RTCWEB
Internet-Draft
Intended status: Standards Track
Expires: August 9, 2012

J. Lennox
Vidyo
P. Witty

A. Romanow
Cisco Systems
February 6, 2012

Real-Time Transport Protocol (RTP) Usage for Telepresence Sessions
draft-lennox-clue-rtp-usage-02

Abstract

This document describes mechanisms and recommended practice for transmitting the media streams of telepresence sessions using the Real-Time Transport Protocol (RTP).

Status of this Memo

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

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

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

This Internet-Draft will expire on August 9, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as

described in the Simplified BSD License.

Table of Contents

1. Introduction

Telepresence systems, of the architecture described by [I-D.ietf-clue-telepresence-use-cases] and [I-D.ietf-clue-telepresence-requirements], will send and receive multiple media streams, where the number of streams in use is potentially large and asymmetric between endpoints, and streams can come and go dynamically. These characteristics lead to a number of architectural design choices which, while still in the scope of potential architectures envisioned by the Real-Time Transport Protocol [RFC3550], must be fairly different than those typically implemented by the current generation of voice or video conferencing systems.

Furthermore, captures, as defined by the CLUE Framework [I-D.ietf-clue-framework], are a somewhat different concept than RTP's concept of media streams, so there is a need to communicate the associations between them.

This document makes recommendations, for this telepresence architecture, about how streams should be encoded and transmitted in RTP, and how their relation to captures should be communicated.

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] and indicate requirement levels for compliant implementations.

3. RTP requirements for CLUE

CLUE will permit a SIP call to include multiple media streams: easily dozens at a time (given, e.g., a continuous presence screen in a multi-point conference), potentially out of a possible pool of hundreds. Furthermore, endpoints will have an asymmetric number of media streams.

Two main backwards compatibility issues exist: firstly, on an initial SIP offer we can not be sure that the far end will support CLUE, and therefore a CLUE endpoint must not offer a selection of RTP sessions which would confuse a CLUEless endpoint. Secondly, there exist many SIP devices in the network through which calls may be routed; even if we know that the far end supports CLUE, re-offering with a larger selection of RTP sessions may fall foul of one of these middle boxes.

We also desire to simplify NAT and firewall traversal by allowing endpoints to deal with only a single static address/port mapping per media type rather than multiple mappings which change dynamically over the duration of the call.

A SIP call in common usage today will typically offer one or two video RTP sessions (one for presentation, one for main video), and one audio session. Each of these RTP sessions will be used to send either zero or one media streams in either direction, with the presence of these streams negotiated in the SDP (offering a particular session as send only, receive only, or send and receive), and through BFCP (for presentation video).

In a CLUE environment this model -- sending zero or one source (in each direction) per RTP session -- doesn't scale as discussed above, and mapping asymmetric numbers of sources to sessions is needlessly complex.

Therefore, telepresence systems SHOULD use a single RTP session per media type, as shown in Figure 1, except where there's a need to give sessions different transport treatment. All sources of the same media type, although from distinct captures, are sent over this single RTP session.

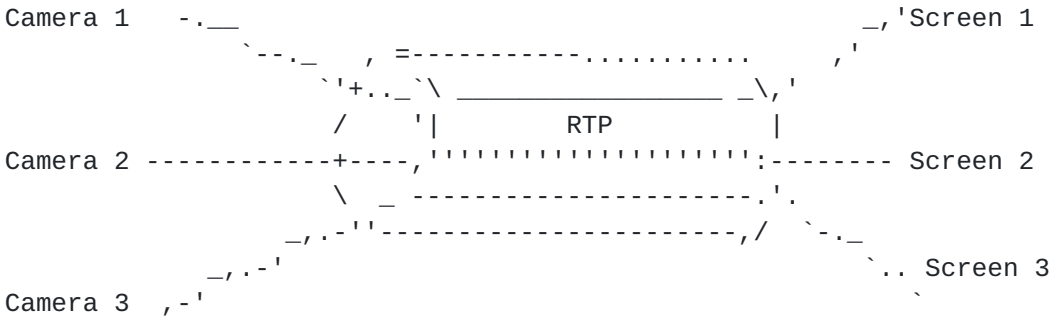

Figure 1: Multiplexing multiple media streams into one RTP session

During call setup, a single RTP session is negotiated for each media type. In SDP, only one media line is negotiated per media and multiple media streams are sent over the same UDP channel negotiated using the SDP media line.

