Internet Engineering Task Force (IETF)
Request for Comments: 6849
Category: Standards Track
ISSN: 2070-1721

H. Kaplan, Ed.
Acme Packet
K. Hedayat
EXFO
N. Venna
Saperix
P. Jones
Cisco Systems, Inc.
N. Stratton
BlinkMind, Inc.
February 2013

An Extension to the Session Description Protocol (SDP) and Real-time Transport Protocol (RTP) for Media Loopback

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 real-time voice/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, administrators are left without the necessary tools to actively monitor, manage, and diagnose quality issues with their service. The extension defined herein adds new Session Description Protocol (SDP) media types and attributes that enable 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.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6849.

Copyright Notice

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 real-time 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., RTP Control Protocol (RTCP) [RFC3550] and RTCP Extended Reports (RTCP-XR) [RFC3611] information). Another method involves deployment of special endpoints that always loop incoming media back for all 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 document introduces new SDP media types and attributes that enable establishment of media sessions where the media is looped back to the transmitter. The SDP 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, using the SDP offer/answer exchange. 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.

1.1.1. Encapsulated Packet Loopback

In the encapsulated packet loopback case, the entire incoming RTP packet is encapsulated as payload within an outer RTP packet that is specific to this use case and specified in 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 encapsulating outer RTP header and the encapsulated RTP packet payload. The loopback source can also play back the returned media content for evaluation.

Because the encapsulating RTP packet header extends the packet size, it could encounter difficulties in an environment where the original RTP packet size is close to the path Maximum Transmission Unit (MTU) size. The encapsulating payload format 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 RTP packet, using a payload format specific to this use case and specified in 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 re-encoded according to an outgoing format determined by SDP negotiation. The re-encoded content is returned to the loopback source as an RTP packet with payload type corresponding to the re-encoding format.

This usage allows troubleshooting 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 [RFC2119].

SDP: Session Description Protocol, as defined in [RFC4566]. This document assumes that the SDP offer/answer model is followed, per [RFC3264], but does not assume any specific signaling protocol for carrying the SDP.

The following terms are borrowed from [RFC3264] definitions: offer, offerer, answer, answerer, and agent.

3. Overview of Operation

This document defines two loopback 'types', two 'roles', and two encoding formats for loopback. For any given SDP offerer or answerer pair, one side is the source of RTP packets, while the other is the mirror looping packets/media back. Those define the two loopback roles. As the mirror, two 'types' of loopback can be performed: packet-level or media-level. When media-level is used, there is no further choice of encoding format -- there is only one format: whatever is indicated for normal media, since the "looping" is performed at the codec level. When packet-level looping is performed, however, the mirror can either send back RTP in an encapsulated format or direct loopback format. The rest of this document describes these loopback types, roles, and encoding formats, and the SDP offer/answer rules for indicating them.

3.1. SDP Offerer Behavior

An SDP offerer compliant to this specification and attempting to establish a media session with media loopback will include "loopback" media attributes for each individual media description in the offer message that it wishes to have looped back. Note that the offerer may choose to only request loopback for some media descriptions/streams but not others. For example, it might wish to request loopback for a video stream but not audio, or vice versa.

The offerer will look for the "loopback" media attributes in the media description(s) of the response from the SDP answer for confirmation that the request is accepted.

3.2. SDP Answerer Behavior

In order to accept a loopback offer (that is, an offer containing "loopback" in the media description), an SDP answerer includes the "loopback" media attribute in each media description for which it desires loopback.

An answerer can reject an offered stream (either with loopback-source or loopback-mirror) 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 is rejected by setting the stream's media port number to zero in the answer as defined in RFC 3264 [RFC3264] or by rejecting the entire offer (i.e., by rejecting the session request entirely).

Note that an answerer that is not compliant to this specification and that receives an offer with the "loopback" media attributes would ignore the attributes and treat the incoming offer as a normal request. If the offerer does not wish to establish a "normal" RTP session, it would need to terminate the session upon receiving such an answer.

4. New SDP Attributes

Three new SDP media-level attributes are defined: one indicates the type of loopback, and the other two define the role of the agent.

