Content Splicing for RTP Sessions
draft-ietf-avtext-splicing-for-rtp-09

Abstract

   Content splicing is a process that replaces the content of a main
   multimedia stream with other multimedia content, and delivers the
   substitutive multimedia content to the receivers for a period of
   time. Splicing is commonly used for local advertisement insertion by
   cable operators, replacing a national advertisement content with a
   local advertisement.

   This memo describes some use cases for content splicing and a set of
   requirements for splicing content delivered by RTP. It provides
   concrete guidelines for how an RTP mixer can be used to handle
   content splicing.

Status of this Memo

Copyright Notice

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**

   This document outlines how content splicing can be used in RTP
   sessions. Splicing, in general, is a process where part of a
   multimedia content is replaced with other multimedia content, and
   delivered to the receivers for a period of time. The substitutive
   content can be provided for example via another stream or via local
   media file storage. One representative use case for splicing is
   local advertisement insertion, allowing content providers to replace
   the national advertising content with its own regional advertising
   content prior to delivering the regional advertising content to the
   receivers. Besides the advertisement insertion use case, there are
   other use cases in which splicing technology can be applied. For
   example, splicing a recorded video into a video conferencing session,
   or implementing a playlist server that stitches pieces of video
   together.

   Content splicing is a well-defined operation in MPEG-based cable TV
   systems. Indeed, the Society for Cable Telecommunications Engineers
   (SCTE) has created two standards, [SCTE30] and [SCTE35], to
   standardize MPEG2-TS splicing procedure. SCTE 30 creates a
   standardized method for communication between advertisements server
   and splicer, and SCTE 35 supports splicing of MPEG2 transport
   streams.

   When using multimedia splicing into the internet, the media may be
   transported by RTP. In this case the original media content and
   substitutive media content will use the same time period, but may
   contain different numbers of RTP packets due to different media
   codecs and entropy coding. This mismatch may require some
   adjustments of the RTP header sequence number to maintain
   consistency. [RFC3550] provides the tools to enabled seamless
   content splicing in RTP session, but to date there has been no clear
   guidelines on how to use these tools.

   This memo outlines the requirements for content splicing in RTP
   sessions and describes how an RTP mixer can be used to meet these
   requirements.

2. **System Model and Terminology**

   In this document, an intermediary network element, the Splicer
   handles RTP splicing. The Splicer can receive main content and
   substitutive content simultaneously, but will send one of them at one
   point of time.

   When RTP splicing begins, the splicer sends the substitutive content

   to the RTP receiver instead of the main content for a period of time.
   When RTP splicing ends, the splicer switches back sending the main
   content to the RTP receiver.

   A simplified RTP splicing diagram is depicted in Figure 1, in which
   only one main content flow and one substitutive content flow are
   given. Actually, the splicer can handle multiple splicing for
   multiple RTP sessions simultaneously. RTP splicing may happen more
   than once in multiple time slots during the lifetime of the main RTP
   stream. The methods how splicer learns when to start and end the
   splicing is out of scope for this document.

```
     +----------------+
     |                |  Main Content +-----------+
     |    Main RTP    |-------------->|           |  Output Content
     |    Content     |              |  Splicer   |--------------->
     +----------------+   ---------->|           |
                      |              +-----------+
                      |
                      |  Substitutive Content
                      |
                      |
           +-----------------------+
           |    Substitutive RTP   |
           |         Content       |
           |            or         |
           |    Local File Storage |
           +-----------------------+
```
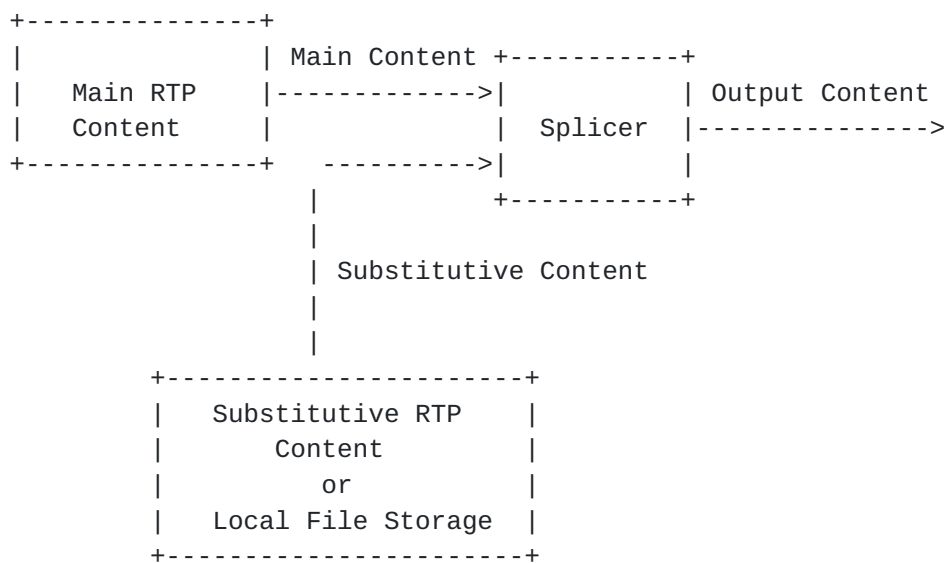
              Figure 1: RTP Splicing Architecture

