K. Hedayat
EXFO
N. Venna
Saperix
P. Jones
Cisco Systems, Inc.
A. Roychowdhury
Hughes Systique Corp.
C. SivaChelvan
Cisco Systems, Inc.
N. Stratton
BlinkMind, Inc.

September 6, 2011

An Extension to the Session Description Protocol (SDP) for Media Loopback

draft-ietf-mmusic-media-loopback-16

Status of this Memo

Copyright Notice

Abstract

The wide deployment of Voice over IP (VoIP), Real-time Text and Video over IP services has introduced new challenges in managing and maintaining voice/real-time Text/video quality, reliability, and overall performance. In particular, media delivery is an area that needs attention. One method of meeting these challenges is monitoring the media delivery performance by looping media back to the transmitter. This is typically referred to as "active monitoring" of services. Media loopback is especially popular in ensuring the quality of transport to the edge of a given VoIP, Real-time Text or Video over IP service. Today in networks that deliver real-time media, short of running 'ping' and 'traceroute' to the edge, service providers are left without the necessary tools to actively monitor, manage, and diagnose quality issues with their service. The extension defined herein adds new SDP media attributes which enables establishment of media sessions where the media is looped back to the transmitter. Such media sessions will serve as monitoring and troubleshooting tools by providing the means for measurement of more advanced VoIP, Real-time Text and Video over IP performance metrics.

Table of Contents

1. Introduction

The overall quality, reliability, and performance of VoIP, Real-time Text and Video over IP services rely on the performance and quality of the media path. In order to assure the quality of the delivered media there is a need to monitor the performance of the media transport. One method of monitoring and managing the overall quality of VoIP, Real-time Text and Video over IP Services is through monitoring the quality of the media in an active session. This type of "active monitoring" of services is a method of proactively managing the performance and quality of VoIP based services.

The goal of active monitoring is to measure the media quality of a VoIP, Real-time Text or Video over IP session. A way to achieve this goal is to request an endpoint to loop media back to the other endpoint and to provide media statistics (e.g., RTCP and RTCP XR information). Another method involves deployment of special endpoints that always loop incoming media back for sessions. Although the latter method has been used and is functional, it does not scale to support large networks and introduces new network management challenges. Further, it does not offer the granularity of testing a specific endpoint that may be exhibiting problems.

The extension defined in this memo introduces new SDP media attributes that enable establishment of media sessions where the media is looped back to the transmitter. The offer/answer model

[RFC3264] is used to establish a loopback connection. Furthermore, this extension provides guidelines on handling RTP [RFC3550], as well as usage of RTCP [RFC3550] and RTCP XR [RFC3611] for reporting media related measurements.

1.1 Use Cases Supported

As a matter of terminology in this document, packets flow from one peer acting as a "loopback source", to the other peer acting as a "loopback mirror", which in turn returns packets to the loopback source. In advance of the session, the peers negotiate to determine which one acts in which role. The negotiation also includes details such as the type of loopback to be used.

This specification supports three use cases: "encapsulated packet loopback", "direct loopback", and "media loopback". These are distinguished by the treatment of incoming RTP packets at the loopback mirror.

As a supplement to these use cases, this specification also allows the loopback source to request the loopback mirror to begin sending a media stream to the loopback source, ending when the mirror begins to receive packets from the source. This facility is needed in some circumstances to establish the media path through middleboxes lying between the peers.

1.1.1 Encapsulated Packet Loopback

In the encapsulated packet loopback case, the entire incoming RTP packet is encapsulated as payload within an outer payload type that is specific to this use case and specified below (Section 7.1). The encapsulated packet is returned to the loopback source. The loopback source can generate statistics for one-way path performance up to the RTP level for each direction of travel by examining sequence numbers and timestamps in the outer header and the encapsulated RTP packet payload. The loopback source can also play back the returned media content for evaluation.

Because the encapsulating payload extends the packet size, it could encounter difficulties in an environment where the original RTP packet size is close to the path MTU size. The encapsulating payload type therefore offers the possibility of RTP-level fragmentation of the returned packets. The use of this facility could affect statistics derived for the return path. In addition, the increased bit rate required in the return direction may affect these statistics more directly in a restricted-bandwidth situation.