4.1. Loopback-Type Attribute

This specification defines a new "loopback" attribute, which indicates that the agent wishes to perform loopback, and the type of loopback that the agent is able to do. The loopback-type is a value media attribute [RFC4566] with the following syntax:

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

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

```
attribute              =/ loopback-attr
; attribute defined in RFC 4566

loopback-attr          = "loopback:" SP 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 are to be used for this type of loopback. This type of loopback applies to the encapsulated and direct loopback use cases described in Section 1.1.

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. This type of loopback applies to the media loopback use case described in Section 1.1.3.

4.2. Loopback-Role Attributes: loopback-source and loopback-mirror

The loopback role defines two property media attributes [RFC4566] that are used to indicate the role of the agent generating the SDP offer or answer. The syntax of the two loopback-role media attributes is as follows:

```
a=loopback-source
```

and

```
a=loopback-mirror
```

Following is the Augmented BNF [RFC5234] for loopback-source and loopback-mirror:

```
attribute             =/ loopback-source / loopback-mirror
; attribute defined in RFC 4566
loopback-source       = "loopback-source"
loopback-mirror       = "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.

The "m=" line in the SDP includes all the payload types that will be used during the loopback session. The complete payload space for the session 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. Rules for Generating the SDP Offer/Answer

5.1. Generating the SDP Offer for Loopback Session

If an offerer wishes to make a loopback request, it includes both the loopback-type and loopback-role attributes in a valid SDP offer:

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

Since media loopback requires bidirectional RTP, its normal direction mode is "sendrecv"; the "sendrecv" direction attribute MAY be encoded in SDP or not, as per Section 5.1 of [RFC3264], since it is implied by default. If either the loopback source or mirror wishes to disable loopback use during a session, the direction mode attribute "inactive" MUST be used as per [RFC3264]. The direction mode attributes "recvonly" and "sendonly" are incompatible with the loopback mechanism and MUST NOT be indicated when generating an SDP offer or answer. When receiving an SDP offer or answer, if "recvonly" or "sendonly" is indicated for loopback, the SDP-receiving agent SHOULD treat it as a protocol failure of the loopback negotiation and terminate the session through its normal means (e.g., by sending a SIP BYE if SIP is used) or reject the offending media stream.

The offerer may offer more than one loopback-type in the SDP offer. The port number and the address in the offer (m/c= lines) indicate where the offerer would like to receive the media stream(s). The payload type numbers indicate the value of the payload the offerer expects to receive. However, the answer might indicate a subset of payload type numbers from those given in the offer. In that case, the offerer MUST only send the payload types received in the answer, per normal SDP offer/answer rules.

If the offer indicates rtp-pkt-loopback support, the offer MUST also contain either an encapsulated or direct loopback encoding format encoding name, or both, as defined in Sections 7.1 and 7.2 of this document. If the offer only indicates rtp-media-loopback support, then neither encapsulated nor direct loopback encoding formats apply and they MUST NOT be in the offer.

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 direct loopback) as defined in Section 7.

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

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

5.2. Generating the SDP Answer for Loopback Session

As with the offer, an SDP answer for loopback follows SDP offer/answer rules for the direction attribute, but directions of "sendonly" or "recvonly" do not apply for loopback operation.

The port number and the address in the answer (m/c= lines) 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.

An answerer includes both the loopback-role and loopback-type attributes in the answer to indicate that it will accept the loopback request. When a stream is offered with the loopback-source attribute, the corresponding stream in the response will be loopback-mirror and vice versa, provided the 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
```

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

```
m=audio 12345 RTP/AVP 0 8
a=loopback:rtp-media-loopback
a=loopback-mirror
```

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
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 12345 RTP/AVP 0 8
a=loopback:rtp-media-loopback
a=loopback-mirror
```