   This document uses the following terminologies.

   Output RTP Stream

      The RTP stream that the RTP receiver is currently receiving. The
      content of output 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.

Main RTP Sender

   The sender of RTP packets carrying the main RTP stream.

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 with new content that will replace the content in the
   main RTP stream. 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 the splicer before the substitutive content is delivered
   to receiver.

Substitutive RTP Sender

   The sender of RTP packets carrying the substitutive RTP stream.

Splicing In Point

   A virtual point in the RTP stream, suitable for substitutive
   content entry, typically in the boundary between two independently
   decodable frames.

Splicing Out Point

   A virtual point in the RTP stream, suitable for substitutive
   content exist, typically in the boundary between two independently
   decodable frames.

Splicer

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

the RTP receiver.

3. Requirements for RTP Splicing

In order to allow seamless content splicing at the RTP layer, the
following requirements must be met. Meeting these will also allow,
but not require, seamless content splicing at layers above RTP.

REQ-1:

   The splicer should be agnostic about the network and transport
   layer protocols used to deliver the RTP streams.

REQ-2:

   The splicing operation at the RTP layer must allow splicing at any
   point required by the media content, and must not constrain when
   splicing in or splicing out operations can take place.

REQ-3:

   Splicing of RTP content must be backward compatible with the RTP/
   RTCP protocol, associated profiles, payload formats, and
   extensions.

REQ-4:

   The splicer will modify the content of RTP packets, and break the
   end-to-end security, e.g., breaking data integrity and source
   authentication. If the Splicer is designated to insert
   substitutive content, it must be trusted, i.e., be in the security
   context(s) as the main RTP sender, the substitutive RTP sender,
   and the receivers. If encryption is employed, the splicer must be
   able to decrypt the inbound RTP packets and re-encrypt the
   outbound RTP packets after splicing.

REQ-5:

   The splicer should rewrite as necessary and forward RTCP messages
   (e.g., including packet loss, jitter, etc.) sent from downstream
   receiver to the main RTP sender or the substitutive RTP sender,
   and thus allow the main RTP sender or substitutive RTP sender to
   learn the performance of the downstream receiver when its content
   is being passed to RTP receiver. In addition, the splicer should
   rewrite RTCP messages from the main RTP sender or substitutive RTP
   sender to the receiver.

REQ-6:

   The splicer must not affect other RTP sessions running between the
   RTP sender and the RTP receiver, and must be transparent for the
   RTP sessions it does not splice.

REQ-7:

   The splicer should be able to modify the RTP stream across a
   splicing in or splicing out point such that the splicing point is
   not easy to be detected in the RTP stream. For the advertisement
   insertion use case, it is important to make it difficult for the
   receiver to detect where an advertisement insertion is starting or
   ending from the RTP packets. Ensuring the splicing point is not
   visible in the media content may be easy with some codecs, but
   extremely difficult with others; in the worst case, the splicer
   may need to perform full media transcoding if it has to hide the
   splicing point in the media content. This memo only focuses on
   making the splicing invisible at the RTP layer. How (or if) the
   splicing is made invisible in the media stream is outside the
   scope of this memo.

4. Content Splicing for RTP sessions

   The RTP specification [RFC3550] defines two types of middlebox: RTP
   translators and RTP mixers. Splicing is best viewed as a mixing
   operation. The splicer generates a new RTP stream that is a mix of
   the main RTP stream and the substitutive RTP stream. An RTP mixer is
   therefore an appropriate model for a content splicer. In next four
   subsections (from subsection 4.1 to subsection 4.4), the document
   analyzes how the mixer handles RTP splicing and how it satisfies the
   general requirements listed in section 3. In subsection 4.5, the
   document looks at REQ-7 in order to hide the fact that splicing take
   place.