1.1.2 Direct Loopback

In the direct loopback case, the loopback mirror copies the payload of the incoming RTP packet into a new packet, the payload type of which is again specific to this use case and specified below (Section 7.2). The loopback mirror returns the new packet to the packet source. There is no provision in this case for RTP-level fragmentation.

This use case has the advantage of keeping the packet size the same in both directions. The packet source can compute only two-way path statistics from the direct loopback packet header, but can play back the returned media content.

It has been suggested that the loopback source, knowing that the incoming packet will never be passed to a decoder, can store a timestamp and sequence number inside the payload of the packet it sends to the mirror, then extract that information from the returned direct loopback packet and compute one-way path statistics as in the previous case. Obviously, playout of returned content is no longer possible if this is done.

1.1.3 Media Loopback

In the media loopback case, the loopback mirror submits the incoming packet to a decoder appropriate to the incoming payload type. The packet is taken as close as possible to the analog level, then reencoded according to an outgoing format determined by negotiation. The reencoded content is returned to the loopback source as an RTP packet with payload type corresponding to the reencoding format.

This usage allows trouble-shooting at the codec level. The capability for path statistics is limited to what is available from RTCP reports.

2. Terminology

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

3. Offering Entity Behavior

An offering entity compliant to this memo and attempting to establish a media session with media loopback MUST include "loopback" media attributes for each individual media description in the offer message. The offering entity MUST look for the "loopback" media attributes in the media description(s) of the response from the answering entity for confirmation that the request is accepted.

4. Answering Entity Behavior

An answering entity compliant to this specification and receiving an offer containing media descriptions with the "loopback" media attributes MUST acknowledge the request by including the received "loopback" media attributes for each media description in its response if it agrees to do the loopback. If the answerer does not want to do loopback or wants to reject the "loopback" request for specific media types, it MAY do so as defined in section 5.4.1 of this specification.

An answering entity that is not compliant to this specification and which receives an offer with the "loopback" media attributes MAY ignore the attribute and treat the incoming offer as a normal request.

5. SDP Constructs Syntax

Two new media attributes are defined: one indicates the type of loopback and the other indicates the mode of the loopback.

5.1 Loopback Type Attribute

The loopback type is a property media attribute with the following syntax:

```
a=loopback:<loopback-type>
```

Following is the Augmented BNF [RFC5234] for loopback-type:

```
loopback-type          = loopback-choice [1*SP loopback-choice]
loopback-choice        = loopback-type-pkt / loopback-type-media
loopback-type-pkt      = "rtp-pkt-loopback"
loopback-type-media    = "rtp-media-loopback"
```

The loopback type is used to indicate the type of loopback. The loopback-type values are rtp-pkt-loopback, and rtp-media-loopback.

rtp-pkt-loopback: In this mode, the RTP packets are looped back to the sender at a point before the encoder/decoder function in the receive direction to a point after the encoder/decoder function in the send direction. This effectively re-encapsulates the RTP payload with the RTP/UDP/IP headers appropriate for sending it in the reverse direction. Any type of encoding related functions, such as packet loss concealment, MUST NOT be part of this type of loopback path. In this mode the RTP packets are looped back with a new payload type and format. Section 7 describes the payload formats that MUST be used for this type of loopback.

rtp-media-loopback: This loopback is activated as close as possible to the analog interface and after the decoder so that the RTP packets are subsequently re-encoded prior to transmission back to the sender.

5.2 Loopback Mode Attribute

The loopback mode is a value media attribute that is used to indicate the mode of the loopback. These attributes are additional mode attributes like sendonly, recvonly, etc. The syntax of the loopback mode media attribute is:

```
a=<loopback-mode>:<fmt>...
```

The loopback-mode values are loopback-source and loopback-mirror.

loopback-source: This attribute specifies that the entity that generated the SDP is the media source and expects the receiver of the SDP message to act as a loopback-mirror.

loopback-mirror: This attribute specifies that the entity that generated the SDP will mirror (echo) all received media back to the sender of the RTP stream. No media is generated locally by the looping back entity for transmission in the mirrored stream.

