AVTEXT Working Group
Internet-Draft
Intended status: Informational
Expires: June 24, 2012

J. Xia
Huawei

December 22, 2011

Content Splicing for RTP Sessions
draft-ietf-avtext-splicing-for-rtp-03

Abstract

This memo outlines RTP splicing. Splicing is a process that replaces the content of the main multimedia stream with other multimedia content, and delivers the substitutive multimedia content to receiver for a period of time. This memo provides some RTP splicing use cases, then we enumerate a set of requirements and analyze whether an existing RTP level middlebox can meet these requirements, at last we provide concrete guidelines for how the chosen middlebox works to handle RTP splicing.

Status of this Memo

This Internet-Draft is submitted to IETF 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 June 24, 2012.

Table of Contents

1. Introduction

This document outlines how splicing can be used for RTP sessions. Splicing is a process that replaces the content of the main RTP stream with other multimedia content, and delivers the substitutive content to receiver for a period of time. The substitutive content can be provided for example via another RTP stream or local media file storage.

One representative use case for splicing is advertisements insertion, which allows operators to replace the national advertising content with its own regional advertising content prior to delivering the regional advertising content to receiver.

Besides the advertisement insertion use case, there are other use cases to which RTP splicing technology can apply. For example, splicing a recorded video into a video conferencing session, and implementing a playlist server that stitches pieces of video together and so forth.

So far [SCTE30] and [SCTE35] have standardized MPEG2-TS splicing running over cable. The introduction of multimedia splicing into internet requires changes to transport layer, but to date there is no guideline for how to handle content splicing for RTP sessions [RFC3550].

In this document, we first describe a set of requirements of RTP splicing. Then we provide a method about how an intermediary node can be used to process RTP splicing to meet these requirements from the aspects of feasibility, implementation complexity and backward compatibility.

2. Terminology

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

Current RTP Stream

> The RTP stream that the RTP receiver is currently receiving. The content of current RTP stream can be either main content or substitutive content.

Main Content

The multimedia content that are conveyed in main RTP stream. Main content will be replaced by the substitutive content during splicing.

Main RTP Stream

The RTP stream that the Splicer is receiving. The content of main RTP stream can be replaced by substitutive content for a period of time.

Substitutive Content

The multimedia content that replaces the main content during splicing. The substitutive content can for example be contained in an RTP stream from a media sender or fetched from local media file storage.

Substitutive RTP Stream

A RTP stream that may provide substitutive content. Substitutive RTP stream and main RTP stream are two separate streams. If the substitutive content is provided via substitutive RTP stream, the substitutive RTP Stream must pass through Splicer before the substitutive content is delivered to receiver.

Splicing In Point

A virtual point in the RTP stream, suitable for substitutive content entry, that exists in the boundary of two independently decodable frames.

Splicing Out Point

A virtual point in the RTP stream, suitable for substitutive content exit, that exists in the boundary of two independently decodable frames.

Splicer

An intermediary node that inserts substitutive content into main RTP stream. Splicer sends substitutive content to RTP receiver instead of main content during splicing. It is also responsible for processing RTCP traffic between media source and RTP receiver.

3. RTP Splicing Discussion and Requirements

In this document, we assume an intermediary network element, which is referred to as Splicer, to play the key role to handle RTP splicing. A simplified RTP splicing diagram is depicted in Figure 1, in which only one main content flow and one substitutive content flow are given.