A number of protocol issues involved in multiplexing RTP streams into a single session are discussed in [I-D.westerlund-avtcore-multiplex-architecture] and [I-D.lennox-rtcweb-rtp-media-type-mux]. In the rest of this document we concentrate on examining the mapping of RTP streams to requested

CLUE captures in the specific context of telepresence systems.

The CLUE architecture requires more than simply source multiplexing, as defined by [RFC3550]. The key issue is how a receiver interprets the multiplexed streams it receives, and correlates them with the captures it has requested. In some cases, the CLUE Framework [I-D.ietf-clue-framework]'s concept of the "capture" maps cleanly to the RTP concept of an SSRC, but in many cases it does not.

First we will consider the cases that need to be considered. We will then examine the two most obvious approaches to mapping streams for captures, showing their pros and cons. We then describe a third possible alternative.

4. RTCP requirements for CLUE

When sending media streams, we are also required to send corresponding RTCP information. However, while a unidirectional RTP stream (as identified by a single SSRC) will contain a single stream of media, the associated RTCP stream will include sender information about the stream, but will also include feedback for streams sent in the opposite direction. On a simple point-to-point case, it may be possible to naively forward on RTCP in a similar manner to RTP, but in more complicated use cases where multipoint devices are switching streams to multiple receivers, this simple approach is insufficient.

As an example, receiver report messages are sent with the source SSRC of a single media stream sent in the same direction as the RTCP, but contain within the message zero or more receiver report blocks for streams sent in the other direction. Forwarding on the receiver report packets to the same endpoints which are receiving the media stream tagged with that SSRC will provide no useful information to endpoints receiving the messages, and does not guarantee that the reports will ever reach the origin of the media streams on which they are reporting.

CLUE therefore requires devices to more intelligently deal with received RTCP messages, which will require full packet inspection, including SRTCP decryption. The low rate of RTCP transmission/ reception makes this feasible to do.

5. Multiplexing multiple streams or multiple sessions?

It may not be immediately obvious whether this problem is best described as multiplexing multiple RTP sessions onto a single transport layer, or as multiplexing multiple media streams onto a

single RTP session. Certainly, the different captures represent independent purposes for the media that is sent; however, as any stream may be switched into any of the multiplexed captures, we maintain the requirement that all media streams within a CLUE call must have a unique SSRC -- this is also a requirement for the above use of RTCP.

Because of this, CLUE's use of RTP can best be described as multiplexing multiple streams onto one RTP session, but with additional data about the streams to identify their intended destinations. A solution to perform this multiplexing may also be sufficient to multiplex multiple RTP sessions onto one transport session, but this is not a requirement.

6. Transmission of presentation sources

Most existing videoconferencing systems use separate RTP sessions for main and presentation video sources, distinguished by the SDP content attribute [RFC4796]. The use of the CLUE telepresence framework [I-D.ietf-clue-framework] to describe multiplexed streams can remove this need. However, it could still be useful in some cases to make the distinction between presentation and main video sources at the transport layer. In particular, if different treatment is desired at the transport layer or below (e.g. different VLANs, different QoS characteristics, etc.) for main video vs. presentation, the use of multiple RTP sessions m lines with different transport addresses could would be necessary.

In this case, we need to signal within the CLUE messaging the RTP session in which a requested capture is intended to be received.

7. Use Cases

There are three distinct use cases relevant for telepresence systems: static stream choice, dynamically changing streams chosen from a finite set, and dynamic changing streams chosen from an unbounded set.

Static stream choice:

In this case, the streams sent over the multiplex are constant over the complete session. An example is a triple-camera system to MCU in which left, center and right streams are sent for the duration of the session.

This describes an endpoint to endpoint, endpoint to multipoint

device, and equivalently a transcoding multipoint device to endpoint.

This is illustrated in Figure 2.

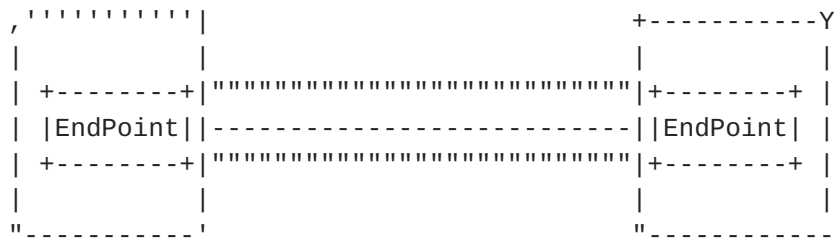

Figure 2: Point to Point Static Streams

Dynamic streams from a finite set:

In this case, the receiver has requested a smaller number of streams than the number of media sources that are available, and expects the sender to switch the sources being sent based on criteria chosen by the sender. (This is called auto-switched in the CLUE Framework [I-D.ietf-clue-framework].)