4.1. RTP Processing in RTP Mixer

   A splicer could be implemented as a mixer that receives the main RTP
   stream and the substitutive content (possibly via a substitutive RTP
   stream), and sends a single output RTP stream to the receiver(s).
   That output RTP stream will contain either the main content or the
   substitutive content. The output RTP stream will come from the
   mixer, and will have the SSRC of the mixer rather than the main RTP
   sender or the substitutive RTP sender.

   The mixer uses its own SSRC, sequence number space and timing model

when generating the output stream. Moreover, the mixer may insert
the SSRC of main RTP stream into CSRC list in the output media
stream.

At the splicing in point, when the substitutive content becomes
active, the mixer chooses the substitutive RTP stream as input stream
at splicing in point, and extracts the payload data (i.e.,
substitutive content). If the substitutive content comes from local
media file storage, the mixer directly fetches the substitutive
content. After that, the mixer encapsulates substitutive content
instead of main content as the payload of the output media stream,
and then sends the output RTP media stream to receiver. The mixer
may insert the SSRC of substitutive RTP stream into CSRC list in the
output media stream. If the substitutive content comes from local
media file storage, the mixer should leave the CSRC list blank.

At the splicing out point, when the substitutive content ends, the
mixer retrieves the main RTP stream as input stream at splicing out
point, and extracts the payload data (i.e., main content). After
that, the mixer encapsulates main content instead of substitutive
content as the payload of the output media stream, and then sends the
output media stream to the receivers. Moreover, the mixer may insert
the SSRC of main RTP stream into CSRC list in the output media stream
as before.

Note that if the content is too large to fit into RTP packets sent to
RTP receiver, the mixer needs to transcode or perform application-
layer fragmentation. Usually the mixer is deployed as part of a
managed system and MTU will be carefully managed by this system.
This document does not raise any new MTU related issues compared to a
standard mixer described in [RFC3550].

Splicing may occur more than once during the lifetime of main RTP
stream, this means the mixer needs to send main content and
substitutive content in turn with its own SSRC identifier. From
receiver point of view, the only source of the output stream is the
mixer regardless of where the content is coming from.

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 network performance and communicate this to
the RTP sender via RTCP reception reports.

According to the description in section 7.3 of [RFC3550], the mixer
splits the RTCP flow between sender and receiver into two separate
RTCP loops, RTP sender has no idea about the situation on the

receiver. But splicing is a processing that the mixer selects one
media stream from multiple streams rather than mixing them, so the
mixer can leave the SSRC identifier in the RTCP report intact (i.e.,
the SSRC of downstream receiver), this enables the main RTP sender or
the substitutive RTP sender to learn the situation on the receiver.

If the RTCP report corresponds to a time interval that is entirely
main content or entirely substitutive content, 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, in the same manner, the number of output RTP packets
containing main content is equal to the number of input main RTP
packets (from main RTP stream) during non-splicing unless the mixer
fragment the input RTP packets. This means that the mixer does not
need to modify the loss packet fields in reception report blocks in
RTCP reports. But if the mixer fragments the input RTP packets, it
may need to modify the loss packet fields to compensate for the
fragmentation. Whether the input RTP packets are fragmented or not,
the mixer still needs to change the SSRC field in report block to the
SSRC identifier of the main RTP sender or the substitutive RTP
sender, and rewrite the extended highest sequence number field to the
corresponding original extended highest sequence number before
forwarding the RTCP report to the main RTP sender or the substitutive
RTP sender.

If the RTCP report spans the splicing in point or the splicing out
point, it reflects the characteristics of the combination of main RTP
packets and substitutive RTP packets. In this case, the mixer needs
to divide the RTCP report into two separate RTCP reports and send
them to their original RTP senders respectively. For each RTCP
report, the 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.

If the mixer receives an RTCP extended report (XR) block, it should
rewrite the XR report block in a similar way to the reception report
block in the RTCP report.

Besides forwarding the RTCP reports sent from RTP receiver, the mixer
can also generate its own RTCP reports to inform the main RTP sender
or the substitutive RTP sender of the reception quality of the
content reaches the mixer when the content is not sent to the RTP
receiver. These RTCP reports use the SSRC of the mixer. If the
substitutive content comes from local media file storage, the mixer
does not need to generate RTCP reports for the substitutive stream.

Based on above RTCP operating mechanism, the RTP sender whose content
is being passed to receiver will see the reception quality of its

stream as received by the mixer, and the reception quality of spliced
stream as received by the receiver. The RTP sender whose content is
not being passed to receiver will only see the reception quality of
its stream as received by the mixer.