<fmt> is a media format description. The format description has the semantics as defined in section 5.14 of RFC 4566[RFC4566]. When

loopback-mode is specified as loopback-source, the media format corresponds to the RTP payload types the entity that generated the SDP is willing to send. When loopback-mode is specified as loopback-mirror, the media format corresponds to the RTP payload types the mirror is willing to receive. The "m=" line in the SDP MUST include all the payload types that will be used during the loopback session including those specified in the loopback-mode attribute line. The complete payload space for the call is specified in the "m=" line and the rtpmap attribute is used to map from the payload type number to an encoding name denoting the payload format to be used.

5.3 Generating the Offer for Loopback Session

If an offerer wishes to make a loopback request, it MUST include both the loopback-type and loopback-mode parameters in a valid SDP offer:

```
Example:   m=audio 41352 RTP/AVP 0 8 100
           a=loopback:rtp-media-loopback
           a=loopback-source:0 8 100
           a=rtpmap:0 pcmu/8000
           a=rtpmap:8 pcma/8000
           a=rtpmap:100 G7221/16000/1
```

Note: A loopback offer in a given media description MUST NOT contain the standard mode attributes sendonly, recvonly, sendrecv, or inactive. The loopback-mode attributes (loopback-source and loopback-mirror) replace the standard attributes.

The offerer may offer more than one loopback-type in the SDP offer. The port number and the address in the offer (m= line) indicate where the offerer would like to send and receive the media stream. The payload type numbers indicate the value of the payload the offerer expects to send and receive. If the offerer is the loopback-source, the subset of payload types indicated in the a=loopback-source line are the payload types for the codecs the offerer is willing to send. However, the answer might indicate a different payload type number for the same codec in the loopback-mirror line. In that case, the offerer MUST send the payload type received in the answer. If the offerer is the loopback-mirror, the subset of payload types indicated in the a=loopback-mirror line are the payload types for the codecs the offerer is willing to receive.

If loopback-type is rtp-pkt-loopback, the loopback-mirror MUST send and the loopback-source MUST receive the looped back packets

encoded in one of the two payload formats (encapsulated RTP or payload loopback) as defined in section 7.

```
Example:   m=audio 41352 RTP/AVP 0 8 112
           a=loopback:rtp-pkt-loopback
           a=loopback-source:0 8
           a=rtpmap:112 encaprtp/8000

Example:   m=audio 41352 RTP/AVP 0 8 112
           a=loopback:rtp-pkt-loopback
           a=loopback-source:0 8
           a=rtpmap:112 rtploopback/8000
```

5.4 Generating the Answer for Loopback Session

As with the offer, a loopback answer in a given media description MUST NOT contain the standard mode attributes sendonly, recvonly, sendrecv, or inactive. The loopback-mode attributes (loopbackThe port number and the address in the answer (m= line) indicate where the answerer would like to receive the media stream. The payload type numbers indicate the value of the payload types the answerer expects to send and receive. If the offerer is the loopback-source, the answerer MUST be a loopback-mirror and the subset of payload types indicated in the a=loopback-mirror line are the payload types for the codecs the answerer is willing to receive. Similarly, if the offerere is the loopback-mirror, the answerer MUST be aloopback-source and the subset of payload types indicated in the a=loopback-source line are the payload types for the codecs the answerer is willing to send.

If an answerer wishes to accept the loopback request it MUST include both the loopback mode and loopback type attribute in the answer. When a stream is offered with the loopback-source attribute, the corresponding stream in the response MUST be loopback-mirror and vice versa, provided that answerer is capable of supporting the requested loopback-type.

For example, if the offer contains the loopback-source attribute:

```
m=audio 41352 RTP/AVP 0 8
a=loopback:rtp-media-loopback
a=loopback-source:0 8
```

The answer that is capable of supporting the offer MUST contain the loopback-mirror attribute:

```
m=audio 41352 RTP/AVP 0 8
```

```
a=loopback:rtp-media-loopback
a=loopback-mirror:0 8
```

If a stream is offered with multiple loopback type attributes, the answer MUST include only one of the loopback types that are accepted by the answerer. The answerer SHOULD give preference to the first loopback-type in the SDP offer.

