “previous SDP”. If the offer is the same, the answer MAY be the same as the previous SDP from the answerer, or it MAY be different. If the offered SDP is different from the previous SDP, some constraints are placed on its construction, discussed below.

Nearly all aspects of the session can be modified. New streams can be added, existing streams can be deleted, and parameters of existing streams can change. When issuing an offer that modifies the session, the “o=“ line of the new SDP MUST be identical to that in the previous SDP, except that the version in the origin field MUST increment by one from the previous SDP. If the version in the origin line does not increment, the SDP MUST be identical to the SDP with that version number. The answerer MUST be prepared to receive an offer that contains SDP with a version that has not changed; this is effectively a no-op. However, the answerer MUST generate a valid answer (which MAY be the same as the previous SDP from the answerer, or MAY be different), according to the procedures defined in Section 6.

If an SDP is offered, which is different from the previous SDP, the new SDP MUST have a matching media stream for each media stream in the previous SDP. In other words, if the previous SDP had N “m=“ lines, the new SDP MUST have at least N “m=“ lines. The i-th media stream in the previous SDP, counting from the top, matches the i-th media stream in the new SDP, counting from the top. This matching is necessary in order for the answerer to determine which stream in the new SDP corresponds to a stream in the previous SDP. Because of these requirements, the number of “m=“ lines in a stream never decreases, but either stays the same or increases. Deleted media streams from a previous SDP MUST NOT be removed in a new SDP; however, attributes for these streams need not be present.

8.1 Adding a Media Stream

New media streams are created by new additional media descriptions below the existing ones, or by reusing the “slot” used by an old media stream which had been disabled by setting its port to zero. Reusing its slot means that the new media description replaces the old one, but retains its positioning relative to other media descriptions in the SDP. New media descriptions MUST appear below any existing media sections. The rules for formatting these media descriptions are identical to those described in Section 5.

When the answerer receives an SDP with more media descriptions than the previous SDP from the offerer, or it receives an SDP with a media stream in a slot where the port was previously zero, the answerer knows that new media streams are being added. These can be rejected or accepted by placing an appropriately structured media description in the answer. The procedures for constructing the new media description in the answer are described in Section 6.

8.2 Removing a Media Stream

Existing media streams are removed by creating a new SDP with the port number for that stream set to zero. The stream description MAY omit all attributes present previously, and MAY list just a single media format.

A stream that is offered with a port of zero MUST be marked with port zero in the answer. Like the offer, the answer MAY omit all attributes present previously, and MAY list just a single media format from amongst those in the offer.

Removal of a media stream implies that media is no longer sent for that stream, and any media that is received is discarded. In the case of RTP, RTCP transmission also ceases, as does processing of any received RTCP packets. Any resources associated with it can be released. The user interface might indicate that the stream has terminated, by closing the associated window on a PC, for example.

8.3 Modifying a Media Stream

Nearly all characteristics of a media stream can be modified.

8.3.1 Modifying Address, Port or Transport

The port number for a stream MAY be changed. To do this, the offerer creates a new media description, with the port number in the m line different from the corresponding stream in the previous SDP. If only the port number is to be changed, the rest of the media stream description SHOULD remain unchanged. The offerer MUST be prepared to receive media on both the old and new ports as soon as the offer is sent. The offerer SHOULD NOT cease listening for media on the old port until the answer is received and media arrives on the new port. Doing so could result in loss of media during the transition. Received, in this case, means that the media is passed to a media sink. This means that if there is a playout buffer, the agent would continue to listen on the old port until the media on the new port reached the top of the playout buffer. At that time, it MAY cease listening for media on the old port.

The corresponding media stream in the answer MAY be the same as the stream in the previous SDP from the answerer, or it MAY be different. If the updated stream is accepted by the answerer, the answerer SHOULD begin sending traffic for that stream to the new port immediately. If the answerer changes the port from the previous SDP, it MUST be prepared to receive media on both the old and new ports as soon as the answer is sent. The answerer MUST NOT cease listening for media on the old port until media arrives on the new port. At that time, it MAY cease listening for media on the old port. The same is true for an offerer that sends an updated offer with a new port; it MUST NOT cease listening for media on the old port until media arrives on the new port.