An example is a triple-camera system to two-screen system, in which the sender needs to switch either LC -> LR, or CR -> LR. (Note in particular, in this example, that the center camera stream could be sent as either the left or the right auto-switched capture.)

This describes an endpoint to endpoint, endpoint to multipoint device, and a transcoding device to endpoint.

This is illustrated in Figure 3.

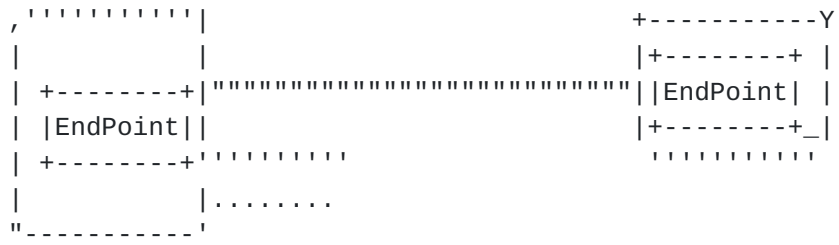

Figure 3: Point to Point Finite Source Streams

Dynamic streams from an unbounded set:

This case describes a switched multipoint device to endpoint, in which the multipoint device can choose to send any streams received from any other endpoints within the conference to the endpoint.

For example, in an MCU to triple-screen system, the MCU could send e.g. LCR of a triple-camera system -> LCR, or CCC of three single-camera endpoints -> LCR.

This is illustrated in Figure 4.

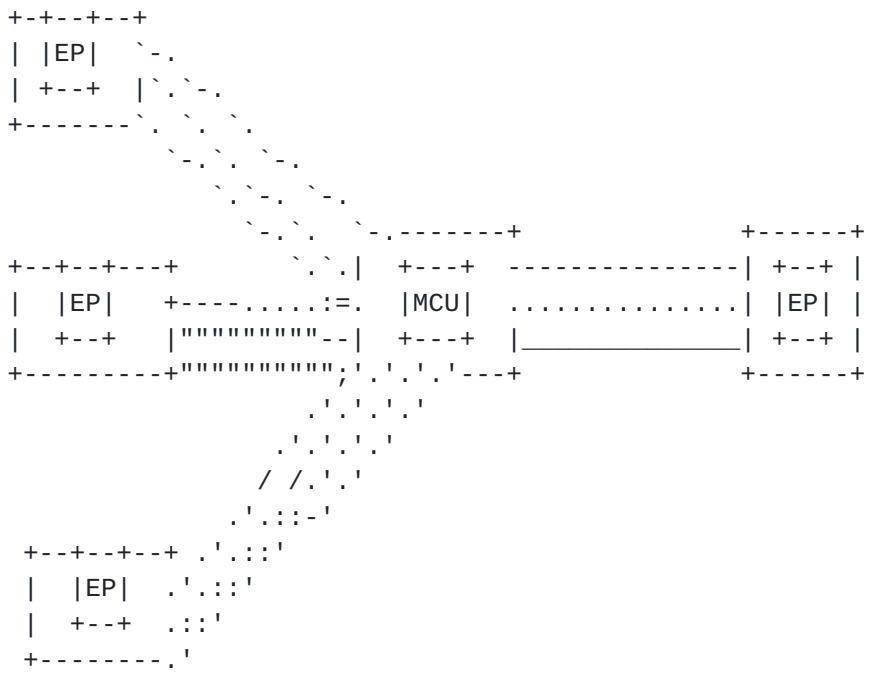

Figure 4: Multipoint Unbounded Streams

Within any of these cases, every stream within the multiplexed session MUST have a unique SSRC. The SSRC is chosen at random [RFC3550] to ensure uniqueness (within the conference), and contains no meaningful information.

Any source may choose to restart a stream at any time, resulting in a new SSRC. For example, a transcoding MCU might, for reasons of load balancing, transfer an encoder onto a different DSP, and throw away all context of the encoding at this state, sending an RTCP BYE message for the old SSRC, and picking a new SSRC for the stream when started on the new DSP.