For example, if the offer contains:

```
m=audio 41352 RTP/AVP 0 8 112
a=loopback:rtp-media-loopback rtp-pkt-loopback
a=loopback-source:0 8
a=rtpmap:112 encaprtp/8000
```

The answer that is capable of supporting the offer and chooses to loopback the media using the rtp-media-loopback type MUST contain:

```
m=audio 41352 RTP/AVP 0 8
a=loopback:rtp-media-loopback
a=loopback-mirror:0 8
```

As specified in section 7, if the loopback-type is rtp-pkt-loopback, either the encapsulated RTP payload format or direct loopback RTP payload format MUST be used for looped back packets.

For example, if the offer contains:

```
m=audio 41352 RTP/AVP 0 8 112 113
a=loopback:rtp-pkt-loopback
a=loopback-source:0 8
a=rtpmap:112 encaprtp/8000
a=rtpmap:113 rtploopback/8000
```

The answer that is capable of supporting the offer must contain one of the following:

```
m=audio 41352 RTP/AVP 0 8 112
a=loopback:rtp-pkt-loopback
a=loopback-mirror:0 8
a=rtpmap:112 encaprtp/8000

m=audio 41352 RTP/AVP 0 8 113
a=loopback:rtp-pkt-loopback
a=loopback-mirror:0 8
a=rtpmap:113 rtploopback/8000
```

5.4.1 Rejecting the Loopback Offer

An offered stream (either with loopback-source or loopback-mirror) MAY be rejected if the loopback-type is not specified, the specified loopback-type is not supported, or the endpoint cannot honor the offer for any other reason. The Loopback request may be rejected by setting the media port number to zero in the answer as per RFC 3264 [RFC3264].

5.5 Offerer Processing of the Answer

The answer to a loopback-source MUST be loopback-mirror. The answer to a loopback-mirror MUST be loopback-source. The loopback-mode line MUST contain at least one codec the answerer is willing to send or receive depending on whether it is the loopback-source or the loopback-mirror. In addition, the "m=" line MUST contain at least one codec that the answerer is willing to send or receive depending on whether it is the loopback-mirror or the loopback-source.

If the answer does not contain a=loopback-mirror or a=loopback-source, it is assumed that the loopback extensions are not supported by the target UA.

5.6 Modifying the Session

At any point during the loopback session, either participant may issue a new offer to modify the characteristics of the previous session. In case of SIP this is defined in section 8 of RFC 3264 [RFC3264]. This also includes transitioning from a normal media processing mode to loopback mode, and vice a versa.

5.7 Establishing Sessions Between Entities Behind NAT

ICE/STUN/TURN provide a general solution to establishing media sessions between entities that are behind NATs. Loopback sessions that involve one or more end points behind NATs SHOULD use these general solutions wherever possible.

6. RTP Requirements

A loopback-mirror that is compliant to this specification and accepting a media with rtp-pkt-loopback loopback-type MUST loopback the incoming RTP packets using either the encapsulated RTP payload format or the direct loopback RTP payload format as defined in section 7 of this specification.

An answering entity that is compliant to this specification and accepting a media with the loopback type rtp-media-loopback MUST transmit all received media back to the sender. The incoming media MUST be treated as if it were to be played (e.g. the media stream MAY receive treatment from PLC algorithms). The answering entity MUST re-generate all the RTP header fields as it would when transmitting media. The answering entity MAY choose to encode the loopback media according to any of the media descriptions supported by the offering entity. Furthermore, in cases where the same media type is looped back, the answering entity MAY choose to preserve number of frames/packet and bitrate of the encoded media according to the received media.

7. Payload formats for Packet loopback

The payload formats described in this section MUST be used by a loopback-mirror when rtp-pkt-loopback is the specified loopback-type. Two different formats are specified here - an encapsulated RTP payload format and a direct loopback RTP payload format. The encapsulated RTP payload format should be used when the incoming RTP header information needs to be preserved during the loopback operation. This is useful in cases where loopback source needs to measure performance metrics in both directions. However, this comes at the expense of increased packet size as described in section 7.1. The direct loopback RTP payload format should be used when bandwidth requirement prevents the use of encapsulated RTP payload format.