Of course, if the offered stream is rejected, the offerer can cease being prepared to receive using the new port as soon as the rejection is received.

To change the IP address where media is sent to, the same procedure is followed for changing the port number. The only difference is that the connection line is updated, not the port number.

The transport for a stream MAY be changed. The process for doing this is identical to changing the port, except the transport is updated, not the port.

8.3.2 Changing the Set of Media Formats

The list of media formats used in the session MAY be changed. To do this, the offerer creates a new media description, with the list of media formats in the "m=" line different from the corresponding media stream in the previous SDP. This list MAY include new formats, and MAY remove formats present from the previous SDP. However, in the case of RTP, the mapping from a particular dynamic payload type number to a particular codec within that media stream MUST NOT change for the duration of a session. For example, if A generates an offer with G.711 assigned to dynamic payload type number 46, payload type number 46 MUST refer to G.711 from that point forward in any offers or answers for that media stream within the session. However, it is acceptable for multiple payload type numbers to be mapped to the same codec, so that an updated offer could also use payload type number 72 for G.711.

> The mappings need to remain fixed for the duration of the session because of the loose synchronization between signaling exchanges of SDP and the media stream.

The corresponding media stream in the answer is formulated as described in Section 6, and may result in a change in media formats as well. Similarly, as described in Section 6, as soon as it sends its answer, the answerer MUST begin sending media using any formats in the offer that were also present in the answer, and SHOULD use the most preferred format in the offer that was also listed in the answer (assuming the stream allows for sending), and MUST NOT send using any formats that are not in the offer, even if they were present in a previous SDP from the peer. Similarly, when the offerer receives the answer, it MUST begin sending media using any formats in the answer, and SHOULD use the most preferred one (assuming the stream allows for sending), and MUST NOT send using any formats that are not in the answer, even if they were present in a previous SDP from the peer.

When an agent ceases using a media format (by not listing that format in an offer or answer, even though it was in a previous SDP) the agent will still need to be prepared to receive media with that format for a brief time. How does it know when it can be prepared to stop receiving with that format? If it needs to know, there are three techniques that can be applied. First, the agent can change ports in addition to changing formats. When media arrives on the new port, it knows that the peer has ceased sending with the old format, and it can cease being prepared to receive with it. This approach has the benefit of being media format independent. However, changes in ports may require changes in resource reservation or rekeying of security protocols. The second approach is to use a totally new set of dynamic payload types for all codecs when one is discarded. When media is received with one of the new payload types, the agent knows that the peer has ceased sending with the old format. This approach doesn't affect reservations or security contexts, but it is RTP specific and wasteful of a very small payload type space. A third approach is to use a timer. When the SDP from the peer is received, the timer is set. When it fires, the agent can cease being prepared to receive with the old format. A value of one minute would typically be more than sufficient. In some cases, an agent may not care, and thus continually be prepared to receive with the old formats. Nothing need be done in this case.

Of course, if the offered stream is rejected, the offer can cease being prepared to receive using any new formats as soon as the rejection is received.

8.3.3 Changing Media Types

The media type (audio, video, etc.) for a stream MAY be changed. It is RECOMMENDED that the media type be changed (as opposed to adding a new stream), when the same logical data is being conveyed, but just in a different media format. This is particularly useful for changing between voiceband fax and fax in a single stream, which are both separate media types. To do this, the offerer creates a new media description, with a new media type, in place of the description in the previous SDP which is to be changed.

The corresponding media stream in the answer is formulated as described in Section 6. Assuming the stream is acceptable, the answerer SHOULD begin sending with the new media type and formats as soon as it receives the offer. The offerer MUST be prepared to receive media with both the old and new types until the answer is received, and media with the new type is received and reaches the top of the playout buffer.