Because of this possibility of changing the SSRC at any time, all our use cases can be considered as simplifications of the third and most

difficult case, that of dynamic streams from an unbounded set. Thus, this is the primary case we will consider.

8. Mapping streams to requested captures

The goal of any scheme of is to allow the receiver to match the received streams to the requested captures. As discussed in Section 7, during the lifetime of the transmission of one capture, we may see one or multiple media streams which belong to this capture, and during the lifetime of one media stream, it may be assigned to one or more captures.

Demultiplexing of streams is done by SSRC; each stream is known to have a unique SSRC. However, this SSRC contains no information about capture IDs. There are two obvious choices for providing the mapping from SSRC to captures: sending the mapping outside of the media stream, or tagging media packets with the capture ID. There may be other choices, e.g., payload type number, which might be appropriate for multiplexing one audio with one video stream on the same RTP session, but this not relevant for the cases discussed here.

To cope with receivers with limited decoding resources, for example a hardware based telepresence endpoint with a fixed number of decoding modules, each capable of handling only a single stream, it is particularly important to ensure that the number of streams which the transmitter is expecting the receiver to decode never exceeds the maximum number the receiver has requested. In this case the receiver will be forced to drop some of the received streams, causing a poor user experience, and potentially higher bandwidth usage, should it be required to retransmit I-frames.

On a change of stream, such a receiver can be expected to have a one-out, one-in policy, so that the decoder of the stream currently being received on a given capture is stopped before starting the decoder for the stream replacing it. The sender MUST therefore indicate to the receiver which stream will be replaced upon a stream change.

8.1. Sending SSRC to capture ID mapping outside the media stream

Every RTP packet includes an SSRC, which can be used to demultiplex the streams. However, although the SSRC uniquely identifies a stream, it does not indicate which of the requested captures that stream is tied to. If more than one capture is requested, a mapping from SSRC to capture ID is therefore required so that the media receiver can treat each received stream correctly.

As described above, the receiver may need to know in advance of

receiving the media stream how to allocate its decoding resources. Although implementations MAY cache incoming media received before knowing which multiplexed stream it applies to, this is optional, and other implementations may choose to discard media, potentially requiring an expensive state refresh, such as an Full Intra Request (FIR) [RFC5104].

In addition, a receiver will have to store lookup tables of SSRCs to stream IDs/decoders etc. Because of the large SSRC space (32 bits), this will have to be in the form of something like a hash map, and a lookup will have to be performed for every incoming packet, which may prove costly for e.g. MCUs processing large numbers of incoming streams.

Consider the choices for where to put the mapping from SSRC to capture ID. This mapping could be sent in the CLUE messaging. The use of a reliable transport means that it can be sure that the mapping will not be lost, but if this reliability is achieved through retransmission, the time taken for the mapping to reach all receivers (particularly in a very large scale conference, e.g., with thousands of users) could result in very poor switching times, providing a bad user experience.

A second option for sending the mapping is in RTCP, for instance as a new SDES item. This is likely to follow the same path as media, and therefore if the mapping data is sent slightly in advance of the media, it can be expected to be received in advance of the media. However, because RTCP is lossy and, due to its timing rules, cannot always be sent immediately, the mapping may not be received for some time, resulting in the receiver of the media not knowing how to route the received media. A system of acks and retransmissions could mitigate this, but this results in the same high switching latency behaviour as discussed for using CLUE as a transport for the mapping.

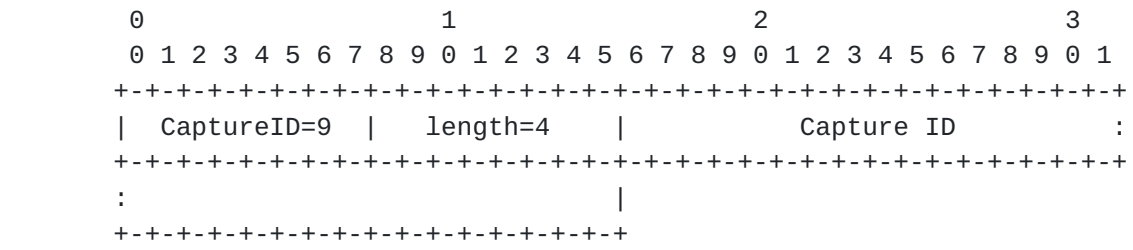

Figure 5: SDES item for encoding of the Capture ID

8.2. Sending capture IDs in the media stream

The second option is to tag each media packet with the capture ID. This means that a receiver immediately knows how to interpret received media, even when an unknown SSRC is seen. As long as the media carries a known capture ID, it can be assumed that this media stream will replace the stream currently being received with that capture ID.