To keep the implementation of loopback-mirrors simple it is mandated that no payload format other than encapsulated or direct loopback formats can be used in the packets generated by a loopback-mirror. As described in RFC 3550 [RFC3550], sequence numbers and timestamps in the RTP header are generated with initial random values for security reasons. If this were not mandated and the source payload is sequence number aware, the loopback-mirror will be required to understand that payload format to generate looped back packets that do not violate RFC 3550 [RFC3550]. Requiring looped back packets to be in one of the two formats means loopback-mirror does not have to look into the actual payload received before generating the loopback packets.

7.1 Encapsulated Payload format

A received RTP packet is encapsulated in the payload section of the RTP packet generated by a loopback-mirror. Each received packet MUST be encapsulated in a different packet, the encapsulated packet MUST be fragmented only if required (for example: due to MTU limitations).

7.1.1 Usage of RTP Header fields

Payload Type (PT): The assignment of an RTP payload type for this packet format is outside the scope of this document; it is either specified by the RTP profile under which this payload format is used or more likely signaled dynamically out-of-band (e.g., using SDP; section 7.1.3 defines the name binding).

Marker (M) bit: If the received RTP packet is looped back in multiple RTP packets, the M bit is set to 1 in the last packet, otherwise it is set to 0.

Extension (X) bit: Defined by the RTP Profile used.

Sequence Number: The RTP sequence number SHOULD be generated by the loopback-mirror in the usual manner with a constant random offset as described in RFC 3550 [RFC3550].

Timestamp: The RTP timestamp denotes the sampling instant for when the loopback-mirror is transmitting this packet to the loopback-source. The RTP timestamp MUST use the same clock rate used by the loopback-source. The initial value of the timestamp SHOULD be random for security reasons (see Section 5.1 of RFC 3550 [RFC3550]).

SSRC: set as described in RFC 3550 [RFC3550].

CC and CSRC fields are used as described in RFC 3550 [RFC3550].

7.1.2 RTP Payload Structure

The RTP header in the encapsulated packet MUST be followed by the payload header defined in this section. If the received RTP packet has to be looped back in multiple packets due to fragmentation, the RTP header in each packet MUST be followed by the payload header defined in this section. The header is devised so that the loopback-source can decode looped back packets in the presence of moderate packet loss [RFC3550].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      receive timestamp                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| F | R |  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        transmit timestamp                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The 12 octets after the receive timestamp are identical to the RTP header in the received packet except for the first 4 bits of the first octet.

Receive Timestamp: 32 bits

The Receive timestamp denotes the sampling instant for when the last octet of the received media packet that is being encapsulated by the loopback-mirror is received from the loopback-source. The Receive timestamp MUST be based on the same clock used by the loopback-source. The initial value of the timestamp SHOULD be random for security reasons (see Section 5.1 of RFC 3550 [RFC3550]).

Fragmentation (F): 2 bits

First Fragment (00) /Last Fragment (01) /No Fragmentation(10)/ Intermediate Fragment (11). This field identifies how much of the received packet is encapsulated in this packet by the loopback-mirror. If the received packet is not fragmented, this field is set to 10; otherwise the packet that contains the first fragments sets this field to 00, the packet that contains the last fragment sets this field to 01, all other packets set this field to 11.

Reserved: 2 bits

This field is reserved for future definition. In the absence of such a definition, the bits in this field MUST be set to zero and MUST be ignored by the receiver.

Any padding octets in the original packet MUST NOT be included in the loopback packet generated by a loopback-mirror. The loopback-mirror MAY add padding octets if required.

7.1.3 Usage of SDP

The payload type number for the encapsulated stream can be negotiated using a mechanism like SDP. There is no static payload type assignment for the encapsulated stream, so dynamic payload type numbers MUST be used. The binding to the name is indicated by an rtpmap attribute. The name used in this binding is "encaprtp".

The following is an example SDP fragment for encapsulated RTP.

```
m=audio 41352 RTP/AVP 112
a=rtpmap:112 encaprtp/8000
```

7.2 Direct loopback RTP payload format

The direct loopback RTP payload format can be used in scenarios where the 16 byte overhead of the encapsulated payload format is significant. This payload format MUST NOT be used in cases where the MTU on the loopback path will cause fragmentation of looped back RTP packets. When using this payload format, the receiver MUST loop back each received packet in a separate RTP packet.