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
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 12345 RTP/AVP 0 8 112
a=loopback:rtp-pkt-loopback
a=loopback-mirror
a=rtpmap:112 encaprtp/8000

m=audio 12345 RTP/AVP 0 8 113
a=loopback:rtp-pkt-loopback
a=loopback-mirror
a=rtpmap:113 rtploopback/8000
```

The previous examples used the 'encaprtp' and 'rtploopback' encoding names, which will be defined in Sections 7.1.3 and 7.2.3.

5.3. Offerer Processing of the SDP Answer

If the received SDP answer does not contain an a=loopback-mirror or a=loopback-source attribute, it is assumed that the loopback extensions are not supported by the remote agent. This is not a protocol failure and instead merely completes the SDP offer/answer exchange with whatever normal rules apply; the offerer MAY decide to end the established RTP session (if any) through normal means of the upper-layer signaling protocol (e.g., by sending a SIP BYE).

5.4. 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, as defined in Section 8 of RFC 3264 [RFC3264]. This also includes transitioning from a normal media processing mode to loopback mode, and vice versa.

5.5. Establishing Sessions between Entities behind NATs

Interactive Connectivity Establishment (ICE) [RFC5245], Traversal Using Relays around NAT (TURN) [RFC5766], and Session Traversal Utilities for NAT (STUN) [RFC5389] provide a general solution to establishing media sessions between entities that are behind Network Address Translators (NATs). Loopback sessions that involve one or more endpoints behind NATs can also use these general solutions wherever possible.

If ICE is not supported, then in the case of loopback, the mirroring entity will not send RTP packets and therefore will not automatically create the NAT pinhole in the way that other SIP sessions do. Therefore, if the mirroring entity is behind a NAT, it MUST send some

packets to the identified address/port(s) of the peer, in order to open the NAT pinhole. Using ICE, this would be accomplished with the STUN connectivity check process or through a TURN server connection. If ICE is not supported, either [RFC6263] or Section 10 of ICE [RFC5245] can be followed to open the pinhole and keep the NAT binding alive/refreshed.

Note that for any form of NAT traversal to function, symmetric RTP/RTCP [RFC4961] MUST be used, unless the mirror can control the NAT(s) in its path to create explicit pinholes. In other words, both agents MUST send packets from the source address and port they receive packets on, unless some mechanism is used to avoid that need (e.g., by using the Port Control Protocol).

6. RTP Requirements

A loopback source MUST NOT send multiple source streams on the same 5-tuple, since there is no means for the mirror to indicate which is which in its mirrored RTP packets.

A loopback mirror that is compliant to this specification and accepts media with the loopback type rtp-pkt-loopback loops back 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.

A device that is compliant to this specification and performing the mirroring using the loopback type rtp-media-loopback MUST transmit all received media back to the sender, unless congestion feedback or other lower-layer constraints prevent it from doing so. The incoming media is treated as if it were to be played; for example, the media stream may receive treatment from Packet Loss Concealment (PLC) algorithms. The mirroring entity re-generates all the RTP header fields as it would when transmitting media. The mirroring 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 mirroring entity can choose to preserve the number of frames/packets and the bit rate 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 the 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 requirements prevent the use of the encapsulated RTP payload format.

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 is encapsulated in a separate encapsulating RTP packet; the encapsulated packet would 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 encapsulating RTP packets, the M bit is set to 1 in every fragment except the last packet; otherwise, it is set to 0.

Extension (X) bit: This bit is 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 as that of the encapsulated packet. The initial value of the timestamp SHOULD be random for security reasons (see Section 5.1 of RFC 3550 [RFC3550]).

Synchronization source (SSRC): This field is set as described in RFC 3550 [RFC3550].

The CSRC count (CC) and contributing source (CSRC) fields are used as described in RFC 3550 [RFC3550].

7.1.2. RTP Payload Structure

The outer RTP header of the encapsulating packet is followed by the payload header defined in this section, after any header extension(s). If the received RTP packet has to be looped back in multiple encapsulating packets due to fragmentation, the encapsulating RTP header in each packet is 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]. The RTP payload of the encapsulating RTP packet starts with the payload header defined in this section.