This gives significant advantages to switching latency, as a switch between sources can be achieved without any form of negotiation with the receiver. There is no chance of receiving media without knowing to which switched capture it belongs.

However, the disadvantage in using a capture ID in the stream that it introduces additional processing costs for every media packet, as capture IDs are scoped only within one hop (i.e., within a cascaded conference a capture ID that is used from the source to the first MCU is not meaningful between two MCUs, or between an MCU and a receiver), and so they may need to be added or modified at every stage.

As capture IDs are chosen by the media sender, by offering a particular capture to multiple recipients with the same ID, this requires the sender to only produce one version of the stream (assuming outgoing payload type numbers match). This reduces the cost in the multicast case, although does not necessarily help in the switching case.

An additional issue with putting capture IDs in the RTP packets comes from cases where a non-CLUE aware endpoint is being switched by an MCU to a CLUE endpoint. In this case, we may require up to an additional 12 bytes in the RTP header, which may push a media packet over the MTU. However, as the MTU on either side of the switch may not match, it is possible that this could happen even without adding extra data into the RTP packet. The 12 additional bytes per packet could also be a significant bandwidth increase in the case of very low bandwidth audio codecs.

8.2.1. Multiplex ID shim

As in draft-westerlund-avtcore-transport-multiplexing

8.2.2. RTP header extension

The capture ID could be carried within the RTP header extension field, using [RFC5285]. This is negotiated within the SDP i.e.

a=extmap:1 urn:ietf:params:rtp-hdrex:clue-capture-id

Packets tagged by the sender with the capture ID will then contain a header extension as shown below

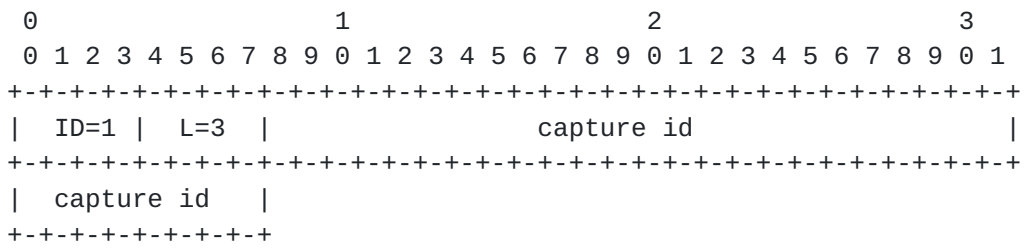

Figure 6: RTP header extension for encoding of the capture ID

To add or modify the capture ID can be an expensive operation, particularly if SRTP is used to authenticate the packet. Modification to the contents of the RTP header requires a reauthentication of the complete packet, and this could prove to be a limiting factor in the throughput of a multipoint device. However, it may be that reauthentication is required in any case due to the nature of SDP. SDP permits the receiver to choose payload types, meaning that a similar option to modify the payload type in the packet header will cause the need to reauthenticate.

8.2.3. Combined approach

The two major flaws of the above methods (high latency switching of SSRC multiplexing, high computational cost on switching nodes) can be mitigated with a combined method. In this, the multiplex ID can be included in packets belonging to the first frame of media (typically an IDR/GDR), but following this only the SSRC is used to demultiplex.

8.2.3.1. Behaviour of receivers

A receiver of a stream should demultiplex on SSRC if it knows the capture ID for the given SSRC, otherwise it should look within the packet for the presence of the stream ID. This has an issue where a stream switches from one capture to a second - for example, in the second use case described in Section 7, where the transmitter chooses to switch the center stream from the receiver's right capture to the left capture, and so the receiver will already know an incorrect mapping from that stream's SSRC to a capture ID.

In this case the receiver should, at the RTP level, detect the presence of the capture ID and update its SSRC to capture ID map.

This could potentially have issues where the demultiplexer has now sent the packet to the wrong physical device - this could be solved by checking for the presence of a capture ID in every packet, but this will have speed implications. If a packet is received where the receiver does not already know the mapping between SSRC and capture ID, and the packet does not contain a capture ID, the receiver may discard it, and MUST request a transmission of the capture ID (see below).

8.2.3.2. Choosing when to send capture IDs

The updated capture ID needs to be known as soon as possible on a switch of SSRCs, as the receiver may be unable to allocate resources to decode the incoming stream, and may throw away the received packets. It can be assumed that the incoming stream is undecodable until the capture ID is received.