7.2.1 Usage of RTP Header fields

Payload Type (PT): The assignment of an RTP payload type for this packet format is outside the scope of this document; it is either specified by the RTP profile under which this payload format is used or more likely signaled dynamically out-of-band (e.g., using SDP; section 7.2.3 defines the name binding).

Marker (M) bit: Set to the value in the received packet.

Extension (X) bit: Defined by the RTP Profile used.

Sequence Number: The RTP sequence number SHOULD be generated by the loopback-mirror in the usual manner with a constant random offset.

Timestamp: The RTP timestamp denotes the sampling instant for when the loopback-mirror is transmitting this packet to the loopback-source. The RTP timestamp MUST be based on the same clock used by the loopback-source. The initial value of the timestamp SHOULD be random for security reasons (see Section 5.1 of RFC 3550 [RFC3550]).

SSRC: set as described in RFC 3550 [RFC3550].

CC and CSRC fields are used as described in RFC 3550 [RFC3550].

7.2.2 RTP Payload Structure

This payload format does not define any payload specific headers. The loopback-mirror simply copies the payload data from the payload portion of the packet received from the loopback-source.

7.2.3 Usage of SDP

The payload type number for the payload loopback stream can be negotiated using a mechanism like SDP. There is no static payload type assignment for the stream, so dynamic payload type numbers MUST be used. The binding to the name is indicated by an rtpmap attribute. The name used in this binding is "rtploopback".

The following is an example SDP fragment for direct loopback RTP format.

```
m=audio 41352 RTP/AVP 112
a=rtpmap:112 rtploopback/8000
```

8. RTCP Requirements

The use of the loopback attribute is intended for monitoring of media quality of the session. Consequently the media performance information should be exchanged between the offering and the answering entities. An offering or answering entity that is compliant to this specification SHOULD support RTCP per [RFC3550] and RTCP-XR per RFC 3611 [RFC3611]. Furthermore, if the client or the server choose to support RTCP-XR, they SHOULD support RTCP-XR Loss RLE report block, Duplicate RLE report block, Statistics Summary report block, and VoIP Metric Reports Block per sections 4.1, 4.2, 4.6, and 4.7 of RFC 3611 [RFC3611]. The client and the server MAY support other RTCP-XR reporting blocks as defined by RFC 3611 [RFC3611].

9. Congestion Control

All the participants in a loopback session SHOULD implement congestion control mechanisms as defined by the RTP profile under which the loopback mechanism is implemented. For audio video profiles, implementations SHOULD conform to the mechanism defined in Section 2 of RFC 3551.

10. Examples

This section provides examples for media descriptions using SDP for different scenarios. The examples are given for SIP-based transactions and are abbreviated and do not show the complete signaling for convenience.

10.1 Offer for specific media loopback type

A client sends an INVITE request with offer SDP which looks like:

```
v=0
o=alice 2890844526 2890842807 IN IP4 host.atlanta.example.com
s=Example
i=An example session
e=alice@example.com
c=IN IP4 host.atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 0
a=loopback:rtp-media-loopback
a=loopback-source:0
a=rtpmap:0 pcmu/8000
```

The client is offering to source the media and expects the server to mirror the RTP stream per rtp-media-loopback loopback type.

A server sends a response with answer SDP which looks like:

```
v=0
o=bob 2890844526 2890842807 IN IP4 host.biloxi.example.com
s=Example
i=An example session
e=bob@example.com
c=IN IP4 host.biloxi.example.com
t=0 0
m=audio 49270 RTP/AVP 0
a=loopback:rtp-media-loopback
a=loopback-mirror:0
a=rtpmap:0 pcmu/8000
```

The server is accepting to mirror the media from the client at the media level.

10.2

Offer for choice of media loopback type

A client sends an INVITE request with offer SDP which looks like:

```
v=0
o=alice 2890844526 2890842807 IN IP4 host.atlanta.example.com
s=Example
i=An example session
e=alice@example.com
c=IN IP4 host.atlanta.example.com
t=0 0
m=audio 49170 RTP/AVP 0 112 113
a=loopback:rtp-media-loopback rtp-pkt-loopback
a=loopback-source:0
a=rtpmap:0 pcmu/8000
a=rtpmap:112 encaprtp/8000
a=rtpmap:113 rtploopback/8000
```

The client is offering to source the media and expects the server to mirror the RTP stream at either the media or rtp level.