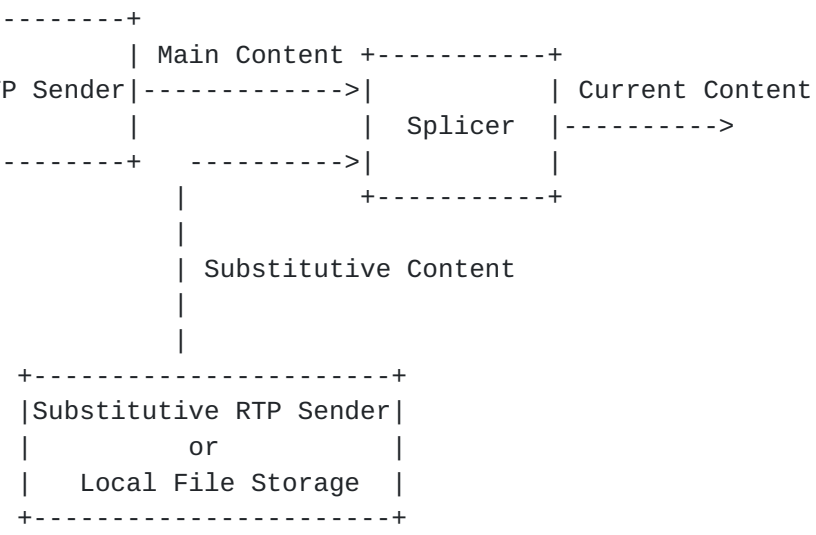

Figure 1: RTP Splicing Architecture

When RTP splicing begins, Splicer stops delivering the main content, instead delivering the substitutive content to RTP receiver for a period of time, and then resumes the main content when splicing ends. The methods how Splicer learns when to start and end the splicing is out of scope for this document. The RTP splicing may happen more than once in case that substitutive content will be dispersedly inserted in multiple time slots during the lifetime of the main RTP stream.

When realizing splicing technology on RTP layer, there are a set of requirements that must be satisfied to at least some degree on Splicer:

REQ-1:

Splicer MUST operate in either unicast or multicast session environment.

REQ-2:

Splicer SHOULD NOT cause perceptible media clipping at the splicing point and adverse impact on the quality of user experience.

REQ-3:

Splicer MUST be backward compatible with RTP/RTCP protocols, and its associated profiles and extensions to those protocols. For example, Splicer MUST be robust to packet loss, network congestion etc.

REQ-4:

Splicer MUST be trusted by media source and receiver, and has the valid security context with media source and RTP receiver respectively.

REQ-5:

Splicer SHOULD allow the media source to learn the performance of the downstream receiver when its content is being passed to RTP receiver.

In a number of deployment scenarios, especially advertisement insertion, there may be one specific requirement. Given that it is unacceptable for advertisers that their advertising content is not delivered to user, this may require RTP splicing to be operated within the following constraint:

If Splicer intends to prevent RTP receiver from identifying and filtering the substitutive content, it SHOULD eliminate the visibility of splicing process on RTP level from RTP receiver point of view.

However, substitutive content and main content are encoded by different encoders and have different parameter sets. In such case, a full media transcoding must be done on Splicer to ensure the completely invisible impact on RTP receiver, but this may be prohibitively expensive and complex. As a trade-off, it is RECOMMENDED to minimize the splicing visibility on RTP receiver, i.e., maintaining RTP header parameters consistent but leaving the RTP payload untranscoded. If one wants to realize complete invisibility, the cost of transcoding must be taken into account.

Henceforth, we refer to the minimum and complete invisibility requirement as User Invisibility Requirement.

To improve the versatility of existing implementations and better interoperability, it is RECOMMENDED to use existing tools in RTP/RTCP protocol family to realize RTP splicing without any protocol extension unless the existing tools are incompetent for splicing.

4. Recommended Solution for RTP Splicing

Given that Splicer is an intermediary node exists between the main media source and the RTP receiver and splicing is not a very complicated processing, there are some chance that any existing RTP-level middlebox may has the incidental capability to meet the requirements described in previous section.

Since Splicer needs to select substitutive content or main content as the input content at one point of time, an RTP mixer seems to have such capability to do this under its own SSRC. Moreover, mixer includes the CSRC list in outgoing packets to indicate the source(s) of content, this facilitates the system debugging. From this point of view, an RTP mixer may have some chance to be Splicer. In next four subsections (from subsection 4.1 to subsection 4.4), we start analyzing how an RTP mixer handles RTP splicing and how it satisfies the general requirements listed in section 3.

In subsection 4.5, we specially consider the special requirement 6 (i.e., User Invisibility Requirement) since it needs to mask any RTP splicing clue on user (e.g, CSRC list must not be included in outgoing packets to prevent user from identifying the difference between main RTP stream and substitutive RTP stream) when mixer is used.

4.1. RTP Processing in RTP Mixer

Once mixer has learnt when to do splicing, it must get ready for the coming splicing in advance, e.g., fetches the substitutive content either from local media file storage or via substitutive RTP stream earlier than splicing in point. If the substitutive content comes from local media file storage, mixer should leave the CSRC list blank in the output stream.

Even if splicing does not begin, mixer still needs to receive the main RTP stream, terminate it and generate a media stream as defined in RFC3550. Using the main RTP packets, mixer generates the current media stream with its own SSRC, sequence number space and timing

model. Moreover, mixer inserts the SSRC of main RTP stream into CSRC list in the current media stream.

When splicing begins, mixer chooses the substitutive RTP stream as input stream at splicing in point, extracts the payload data (i.e., substitutive content), encodes substitutive content and outputs it instead of main content in the current media stream. Moreover, mixer inserts the SSRC of substitutive RTP stream into CSRC list in the current media stream.