The mixer must forward RTCP SDES and BYE packets from the receiver to
the sender, and may forward them in inverse direction as defined in
section 7.3 of [RFC3550].

Once the mixer receives an RTP/AVPF [RFC4585] transport layer
feedback packet, it must handle it carefully as the feedback packet
may contain the information of the content that come from different
RTP senders. In this case the mixer needs to divide the feedback
packet into two separate feedback packets and process the information
in the feedback control information (FCI) in the two feedback
packets, just as the RTCP report process described above.

If the substitutive content comes from local media file storage
(i.e., the mixer can be regarded as the substitutive RTP sender), any
RTCP packets received from downstream relate to the substitutive
content must be terminated on the mixer without any further
processing.

4.3. Media Clipping Considerations

This section provides informative guideline about how media clipping
is shaped and how the mixer deal with the media clipping.

If the time slot for substitutive content mismatches (is shorter or
longer than) the duration of the main content to be replaced, then
media clipping may occur at the splicing point.

If the substitutive content has shorter duration from the main
content, then there will be a gap in the output RTP stream. The RTP
sequence number will be contiguous across this gap, but there will be
an unexpected jump in the RTP timestamp. This gap will cause the
receiver to have nothing to play. This is unavoidable, unless the
mixer adjusts the splice in or splice out point to compensate,
sending more of the main RTP stream in place of the shorter
substitutive stream, or unless the mixer can vary the length of the
substitutive content. It is the responsibility of the higher layer
protocols to ensure that the substitutive content is of the same
duration as the main content to be replaced.

If the insertion duration is longer than the reserved gap duration,
there will be an overlap between the substitutive RTP stream and the
main RTP stream at splicing out point. One straightforward approach
is that the mixer takes an ungraceful action, terminating the

splicing and switching back to main RTP stream even if this may cause
media stuttering on receiver. Alternatively, the mixer may transcode
the substitutive content to play at a faster rate than normal, to
adjust it to the length of the gap in the main content, and generate
a new RTP stream for the transcoded content. This is a complex
operation, and very specific to the content and media codec used.

4.4. Congestion Control Considerations

If the substitutive content has somewhat different characteristics
from the main content it replaces, or if the substitutive content is
encoded with a different codec or has different encoding bitrate, it
might overload the network and might cause network congestion on the
path between the mixer and the RTP receiver(s) that would not have
been caused by the main content.

To be robust to network congestion and packet loss, a mixer that is
performing splicing must continuously monitor the status of
downstream network by monitoring any of the following RTCP reports
that are used:

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].

Once the mixer detects congestion on its downstream link, it will
treat these reports as follows:

1. If the mixer receives the RTCP receiver reports with packet loss
    indication, it will forward the reports to the substitutive RTP
    sender or the main RTP sender as described in section 4.2.

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, the 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,
    the mixer acting as substitutive RTP sender will directly fetch
    the lost substitutive content and retransmit it to RTP receiver.
    The mixer may buffer the sent RTP packets and do the
    retransmission.

    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, the 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. If an ECN-aware mixer receives RTCP ECN feedbacks (RTCP ECN
feedback packets or RTCP XR summary reports) defined in
[I-D.ietf-avtcore-ecn-for-rtp] from the RTP receiver, it must
process them in a similar way to the RTP/AVPF feedback packet or
RTCP XR process described in section 4.2 of this memo.

These three methods require the mixer to run a congestion control
loop and bitrate adaptation between itself and RTP receiver. The
mixer can thin or transcode the main RTP stream or the substitutive
RTP stream, but such operations are very inefficient and difficult,
and bring undesirable delay. Fortunately in this memo, the mixer
acting as splicer can rewrite the RTCP packets sent from the RTP
receiver and forward them to the RTP sender, thus letting the RTP
sender knows that congestion is being experienced on the path between
the mixer and the RTP receiver. Then, the RTP sender applies its
congestion control algorithm and reduces the media bitrate to a value
that is in compliance with congestion control principles for the
slowest link. The congestion control algorithm may be a TCP-friendly
bitrate adaptation algorithm specified in [RFC5348], or a DCCP
congestion control algorithms defined in [RFC5762].

If the substitutive content comes from local media file storage, the
mixer must directly reduce the bitrate as if it were the substitutive
RTP sender.