A server sends a response with answer SDP which looks like:

```
v=0
o=box 2890844526 2890842807 IN IP4 host.biloxi.example.com
s=Example
i=An example session
e=bob@example.com
c=IN IP4 host.biloxi.example.com
t=0 0
m=audio 49270 RTP/AVP 0 112
a=loopback:rtp-pkt-loopback
a=loopback-mirror:0
a=rtpmap:0 pcmu/8000
a=rtpmap:112 encaprtp/8000
```

The server is accepting to mirror the media from the client at the packet level using the encapsulated RTP payload format.

10.3 Response to INVITE request rejecting loopback media

A client sends an INVITE request with offer SDP which looks like:

```
v=0
o=alice 2890844526 2890842807 IN IP4 host.atlanta.example.com
s=Example
i=An example session
e=user@example.com
c=IN IP4 host.atlanta.example.com
```

```
t=0 0
m=audio 49170 RTP/AVP 0
a=loopback:rtp-media-loopback
a=loopback-source:0
a=rtpmap:0 pcmu/8000
```

The client is offering to source the media and expects the server to mirror the RTP stream at the media level.

A server sends a response with answer SDP which looks like:

```
v=0
o=bob 2890844526 2890842807 IN IP4 host.biloxi.example.com
s=Example
i=An example session
e=user@example.com
c=IN IP4 host.biloxi.example.com
t=0 0
m=audio 0 RTP/AVP 0
a=loopback:rtp-media-loopback
a=loopback-mirror:0
a=rtpmap:0 pcmu/8000
```

NOTE: Loopback request may be rejected by either not including the loopback mode attribute (for backward compatibility) or setting the media port number to zero, or both, in the response.

11. Security Considerations

The security considerations of [RFC3261] and [RFC3264] apply. Furthermore, given that media loopback may be automated without the end user's knowledge, the server of the media loopback should be aware of denial of service attacks. It is recommended that sessions with media loopback are authenticated and the frequency of such sessions is limited by the server.

12. Implementation Considerations

The media loopback approach described in this document is a complete solution that would work under all scenarios. However, it is believed that the solution may not be light-weight enough for the common case. In light of this concern, this section clarifies which features of the loopback proposal MUST be implemented for all implementations and which features MAY be deferred if the complete solution is not desired.

All implementations MUST support the rtp-pkt-loopback option for loopback-type attribute. In addition, for the loopback-mode attribute, all implementations of an offerer MUST at a minimum be able to act as a loopback-source. All implementation MUST also at a minimum support the direct media loopback payload type. The rtp-media-loopback attribute MAY be implemented in complete implementations of this draft.

13. IANA Considerations

13.1 SDP Attributes

This document defines three new media-level SDP attributes. IANA has registered the following attributes:

```
   Contact name:              Kaynam Hedayat
<kaynam.hedayat@exfo.com>.
   Attribute name:            "loopback".
   Type of attribute:         Media level.
   Subject to charset:        No.
   Purpose of attribute:      The 'loopback' attribute is used to
                              indicate the type of media loopback.
   Allowed attribute values:  The parameters to 'loopback' may be
                              one or more of "rtp-pkt-loopback" and
                              "rtp-media-loopback". See section 5
                              of this document for syntax.

   Contact name:              Kaynam Hedayat
<kaynam.hedayat@exfo.com>.
   Attribute name:            "loopback-source".
   Type of attribute:         Media level.
   Subject to charset:        No.
   Purpose of attribute:      The 'loopback-source' attribute
                              specifies that the sender is the media
                              source and expects the receiver to act
                              as a loopback-mirror.
   Allowed attribute values:  The parameter to 'loopback-source' is
                              a media format ("<fmt>") description
                              as defined in RFC 4566 Section 5.14.

   Contact name:              Kaynam Hedayat
<kaynam.hedayat@exfo.com>.
   Attribute name:            "loopback-mirror".
   Type of attribute:         Media level.
   Subject to charset:        No.
```

Purpose of attribute: The 'loopback-mirror' attribute specifies that the receiver will mirror (echo) all received media back to the sender of the RTP stream.
Allowed attribute values: The parameter to 'loopback-mirror' is a media format ("<fmt>") description as defined in RFC 4566 Section 5.14.

