Abstract

This memo introduces a common RTP encapsulation for interleaved media. This method can be applied to any RTP payload formats for any applications when latency is not an issue.

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Copyright and License Notice

Copyright (c) 2015 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 before using this document.
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 .................................................. 3
2 Terminology .................................................. 3
3 Interleaving .................................................. 3
4 Interleaving RTP Payload Format ............................. 5
   4.1 RTP Header .............................................. 5
   4.2 Payload Structure ...................................... 6
   4.3 Encapsulation frames .................................. 6
   4.4 Example Packet ........................................ 8
5 SDP negotiation .............................................. 9
6 Security Considerations ..................................... 9
7 IANA Considerations ....................................... 9
8 Acknowledgments ........................................... 9
9 References .................................................. 9
   9.1 Normative References .................................. 9
   9.2 Informative References ................................. 10
Authors’ Addresses ........................................... 10
1 Introduction

Interleaving is an effective method to disperse packet loss bursts into a series of isolated small losses which in general are easier to recover from and produce lower total distortion. It provides the advantages of requiring no increase in bit rate and can be combined with other types of error-resilience techniques like FEC, but at the cost of increasing latency. Interleaving is quite useful for applications which are not such sensitive of latency in network environments afflicted by fading an interference which may lead to burst losses, e.g., DSL and wireless network.

This memo introduces a common RTP encapsulation for interleaved media. This method can be applied to any RTP payload formats for any applications when latency is not an issue.

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].

This document uses the following terms:

3. Interleaving

An interleaver is simply used at the sender or at a middle box to interleave the RTP packets before transmission through the network. There are a lot of interleaving algorithms. A simple one could be that packets are firstly read into the interleaver in rows, with each row corresponding to a sequence block of n packets; and packets are transmitted by columns as soon as m rows of packets fill up. The interleaver permutes the packets so that the location of burst losses are converted into isolated ones. The effectiveness of the interleaver depends on the block size and interleaving depth, however, this is at the cost of latency. At the receiver, an interleaved packet cannot be used until all the packets it depends on are received.

Figure 1 shows the interleaving scheme with n=4 and m=3. For this interleaver, the i-th packet in the original order has to be transmitted in the (((i-1) mod n)*m + ((i-1) div n) + 1)-th place. The highest decoding delay corresponding to this interleave is \((n-1)*(m-1)\).
This kind of interleaving is useful for audio applications that are non-interactive because it would turn a burst of consecutive lost packets into a series of isolated packet loss events. However, for video frames that are usually large enough to be fragmented into several packets, losing several non-consecutive packets from a large frame may not lead to perceptually less degradation. Thus, some implementations may use another interleaving to help loss recovery: The payload of each RTP packet is divided into \( n \) parts and the interleaver combines some part of one packet with some part of other packets to form a new RTP packet. As Figure 2 shows, each of the original RTP packets \( P_1, P_2, P_3, \) and \( P_4 \) are divided into 4 packets. The interleaver combines the first parts of \( P_1, P_2, P_3, \) and \( P_4 \) together to form a new packet \( P_1' \). \( P_2', P_3' \) and \( P_4' \) are formed in the same way. In this case, if \( P_1' \) is lost during the transmission, \( P_1, P_2, P_3, \) and \( P_4 \) can be easily recovered by FEC mechanism.

However, this interleaving method introduces extra complexity when dividing and recovering packets. To reduce delay in some degree, some ways can be considered. For example, the interleaver has the ability to categorize the RTP packets into important ones and unimportant ones based on some application dependent rules, and then only applies this interleaving to the important packets, e.g., packets containing I frames, IDR frames or some other information frames, while leave unimportant ones intact. The detail discussion of how to reduce the interleaving delay is out of scope of this document.
Proposing a common interleaving RTP encapsulation allows any RTP payload format to use the interleaving scheme freely when needed.

4. Interleaving RTP Payload Format

This section introduces a new PTP payload format dedicated for interleaving. That means interleaved RTP packets could use this new RTP payload format when transmitting. It is allowed that interleaving RTP payload format is transmitted together with the uninterleaved payload format so that the de-interleaver can identify the interleaved packets to recover. In such a case, the specifications defined in [draft-ietf-avtcore-multi-media-rtp-session] SHOULD be considered.