In common video codecs (e.g. H.264), decoder refresh frames (either IDR or GDR) also have this property, in that it is impossible to decode any video without first receiving the refresh point. It therefore seems natural to include the capture ID within every packet of an IDR or GDR.

For most audio codecs, where every packet can be decoded independently, there is not such an obvious place to put this information. Placing the capture ID within the first n packets of a stream on a switch is the most simple solution, where n needs to be sufficiently large that it can be expected that at least one packet will have reached the receiver. For example, n=50 on 20ms audio packets will give 1 second of capture IDs, which should give reasonable confidence of arrival.

In the case where a stream is switched between captures, for reasons of coding efficiency, it may be desirable to avoid sending a new IDR frame for this stream, if the receiver's architecture allows the same decoding state to be used for its various captures. In this case, the capture ID could be sent for a small number of frames after the source switches capture, similarly to audio.

8.2.3.3. Requesting Capture ID retransmits

There will, unfortunately, always be cases where a receiver misses the beginning of a stream, and therefore does not have the mapping. One proposal could be to send the capture ID in SDES with every SDES packet; this should ensure that within ~5 seconds of receiving a stream, the capture ID will be received. However, a faster method for requesting the transmission of a capture ID would be preferred.

Again, we look towards the present solution to this problem with video. RFC5104 provides an Full Intra Refresh feedback message, which requests that the encoder provide the stream such that receivers need only the stream after that point. A video receiver without the start of the stream will naturally need to make this request, so by always including the capture ID in refresh frames, we can be sure that the receiver will have all the information it needs to decode the stream (both a refresh point, and a capture ID).

For audio, we can reuse this message. If a receiver receives an audio stream for which it has no SSRC to capture mapping, it should send a FIR message for the received SSRC. Upon receiving this, an audio encoder must then tag outgoing media packets with the capture ID for a short period of time.

Alternately, a new RTCP feedback message could be defined which would explicitly request a refresh of the capture ID mapping.

8.3. Recommendations

We recommend that endpoints MUST support the RTP header extension method of sharing capture IDs, with the extension in every media packet. For low bandwidth situations, this may be considered excessive overhead; in which case endpoints MAY support the combined approach.

This will be advertised in the SDP (in a way yet to be determined); if a receiver advertises support for the combined approach, transmitters which support sending the combined approach SHOULD use it in preference.

9. Security Considerations

The security considerations for multiplexed RTP do not seem to be different than for non-multiplexed RTP.

Capture IDs need to be integrity-protected in secure environments; however, they do not appear to need confidentiality.

10. IANA Considerations

Depending on the decisions, the new RTP header extension element, the new RTCP SDES item, and/or the new AVPF feedback message will need to be registered.

11. References

11.1. Normative References

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

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

11.2. Informative References

[I-D.ietf-clue-framework] Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-03 (work in progress), February 2012.

[I-D.ietf-clue-telepresence-requirements] Romanow, A. and S. Botzko, "Requirements for Telepresence Multi-Streams", draft-ietf-clue-telepresence-requirements-01 (work in progress), October 2011.

[I-D.ietf-clue-telepresence-use-cases] Romanow, A., Botzko, S., Duckworth, M., Even, R., and I. Communications, "Use Cases for Telepresence Multi-streams", draft-ietf-clue-telepresence-use-cases-02 (work in progress), January 2012.

[I-D.lennox-rtcweb-rtp-media-type-mux] Lennox, J. and J. Rosenberg, "Multiplexing Multiple Media Types In a Single Real-Time Transport Protocol (RTP) Session", draft-lennox-rtcweb-rtp-media-type-mux-00 (work in progress), October 2011.

[I-D.westerlund-avtcore-multiplex-architecture] Westerlund, M., Burman, B., and C. Perkins, "RTP Multiplexing Architecture", draft-westerlund-avtcore-multiplex-architecture-00 (work in progress), October 2011.

[RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796, February 2007.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman,

"Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

Authors' Addresses

Jonathan Lennox
Vidyo, Inc.
433 Hackensack Avenue
Seventh Floor
Hackensack, NJ 07601
US

Email: jonathan@vidyo.com

Paul Witty
England
UK

Email: paul.witty@balliol.oxon.org

Allyn Romanow
Cisco Systems
San Jose, CA 95134
USA

Email: allyn@cisco.com