13.2
MIME Types

The IANA has registered the following MIME types:

13.2.1 audio/encaprtp

To: ietf-types@iana.org

Subject: Registration of media type audio/encaprtp

Type name: audio

Subtype name: encaprtp

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

13.2.2 video/encaprtp

To: ietf-types@iana.org

Subject: Registration of media type video/encaprtp

Type name: video

Subtype name: encaprtp

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

13.2.3 text/encaprtp

To: ietf-types@iana.org

Subject: Registration of media type text/encaprtp

Type name: text

Subtype name: encaprtp

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

13.2.4 application/encaprtp

To: ietf-types@iana.org

Subject: Registration of media type application/encaprtp

Type name: application

Subtype name: encaprtp

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

13.2.5 audio/rtploopback

To: ietf-types@iana.org

Subject: Registration of media type audio/rtploopback

Type name: audio

Subtype name: rtploopback

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

13.2.6 video/rtploopback

To: ietf-types@iana.org

Subject: Registration of media type video/rtploopback

Type name: video

Subtype name: rtploopback

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

13.2.7 text/rtploopback

To: ietf-types@iana.org

Subject: Registration of media type text/rtploopback

Type name: text

Subtype name: rtploopback

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

13.2.8 application/rtploopback

To: ietf-types@iana.org

Subject: Registration of media type
application/rtploopback

Type name: application

Subtype name: rtploopback

Required parameters:

rate:RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000; other rates may be specified.

Optional parameters: none

Encoding considerations: This media type is framed binary data.

Security considerations: See Section 12 of this document.

Interoperability considerations: none

Published specification: This MIME type is described fully within this document.

Applications which use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP, Real-Time Text or Video Over IP Service.

Additional information: none

Person & email address to contact for further information:

Kaynam Hedayat
EMail: kaynam.hedayat@exfo.com

Intended usage: COMMON

Restrictions on usage: This media type depends on RTP framing, and hence is only defined for transfer via RTP. Transfer within other framing protocols is not defined at this time.

Author:
Kaynam Hedayat.

Change controller: IETF Audio/Video Transport working group delegated from the IESG.

14. Normative References

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3611] Almeroth, K., Caceres, R., Clark, A., Cole, R., Duffield, N., Friedman, T., Hedayat, K., Sarac, K. and M. Westerlund, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

[RFC5234] Crocker, P. Overell, "Augmented ABNF for Syntax Specification: ABNF", RFC 5234, October 2005.

[RFC2119] Bradner, S.,"Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2736] Handley, M., Perkins, C., "Guidelines for Writers of RTP Payload Format Specifications", RFC 2736, BCP 0036, December 1999.

[RFC3551] Schulzrinne, H., Casner, S., "RTP Profile for Audio and Video Conferences with Minimial Control", STD 65, RFC 3551, July 2003.

[RFC4566] Handley, M., Jacobson, V., Perkins, C., "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, February 2007.

Authors' Addresses

Kaynam Hedayat
EXFO
285 Mill Road
Chelmsford, MA 01824
US

Phone: +1 978 367 5611
EMail: kaynam.hedayat@exfo.com
URI: http://www.exfo.com/

Nagarjuna Venna
Saperix
738 Main Street, #398
Waltham, MA 02451
US

Phone: +1 978 367 5703
EMail: vnagarjuna@saperix.com
URI: http://www.saperix.com/

Paul E. Jones
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
US

Phone: +1 919 392 6948
EMail: paulej@packetizer.com
URI: http://www.cisco.com/

Arjun Roychowdhury
Hughes Systique Corp.
15245 Shady Grove Rd, Ste 330
Rockville MD 20850

US

Phone: +1 301 527 1629
EMail: arjun@hsc.com
URI: http://www. hsc.com/

Chelliah SivaChelvan
Cisco Systems, Inc.
2200 East President George Bush Turnpike
Richardson, TX 75082
US

Phone: +1 972 813 5224
EMail: chelliah@cisco.com
URI: http://www.cisco.com/

Nathan Stratton
BlinkMind, Inc.
2027 Briarchester Dr.
Katy, TX 77450

Phone: +1 832 330 3810
EMail: nathan@robotics.net
URI: http://www.robotics.net/