When splicing ends, mixer retrieves the main RTP stream as input stream at splicing out point, extracts the payload data (i.e., main content), encodes main content and outputs it instead of substitutive content in the current media stream. Moreover, mixer inserts the SSRC of main RTP stream into CSRC list in the current media stream.

The whole RTP splicing procedure is perhaps best explained by a pseudo code example:

```
if (splicing begins) {
   the substitutive RTP stream is terminated on mixer and
   substitutive content is encoded by mixer with its own SSRC
   identifier;

   the sequence numbers of the current RTP packets which contain
   substitutive content are allocated by mixer and maintain
   consistent with the sequence numbers of previous current RTP
   packets, until the splicing end;

   the timestamp of the current RTP packet increments linearly;

   the CSRC list of the current RTP packet indicates SSRC of
   substitutive RTP stream;
}

else {
   the main RTP stream is terminated on mixer and main content is
   encoded by mixer with its own SSRC identifier;

   the sequence numbers of the current RTP packets which contain main
   content are allocated by mixer and maintain consistent with the
   sequence numbers of previous current RTP packets, until the
   splicing begins;

   the timestamp of the current RTP packets increments linearly;

   the CSRC list the current RTP indicates SSRC of main RTP stream;
}
```

Splicing may occur more than one time during the lifetime of main RTP stream, this means mixer needs to output main content and substitutive content in turn with its own SSRC identifier. From user point of view, the only source of the current stream is mixer wherever the content comes from.

Note that, the substitutive content should be outputted in the range of splicing duration. Any gap or overlap between main RTP stream and substitutive RTP stream may induce media clipping at splicing point. More details about preventing media clipping are introduced in section 4.3.

4.2. RTCP Processing in RTP Mixer

By monitoring available bandwidth and buffer levels and by computing network metrics such as packet loss, network jitter, and delay, RTP receiver can learn the situation on it and can communicate this information to media source via RTCP reception reports.

According to the description in section 7.3 of [RFC3550], mixer divides RTCP flow between media source and receiver into two separate RTCP loops, media source probably has no idea about the situation on receiver. Hence, mixer may use some mechanisms, allowing media source to at least some degree to have some knowledge of the situation on receiver when its content is being passed to receiver.

Because splicing is a processing that mixer selects one media stream from multiple streams rather than mixing them, the number of output RTP packets containing substitutive content is equal to the number of input substitutive RTP packets (from substitutive RTP stream) during splicing, the mixer does not need to modify loss packet fields in receiver report blocks unless the reporting intervals spans the splicing point. But mixer needs to change the SSRC field in report block to the SSRC identifier of original media source and rewrite the extended highest sequence number field to the corresponding original extended highest sequence number before forwarding the RTCP reception reports to original media source.

When a RTCP receiver report spans the splicing point, it reflects the characteristics of the combination of main RTP packets and substitutive RTP packets, in which case, mixer needs to divide the receiver report into two separated receiver reports and send them to their original media sources respectively. For each separated receiver report, mixer also needs to make the corresponding changes to the packet loss fields in report block besides the SSRC field and the extended highest sequence number field.

Based on above RTCP operating mechanism, the media source will see

the reception quality of its stream received by mixer, and the reception quality of spliced stream received by RTP receiver.

If the substitutive content comes from local media file storage (i.e., mixer can be regarded as the substitutive media source), the reception reports should be terminated on mixer without any further processing.

For the media source whose content is terminated on mixer and is not being passed to receiver, mixer must act as a receiver and send reception reports to the media source.

4.3. Media Clipping Considerations

This section provides informative guideline about how media clipping may shape and how mixer deal with the media clipping.

If the time slot for substitutive RTP stream mismatches (shorter or longer than) the duration of the reserved main RTP stream for replacing, the media clipping may occur at the splicing point which usually is the joint between two independently decodable frames.

At the splicing in point, mixer can fill the substitutive content up receiver's buffer with several seconds earlier than the presentation time of substitutive content so that smooth playback can be achieved without pauses or stuttering on RTP receiver.