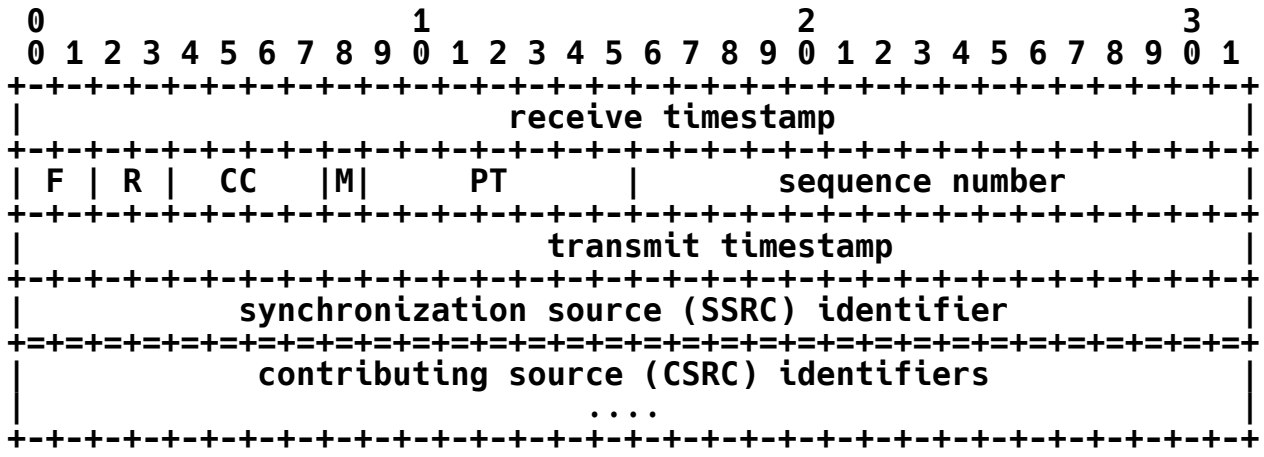

Figure 1. Encapsulating RTP Packet Payload Header

The 12 octets after the receive timestamp are identical to the encapsulated RTP header of the received packet except for the first 2 bits of the first octet. In effect, the received RTP packet is encapsulated by creating a new outer RTP header followed by 4 new bytes of a receive timestamp, followed by the original received RTP header and payload, except that the first two bits of the received RTP header are overwritten as defined here.

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 same clock rate MUST be 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

Possible values are First Fragment (00), Last Fragment (01), No Fragmentation (10), or 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, and all other packets set this field to 11.

7.1.3. Usage of SDP

The payload type number for the encapsulated stream can be negotiated using SDP. There is no static payload type assignment for the encapsulating 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 of concern, or simply due to local policy. When using this payload format, the receiver loops back each received RTP packet payload (not header) in a separate RTP packet.

Because a direct loopback format does not retain the original RTP headers, there will be no indication of the original payload-type sent to the mirror, in looped-back packets. Therefore, the loopback source SHOULD only send one payload type per loopback RTP session if direct mode is used.

7.2.1. Usage of RTP Header Fields

Payload Type (PT): The assignment of an RTP payload type for the encapsulating 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: This bit is set to the value in the received packet.

Extension (X) bit: This bit is 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 per [RFC3550].

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

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

The 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 RTP payload data from the payload portion of the RTP 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 the direct loopback RTP format.

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

8. SRTP Behavior

Secure RTP (SRTP) [RFC3711] MAY be used for loopback sessions. SRTP operates at a lower logical layer than RTP, and thus if both sides negotiate to use SRTP, each side uses its own key and performs encryption/decryption, authentication, etc. Therefore, the loopback function on the mirror occurs after the SRTP packet has been decrypted and authenticated, as a normal cleartext RTP packet without a Master Key Identifier (MKI) or authentication tag; once the

cleartext RTP packet or payload is mirrored -- either at the media-layer, direct packet-layer, or encapsulated packet-layer -- it is encrypted by the mirror using its own key.

In order to provide the same level of protection to both forward and reverse media flows (media to and from the mirror), if SRTP is used it MUST be used in both directions with the same properties.

9. RTCP Requirements

The use of the loopback attribute is intended for the 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 agent that is compliant to this specification SHOULD support RTCP per [RFC3550] and RTCP-XR per RFC 3611 [RFC3611]. Furthermore, if the offerer or answerer chooses to support RTCP-XR, they SHOULD support the RTCP-XR Loss Run Length Encoding (RLE) Report Block, Duplicate RLE Report Block, Statistics Summary Report Block, and VoIP Metrics Report Block per Sections 4.1, 4.2, 4.6, and 4.7 of RFC 3611 [RFC3611]. The offerer and the answerer MAY support other RTCP-XR reporting blocks as defined by RFC 3611 [RFC3611].

10. Congestion Control

All the participants in a media-level 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 [RFC3551].

For packet-level loopback types, the loopback source SHOULD implement congestion control. The mirror will simply reflect back the RTP packets it receives (either in encapsulated or direct modes); therefore, the source needs to control the congestion of both forward and reverse paths by reducing its sending rate to the mirror. This keeps the loopback mirror implementation simpler and provides more flexibility for the source performing a loopback test.

11. Examples

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

11.1. Offer for Specific Media Loopback Type

An agent sends an SDP offer that looks like:

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

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

An answering agent sends an SDP answer that looks like:

```
v=0
o=bob 1234567890 1122334455 IN IP4 host.biloxi.example.com
s=-
c=IN IP4 host.biloxi.example.com
t=0 0
m=audio 49270 RTP/AVP 0
a=loopback:rtp-media-loopback
a=loopback-mirror
a=rtpmap:0 pcmu/8000
```

The answerer agrees to mirror the media from the offerer at the media level.

11.2. Offer for Choice of Media Loopback Type

An agent sends an SDP offer that looks like:

```
v=0
o=alice 2890844526 2890842807 IN IP4 host.atlanta.example.com
s=-
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
a=rtpmap:0 pcmu/8000
a=rtpmap:112 encaprtp/8000
a=rtpmap:113 rtploopback/8000
```

The offerer is offering to source the media and expects the answerer to mirror the RTP stream at either the media or RTP level.

An answering agent sends an SDP answer that looks like:

```
v=0
o=bob 1234567890 1122334455 IN IP4 host.biloxi.example.com
s=-
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
a=rtpmap:0 pcmu/8000
a=rtpmap:112 encaprtp/8000
```

The answerer agrees to mirror the media from the offerer at the packet level using the encapsulated RTP payload format.

11.3. Answerer Rejecting Loopback Media

An agent sends an SDP offer that looks like:

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

The offerer is offering to source the media and expects the answerer to mirror the RTP stream at the media level.

An answering agent sends an SDP answer that looks like:

```
v=0
o=bob 1234567890 1122334455 IN IP4 host.biloxi.example.com
s=-
c=IN IP4 host.biloxi.example.com
t=0 0
m=audio 0 RTP/AVP 0
a=rtpmap:0 pcmu/8000
```

Note in this case that the answerer did not indicate loopback support, although it could have and still used a port number of 0 to indicate that it does not wish to accept that media session.

Alternatively, the answering agent could have simply rejected the entire SDP offer through some higher-layer signaling protocol means (e.g., by rejecting the SIP INVITE request if the SDP offer was in the INVITE).

12. Security Considerations

The security considerations of [RFC3264] and [RFC3550] apply.

Given that media loopback may be automated without the end user's knowledge, the answerer of the media loopback should be aware of denial-of-service attacks. It is RECOMMENDED that session requests for media loopback be authenticated and the frequency of such sessions limited by the answerer.

If the higher-layer signaling protocol were not authenticated, a malicious attacker could create a session between two parties the attacker wishes to target, with each party acting as the loopback mirror to the other, of the rtp-pkt-loopback type. A few RTP packets sent to either party would then infinitely loop among the two, as fast as they could process them, consuming their resources and network bandwidth.

Furthermore, media loopback provides a means of attack indirection, whereby a malicious attacker creates a loopback session as the loopback source and uses the mirror to reflect the attacker's packets against a target -- perhaps a target the attacker could not reach directly, such as one behind a firewall, for example. Or, the attacker could initiate the session as the loopback mirror, in the hopes of making the peer generate media against another target.

If end-user devices such as mobile phones answer loopback requests without authentication and without notifying the end user, then an attacker could cause the battery to drain, and possibly deny the end user normal phone service or cause network data usage fees. This could even occur naturally if a legitimate loopback session does not terminate properly and the end device does not have a timeout mechanism for such.

For the reasons noted above, end-user devices SHOULD provide a means of indicating to the human user that the device is in a loopback session, even if it is an authenticated session. Devices that answer

or generate loopback sessions SHOULD either perform keepalive/refresh tests of the session state through some means or time out the session automatically.

13. Implementation Considerations

The media loopback approach described in this document is a complete solution that would work under all scenarios. However, it is possible that the solution may not be lightweight enough for some implementations. 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 at least support the rtp-pkt-loopback mode for loopback-type, with direct media loopback payload encoding. In addition, for the loopback role, all implementations of an SDP offerer MUST at least be able to act as a loopback source. These requirements are intended to provide a minimal level of interoperability between different implementations.

14. IANA Considerations

14.1. SDP Attributes

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

Contact name:	Kaynam Hedayat
Email address:	kh274@cornell.edu
Telephone number:	+1-617-899-3279
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 for 'loopback' may be one or more of "rtp-pkt-loopback" and "rtp-media-loopback". See Section 4 of RFC 6849 for syntax.