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 the
substitutive RTP sender, this sender 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

If it is desirable to prevent receivers from detecting that splicing
is occurring at the RTP layer, the mixer must not include a CSRC list
in outgoing RTP packets, and must not forward RTCP messages from the
main RTP sender or from the substitutive RTP sender. Due to the
absence of CSRC list in the output RTP stream, the RTP receiver only
initiates SDES, BYE and APP packets to the mixer without any
knowledge of the main RTP sender and the substitutive RTP sender.

CSRC list identifies the contributing sources, these SSRC identifiers of contributing sources are kept globally unique for each RTP session. The uniqueness of SSRC identifier is used to resolve collisions and detecting RTP-level forwarding loops as defined in section 8.2 of [RFC3550]. The absence of CSRC list in this case will create a danger that loops involving those contributing sources could not be detected. The Loops could occur if either the mixer is misconfigured to form a loop, or a second mixer/translator is added, causing packets to loop back to upstream of the original mixer. So Non-RTP means must be used to detect and resolve loops if the mixer does not add a CSRC list.

5. Implementation Considerations

When the 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 the mixer which uses its own sequence number space and timing model. So the mixer will rewrite RTP sequence number and timestamp whatever splicing is active or not, and generate RTCP flows for both sides. In case the mixer serves multiple main RTP streams simultaneously, this may lead to more overhead on the mixer.

If User Invisibility Requirement is required, CSRC list is not included in outgoing RTP packet, this brings a potential issue with loop detection as briefly described in section 4.5.

6. Security Considerations

The splicing application is subject to the general security considerations of the RTP specification [RFC3550].

The mixer acting as splicer replaces some content with other content in RTP packets, thus breaking any RTP level end-to-end security, such as integrity protection and source authentication. Thus any RTP level or outside security mechanism, such as IPsec or DTLS will use a security association between the splicer and the receiver. When using SRTP the splicer could be provisioned with the same security association as the main RTP sender. Using a limitation in the SRTP security services, the splicer can modify and re-protect the RTP packets without enabling the receiver to detect if the data comes from the original source or from the splicer.

Security goals to have source authentication all the way from the RTP main sender to the receiver through the splicer is not possible with splicing. The nature of this RTP service offered by a network

operator employing a content splicer is that the RTP layer security
relationship is between the receiver and the splicer, and between the
senders and the splicer, are not end-to-end. This appears to
invalidate the invisibility goal, but in the common case the receiver
will consider the splicer as the main media source.

Commonly no RTP level security mechanism is employed. Instead only
payload security mechanisms (e.g., ISMACryp [ISMACryp]) are used. If
any payload internal security mechanisms are used, only the RTP
sender and the RTP receiver can learn the security keying material
generated by such internal security mechanism, in which case, any
middlebox (e.g., splicer) between the RTP sender and the RTP receiver
can't get such keying material, and thus fail to perform splicing.

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, Charles Eckel and Ning Zong.

9. 10. Appendix- Why Mixer Is Chosen

Translator and mixer both can realize splicing by changing a set of
RTP parameters.

Translator has no SSRC, hence it is transparent to RTP sender and
receiver. Therefore, RTP sender sees the full path to the receiver
when translator is passing its content. When translator insert the
substitutive content RTP sender could get a report on the path up to
translator itself. Additionally, if splicing does not occur yet,
translator does not need to rewrite RTP header, the overhead on
translator can be avoided.

If mixer is used to do splicing, it can also allow RTP sender to
learn the situation of its content on receiver or on mixer just like
translator does, which is specified in section 4.2. Compared to
translator, mixer's outstanding benefit is that it is pretty straight
forward to do with RTCP messages, for example, bit-rate adaptation to
handle varying network conditions. But translator needs more
considerations and its implementation is more complex.

From above analysis, both translator and mixer have their own
advantages: less overhead or less complexity on handling RTCP.
Through long and sophisticated discussion, the avtext WG members
prefer less complexity rather than less overhead and incline to mixer
to do splicing.

If one chooses mixer as splicer, the overhead on mixer must be taken
into account even if the splicing does not occur yet.

10. References

10.1. Normative References

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

   [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.

   [I-D.ietf-avtcore-ecn-for-rtp]
              Westerlund, M., "Explicit Congestion Notification (ECN)
              for RTP over UDP", draft-ietf-avtcore-ecn-for-rtp-08 (work
              in progress), May 2012.

10.2. Informative References

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

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

   [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", 2009.

   [SCTE35]   Society of Cable Telecommunications Engineers (SCTE),
              "Digital Program Insertion Cueing Message for Cable",
              2011.

   [ISMACryp]
              Internet Streaming Media Alliance (ISMA), "ISMA Encryption
              and Authentication Specification 2.0", November 2007.

Author's Address

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

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