Compared to buffering method used at splicing in point, things become somewhat complex at splicing out point. The case that insertion duration is shorter than the reserved gap time may cause a little playback latency of main RTP stream on RTP receiver, but not adversely impact the quality of user experience. However, in case that insertion duration is longer than the reserved gap duration, there exists an overlap of the substitutive RTP stream and the main RTP stream at splicing out point. In such case, mixer may take a ungracefule action, terminating the splicing and switching back to main RTP stream even if this may cause media stuttering on receiver

Another reason to cause media clipping is synchronization delay at splicing point if RTP receiver needs to synchronize multiple current streams for playback. How to address this issue is discussed in detail in [RFC6051], which provides three feasible approaches to reduce synchronization delay.

4.4. Congestion Control Considerations

Provided that the substitutive content has somewhat different characteristics to the main content it replaces (e.g., the more

dynamic content, the higher bandwidth occupation), or substitutive content may be encoded with different codec and has different encoding bitrate, some challenge raise to network capacity and receiver buffer size. A more dynamic content or a higher encoding bitrate stream might overload the network and possibly exceed the receiver's media consumption rate, which might flood receiver's buffer and eventually result in a buffer overflow. Either network overload or buffer overflow would induce network congestion and congestion-caused packet loss.

To be robust to network congestion and packet loss, mixer must continuously monitor the network situation by means of a variety of manners:

1. RTCP receiver reports indicate packet loss [RFC3550].

2. RTCP NACKs for lost packet recovery [RFC4585].

3. RTCP ECN Feedback information [I-D.ietf-avtcore-ecn-for-rtp].

Upon detection of above three types of RTCP reports during splicing, mixer will treat them with three different manners as following:

1. If mixer receives the RTCP receiver reports with packet loss indication, it will process them as the description given in section 7.3 of [RFC3550].

2. If mixer receives the RTCP NACK packets defined in [RFC4585] from RTP receiver for packet loss recovery, it first identifies the content category of lost packets to which the NACK corresponds. Then, mixer will generate new RTCP NACK for the lost packets with its own SSRC, and make corresponding changes to their sequence numbers to match original, pre-spliced, packets. If the lost substitutive content comes from local media file storage, mixer acting as substitutive media source will directly fetch the lost substitutive content and retransmit it to RTP receiver.

   It is somewhat complex that the lost packets requested in a single RTCP NACK message not only contain the main content but also the substitutive content. To address this, mixer must divide the RTCP NACK packet into two separate RTCP NACK packets: one requests for the lost main content, and another requests for the lost substitutive content.

3. In [I-D.ietf-avtcore-ecn-for-rtp], two RTCP extensions are defined for ECN feedback: RTP/AVPF transport layer ECN feedback packet for urgent ECN information, and RTCP XR ECN summary report block for regular reporting of the ECN marking information.

If an ECN-aware mixer receives any RTCP ECN feedback (i.e., RTCP ECN feedback packets or RTCP XR summary reports) from RTP receiver, it must operates as description given in section 8.4 of [I-D.ietf-avtcore-ecn-for-rtp], terminating the RTCP ECN feedback packets from downstream receivers, and driving congestion control loop and bitrate adaptation between itself and downstream receiver as if it were the media source. In addition, an ECN-aware RTP mixer must generate RTCP ECN feedback relating to the input RTP streams it terminates, and driving congestion control loop and bitrate adaptation between itself and upstream sender as if it were the RTP sender.

Once mixer learns that congestion is being experienced on its downstream link by means of above three detection mechanisms, it should adapt the bitrate of output stream in response to network congestion. The bitrate adaptation may be determined by a TCP-friendly bitrate adaptation algorithm specified in [RFC5348], or by a DCCP congestion control algorithms defined in [RFC5762].

In practice, during splicing, the real reason to cause congestion usually is the different characteristic of substitutive RTP stream (more dynamic content or higher encoding bitrate) with main RTP stream, and that stream transcoding or thinning on mixer is very inefficient and difficult operation. Therefore, a means that enables substitutive media source to limit the media bitrate it is currently generating even in the absence of congestion on the path between itself and mixer is desirable. The TMMBR message defined in [RFC5104] provides an effective method. When mixer detects congestion on its downstream link during splicing, it uses TMMBR to request substitutive media source to reduce the media bitrate to a value that is in compliance with congestion control principles for the slowest link. Upon reception of TMMBR, substitutive media source applies its congestion control algorithm and responds Temporary Maximum Media Stream Bit Rate Notification (TMMBN) to mixer.

If the substitutive content comes from local media file storage, mixer must directly reduce the substitutive media bitrate as the substitutive media source when it detects any congestion on its downstream link during splicing.

From above analysis, to reduce the risk of congestion and remain the bandwidth consumption stable over time, the substitutive RTP stream is RECOMMENDED to be encoded at an appropriate bitrate to match that of main RTP stream. If the substitutive RTP stream comes from substitutive media source, the source had better has some knowledge about the media encoding bitrate of main content in advance. How it knows that is out of scope in this draft.