8.3.4 Changing Attributes

Any other attributes in a media description MAY be updated in an offer or answer. Generally, an agent MUST send media (if the directionality of the stream allows) using the new parameters once the SDP with the change is received.

8.4 Putting a Unicast Media Stream on Hold

If a party in a call wants to put the other party "on hold", i.e., request that it temporarily stops sending one or more unicast media streams, a party offers the other an updated SDP.

If the stream to be placed on hold was previously a sendrecv media stream, it is placed on hold by marking it as sendonly. If the stream to be placed on hold was previously a recvonly media stream, it is placed on hold by marking it inactive.

This means that a stream is placed "on hold" separately in each direction. Each stream is placed "on hold" independently. The recipient of an offer for a stream on-hold SHOULD NOT automatically return an answer with the corresponding stream on hold. An SDP with all streams "on hold" is referred to as held SDP.

> Certain third party call control scenarios do not work when an answerer responds to held SDP with held SDP.

Typically, when a user "presses" hold, the agent will generate an offer with all streams in the SDP indicating a direction of sendonly, and it will also locally mute, so that no media is sent to the far end, and no media is played out.

RFC 2543 [10] specified that placing a user on hold was accomplished by setting the connection address to 0.0.0.0. Its usage for putting a call on hold is no longer recommended, since it doesn't allow for RTCP to be used with held streams, doesn't work with IPv6, and breaks with connection oriented media. However, it can be useful in an initial offer when the offerer knows it wants to use a particular set of media streams and formats, but doesn't know the addresses and ports at the time of the offer. Of course, when used, the port number MUST NOT be zero, which would specify that the stream has been disabled. An agent MUST be capable of receiving SDP with a connection address of 0.0.0.0, in which case it means that neither RTP nor RTCP should be sent to the peer.

9 Indicating Capabilities

Before an agent sends an offer, it is helpful to know if the media formats in that offer would be acceptable to the answerer. Certain protocols, like SIP, provide a means to query for such capabilities. SDP can be used in responses to such queries to indicate capabilities. This section describes how such an SDP message is formatted. Since SDP has no way to indicate that the message is for the purpose of capability indication, this is determined from the context of the higher layer protocol. The ability of baseline SDP to indicate capabilities is very limited. It cannot express allowed parameter ranges or values, and can not be done in parallel with an offer/answer itself. Extensions might address such limitations in the future.

An SDP constructed to indicate media capabilities is structured as follows. It MUST be a valid SDP, except that it MAY omit both “e=“ and “p=“ lines. The “t=“ line MUST be equal to “0 0”. For each media type supported by the agent, there MUST be a corresponding media description of that type. The session ID in the origin field MUST be unique for each SDP constructed to indicate media capabilities. The port MUST be set to zero, but the connection address is arbitrary. The usage of port zero makes sure that an SDP formatted for capabilities does not cause media streams to be established if it is interpreted as an offer or answer.

The transport component of the “m=“ line indicates the transport for that media type. For each media format of that type supported by the agent, there SHOULD be a media format listed in the “m=“ line. In the case of RTP, if dynamic payload types are used, an rtpmap attribute MUST be present to bind the type to a specific format. There is no way to indicate constraints, such as how many simultaneous streams can be supported for a particular codec, and so on.

```
v=0
o=carol 28908764872 28908764872 IN IP4 100.3.6.6
s=-
t=0 0
c=IN IP4 192.0.2.4
m=audio 0 RTP/AVP 0 1 3
a=rtpmap:0 PCMU/8000
a=rtpmap:1 1016/8000
a=rtpmap:3 GSM/8000
m=video 0 RTP/AVP 31 34
a=rtpmap:31 H261/90000
a=rtpmap:34 H263/90000
```

Figure 1: SDP Indicating Capabilities

The SDP of Figure 1 indicates that the agent can support three audio codecs (PCMU, 1016, and GSM) and two video codecs (H.261 and H.263).