```
Contact name:              Kaynam Hedayat
Email address:             kh274@cornell.edu
Telephone number:          +1-617-899-3279
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:  N/A

Contact name:              Kaynam Hedayat
Email address:             kh274@cornell.edu
Telephone number:          +1-617-899-3279
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:  N/A
```

14.2. Media Types

The IANA has registered the following media types.

14.2.1. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

14.2.2. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given Video Over IP service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

14.2.3. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given real-time text service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

14.2.4. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given real-time application service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

14.2.5. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given VoIP service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

14.2.6. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given Video Over IP service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

14.2.7. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given real-time text service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

14.2.8. 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. This is specified by the loopback source and reflected by the mirror.

Optional parameters: N/A

Encoding considerations: This media type is framed.

Security considerations: See Section 12 of RFC 6849.

Interoperability considerations: N/A

Published specification: RFC 6849.

Applications that use this media type: Applications wishing to monitor and ensure the quality of transport to the edge of a given real-time application service.

Additional information: N/A

Contact: the authors of RFC 6849.

Intended usage: LIMITED USE

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 PAYLOAD working group delegated from the IESG.

15. Acknowledgements

This document's editor would like to thank the original authors of the document: Kaynam Hedayat, Nagarjuna Venna, Paul E. Jones, Arjun Roychowdhury, Chelliah SivaChelvan, and Nathan Stratton. The editor has made fairly insignificant changes in the end. Also, we'd like to thank Magnus Westerlund, Miguel Garcia, Muthu Arul Mozhi Perumal, Jeff Bernstein, Paul Kyzivat, Dave Oran, Flemming Andreasen, Gunnar Hellstrom, Emil Ivov, and Dan Wing for their feedback, comments, and suggestions.

16. References

16.1. Normative References

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

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with 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.

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

[RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

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

[RFC4961] Wing, D., "Symmetric RTP / RTP Control Protocol (RTCP)", BCP 131, RFC 4961, July 2007.

[RFC5234] Crocker, D., Ed., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

16.2. Informative References

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

[RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, April 2010.

[RFC6263] Marjou, X. and A. Sollaud, "Application Mechanism for Keeping Alive the NAT Mappings Associated with RTP / RTP Control Protocol (RTCP) Flows", RFC 6263, June 2011.

Authors' Addresses

Hadriel Kaplan (editor)
Acme Packet
100 Crosby Drive
Bedford, MA 01730
US
EMail: hkaplan@acmepacket.com
URI: http://www.acmepacket.com

Kaynam Hedayat
EXFO
285 Mill Road
Chelmsford, MA 01824
US
EMail: kh274@cornell.edu
URI: http://www.exfo.com/

Nagarjuna Venna
Saperix
c/o DogPatch Labs
One Cambridge Center, 6th Floor
Cambridge, MA 02142
US
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
EMail: paulej@packetizer.com
URI: http://www.cisco.com/

Nathan Stratton
BlinkMind, Inc.
2027 Briarchester Dr.
Katy, TX 77450
US
EMail: nathan@robotics.net
URI: http://www.robotics.net/