4.5. Processing Splicing in User Invisibility Case

Compared to above user visibility case, the primary difference in this case is mixer MUST NOT include CSRC list in outgoing packets (i.e., CSRC count field is set to zero and CSRC list fields are absent).

Therefore, due to the absence of CRSC list in current RTP stream, RTP receiver only initiates SDES, BYE and APP packets to mixer without any knowledge of main media source and substitutive media source. This creates a danger that loops involving those sources could not be detected.

5. Implementation Considerations

When mixer is used to handle RTP splicing, RTP receiver does not need any RTP/RTCP extension for splicing. As a trade-off, additional overhead could be induced on mixer which uses its own sequence number space and timing model. So mixer will rewrite RTP sequence number and timestamp whatever splicing is active or not, and generate RTCP flows for both sides. In case mixer serves multiple main RTP streams simultaneously, this may lead to more overhead on mixer.

In addition, there is a potential issue with loop detection, which would be problematic if User Invisibility Requirement is required.

6. Security Considerations

If any payload internal security mechanisms (e.g., SSH, SSL etc) are used, only media source and RTP receiver can learn the security keying material generated by such internal security mechanism, any middlebox (e.g., mixer) between media source and RTP receiver can't get such keying material. Only when regular transport security mechanisms (e.g., SRTP, IPSec, etc) are used, mixer will process the packets passing through it.

The security considerations of the RTP specification [RFC3550], the Extended RTP profile for RTCP-Based Feedback [RFC4585], and the Secure Real-time Transport Protocol [RFC3711] apply. Mixer must be trusted by main media source and insertion media source, and must be included in the security context.

7. IANA Considerations

No IANA actions are required.

8. Acknowledgments

The following individuals have reviewed the earlier versions of this specification and provided very valuable comments: Colin Perkins, Magnus Westerlund, Roni Even, Tom Van Caenegem, Joerg Ott, David R Oran, Cullen Jennings, Ali C Begen, and Ning Zong.

9. Change Log

9.1. draft-xia-avtext-splicing-for-rtp-01

The following are the major changes compared to previous version 00:

- Use mixer to handle both user visible and invisible splicing.
- Add one subsection to describe media clipping considerations.
- Add one subsection to describe congestion control considerations.

9.2. draft-xia-avtext-splicing-for-rtp-00

The following are the major changes compared to previous AVT I-D version 00:

- Change primary RTP stream to main RTP stream, add current RTP stream as the streaming received by RTP receiver.
- Eliminate the ambiguity of inserted content with substitutive content which replaces the main content rather than pause it.
- Clarify the signaling requirements.
- Delete the description on Mixer and MCU in section 4, mainly focus on the direction whether a Translator can act as a Splicer.
- Add section 5 to describe the exact guidance on how an RTP Translator is used to handle splicing.
- Modify the security considerations section and add acknowledges section.

10. References

10.1. Normative References

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

[RFC2250] Hoffman, D., Fernando, G., Goyal, V., and M. Civanlar, "RTP Payload Format for MPEG1/MPEG2 Video", RFC 2250, January 1998.

[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.

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

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006.

[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.

[RFC5117] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.

[RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, November 2010.

[I-D.ietf-avtcore-ecn-for-rtp] Westerlund, M., "Explicit Congestion Notification (ECN) for RTP over UDP", draft-ietf-avtcore-ecn-for-rtp-05 (work in progress), October 2011.

10.2. Informative References

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, September 2008.

[RFC5760] Ott, J., Chesterfield, J., and E. Schooler, "RTP Control

Protocol (RTCP) Extensions for Single-Source Multicast Sessions with Unicast Feedback", RFC 5760, February 2010.

[RFC5762] Perkins, C., "RTP and the Datagram Congestion Control Protocol (DCCP)", RFC 5762, April 2010.

[SCTE30] Society of Cable Telecommunications Engineers (SCTE), "Digital Program Insertion Splicing API", 2001.

[SCTE35] Society of Cable Telecommunications Engineers (SCTE), "Digital Program Insertion Cueing Message for Cable", 2004.

[H.323] ITU-T Recommendation H.323, "Packet-based multimedia communications systems", June 2006.

Author's Address

Jinwei Xia
Huawei
Software No.101
Nanjing, Yuhuatai District 210012
China

Phone: +86-025-86622310
Email: xiajinwei@huawei.com