10 Example Offer/Answer Exchanges

This section provides example offer/answer exchanges.

10.1 Basic Exchange

Assume that the caller, Alice, has included the following description in her offer. It includes a bidirectional audio stream and two bidirectional video streams, using H.261 (payload type 31) and MPEG (payload type 32). The offered SDP is:

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.anywhere.com
s=
c=IN IP4 host.anywhere.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
m=video 51372 RTP/AVP 31
a=rtpmap:31 H261/90000
m=video 53000 RTP/AVP 32
a=rtpmap:32 MPV/90000
```

The callee, Bob, does not want to receive or send the first video stream, so he returns the SDP below as the answer:

```
v=0
o=bob 2890844730 2890844730 IN IP4 host.example.com
s=
c=IN IP4 host.example.com
t=0 0
m=audio 49920 RTP/AVP 0
a=rtpmap:0 PCMU/8000
m=video 0 RTP/AVP 31
m=video 53000 RTP/AVP 32
a=rtpmap:32 MPV/90000
```

At some point later, Bob decides to change the port where he will receive the audio stream (from 49920 to 65422), and at the same time, add an additional audio stream as receive only, using the RTP payload format for events [9]. Bob offers the following SDP in the offer:

```
v=0
o=bob 2890844730 2890844731 IN IP4 host.example.com
s=
c=IN IP4 host.example.com
t=0 0
m=audio 65422 RTP/AVP 0
a=rtpmap:0 PCMU/8000
m=video 0 RTP/AVP 31
m=video 53000 RTP/AVP 32
a=rtpmap:32 MPV/90000
m=audio 51434 RTP/AVP 110
a=rtpmap:110 telephone-events/8000
a=recvonly
```

Alice accepts the additional media stream, and so generates the following answer:

```
v=0
o=alice 2890844526 2890844527 IN IP4 host.anywhere.com
s=
c=IN IP4 host.anywhere.com
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
m=video 0 RTP/AVP 31
a=rtpmap:31 H261/90000
m=video 53000 RTP/AVP 32
a=rtpmap:32 MPV/90000
m=audio 53122 RTP/AVP 110
a=rtpmap:110 telephone-events/8000
a=sendonly
```

10.2 One of N Codec Selection

A common occurrence in embedded phones is that the Digital Signal Processor (DSP) used for compression can support multiple codecs at a time, but once that codec is selected, it cannot be readily changed on the fly. This example shows how a session can be set up using an initial offer/answer exchange, followed immediately by a second one to lock down the set of codecs.

The initial offer from Alice to Bob indicates a single audio stream with the three audio codecs that are available in the DSP. The stream is marked as inactive, since media cannot be received until a codec is locked down:

```
v=0
o=alice 2890844526 2890844526 IN IP4 host.anywhere.com
s=
c=IN IP4 host.anywhere.com
t=0 0
m=audio 62986 RTP/AVP 0 4 18
a=rtpmap:0 PCMU/8000
a=rtpmap:4 G723/8000
a=rtpmap:18 G729/8000
a=inactive
```

Bob can support dynamic switching between PCMU and G.723. So, he sends the following answer:

```
v=0
o=bob 2890844730 2890844731 IN IP4 host.example.com
s=
c=IN IP4 host.example.com
t=0 0
m=audio 54344 RTP/AVP 0 4
```

```
a=rtpmap:0 PCMU/8000
a=rtpmap:4 G723/8000
a=inactive
```

Alice can then select any one of these two codecs. So, she sends an updated offer with a sendrecv stream:

```
v=0
o=alice 2890844526 2890844527 IN IP4 host.anywhere.com
s=
c=IN IP4 host.anywhere.com
t=0 0
m=audio 62986 RTP/AVP 4
a=rtpmap:4 G723/8000
a=sendrecv
```

Bob accepts the single codec:

```
v=0
o=bob 2890844730 2890844732 IN IP4 host.example.com
s=
c=IN IP4 host.example.com
t=0 0
m=audio 54344 RTP/AVP 4
a=rtpmap:4 G723/8000
a=sendrecv
```

If the answerer (Bob), was only capable of supporting one-of-N codecs, Bob would select one of the codecs from the offer, and place that in his answer. In this case, Alice would do a re-INVITE to activate that stream with that codec.

As an alternative to using "a=inactive" in the first exchange, Alice can list all codecs, and as soon as she receives media from Bob, generate an updated offer locking down the codec to the one just received. Of course, if Bob only supports one-of-N codecs, there would only be one codec in his answer, and in this case, there is no need for a re-INVITE to lock down to a single codec.

11 Security Considerations

There are numerous attacks possible if an attacker can modify offers or answers in transit. Generally, these include diversion of media streams (enabling eavesdropping), disabling of calls, and injection of unwanted media streams. If a passive listener can construct fake offers, and inject those into an exchange, similar attacks are possible. Even if an attacker can simply observe offers and answers, they can inject media streams into an existing conversation.

Offer/answer relies on transport within an application signaling protocol, such as SIP. It also relies on that protocol for security capabilities. Because of the attacks described above, that protocol MUST provide a means for end-to-end authentication and integrity protection of offers and answers. It SHOULD offer encryption of bodies to prevent eavesdropping. However, media injection attacks can alternatively be resolved through authenticated media exchange, and therefore the encryption requirement is a SHOULD instead of a MUST.

Replay attacks are also problematic. An attacker can replay an old offer, perhaps one that had put media on hold, and thus disable media streams in a conversation. Therefore, the application protocol MUST provide a secure way to sequence offers and answers, and to detect and reject old offers or answers.

SIP [7] meets all of these requirements.

12 IANA Considerations

There are no IANA considerations with this specification.

13 Acknowledgements

The authors would like to thank Allison Mankin, Rohan Mahy, Joerg Ott, and Flemming Andreasen for their detailed comments.

Normative References

[1] M. Handley and V. Jacobson, "SDP: session description protocol," RFC 2327, Internet Engineering Task Force, Apr. 1998.

[2] S. Bradner, "Key words for use in RFCs to indicate requirement levels," RFC 2119, Internet Engineering Task Force, Mar. 1997.

[3] R. Kumar and M. Mostafa, "Conventions for the use of the session description protocol (SDP) for ATM bearer connections," RFC 3108, Internet Engineering Task Force, May 2001.

[4] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: a transport protocol for real-time applications," RFC 1889, Internet Engineering Task Force, Jan. 1996.

[5] H. Schulzrinne, "RTP profile for audio and video conferences with minimal control," RFC 1890, Internet Engineering Task Force, Jan. 1996.

Informative References

[6] M. Handley, C. Perkins, and E. Whelan, "Session announcement protocol," RFC 2974, Internet Engineering Task Force, Oct. 2000.

[7] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: session initiation protocol," RFC 3261, Internet Engineering Task Force, June 2002.

[8] H. Schulzrinne, A. Rao, and R. Lanphier, “Real time streaming protocol (RTSP),” RFC 2326, Internet Engineering Task Force, Apr. 1998.

[9] H. Schulzrinne and S. Petrack, “RTP payload for DTMF digits, telephony tones and telephony signals,” RFC 2833, Internet Engineering Task Force, May 2000.

[10] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, “SIP: session initiation protocol,” RFC 2543, Internet Engineering Task Force, Mar. 1999.

14 Authors’ Addresses

Jonathan Rosenberg
dynamicsoft
72 Eagle Rock Avenue
First Floor
East Hanover, NJ 07936

EMail: jdrosen@dynamicsoft.com

Henning Schulzrinne
Dept. of Computer Science
Columbia University
1214 Amsterdam Avenue
New York, NY 10027
USA

EMail: schulzrinne@cs.columbia.edu

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.