4.1 RTP Header

The format of RTP header is specified in [RFC3550] and showed in Figure 2 for convenience.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2 RTP header of RFC3550

The RTP header information to be set according to this RTP payload format is set as follows:

marker (M): 1 bit

The interpretation of the marker is determined by the interleaved frame encapsulated in this payload. It MUST be set to the value that the marker bit of the frame would have been if it were transported in its own RTP packets. If multiple frames are packed into one RTP packet, the marker bit in the RTP header MUST be set to the value in accordance with the last frame.

payload type (PT): 7 bits
The assignment of the RTP payload type for this format is outside the scope of this memo and will not be specified here. The assignment of a payload type has to be performed either through the profile used or in a dynamic way.

sequence number: 16 bits

Set and used in accordance with RFC 3550.

timestamp: 32 bits

Set to the value that the timestamp of the frame would have been if it were transported in its own RTP packets. If multiple frames are packed into one RTP packet, the timestamp in the RTP header MUST be set to the timestamp of the last frame which would have been if it were transported in its own RTP packets.

4.2 Payload Structure

This section describes the interleaving payload structure. The interleaving payload structure is composed from a fixed interleaving header and one or multiple encapsulation frames, which is illustrated in the following figure.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   EPT       |                                                 |
+-+-+-+-+-+-+-+-+                                                 |
|                                                               |
|               one or multiple encapsulation frames            |
|                                                               |
|                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               :...OPTIONAL RTP padding        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Format for the interleaving payload structure

encapsulation payload type (EPT): 7 bits

This field indicates the payload type of the encapsulation frames would have been if it were transported in their own RTP packets. Encapsulation frames with different payload type MUST NOT be packed into one RTP packet. The assignment of the encapsulation payload type has to be performed either through the profile used or in a dynamic way.

4.3 Encapsulation frames
The structure of encapsulation frames is showed in the figure 4.

```
<table>
<thead>
<tr>
<th>T</th>
<th>SN Offset</th>
<th>timestamp offset (optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
<td>-----------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>---</td>
<td>-----------</td>
<td>-----------------------------</td>
</tr>
</tbody>
</table>
```

Original RTP payload

```
<table>
<thead>
<tr>
<th>size (optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
</tbody>
</table>
```

Original RTP payload

```
: ...OPTIONAL RTP padding
```

Figure 4: Format for the encapsulation frame structure

capsulation frame type (T): 1 bit

This field indicates the encapsulation frame type.

T=0: Last encapsulation frame - This frame can be used alone or as the last encapsulation frame in an aggregation RTP packet which combines multiple encapsulation frames.

T=1: Aggregated encapsulation frame - This frame MUST be only used in an aggregation RTP packet combining multiple encapsulation frame. And it MUST NOT be used as the last encapsulation frame when used in the aggregation RTP packet.

SN Offset: 8 bits

This field records the signed offset between the original sequence number that the encapsulation frame would have been if it were transported in its own RTP packets and the sequence number field in this RTP header. The original sequence number implying the decoding order can be calculated as following:

original sequence number = sequence number + SN offset

A negative SN offset indicates that the encapsulation frame is delayed for transmission, and a non-negative SN offset means the encapsulation frame is transmitted ahead of it should be.

If multiple encapsulation frames combined in one RTP packet have the same SN offset, they SHOULD be handled according to their orders arranged in this packet.
timestamp offset: 23 bits

This field records the signed offset between the original
timestamp that the encapsulation frame would have been if it were
transported in its own RTP packets and the timestamp field in this
RTP header. It is only applicable for aggregated encapsulation
frame (T=1). The original timestamp can be calculated as
following:

original timestamp = timestamp + timestamp offset

size: 16 bits

This field is an unsigned size information of the following
encapsulation frame, which does not include encapsulation frame
type, SN offset and timestamp offset. It is only applicable for
aggregated encapsulation frame (T=1).

4.4 Example Packet

This section presents two example of an interleaving RTP packet.
Figure 5 shows an example of an packet that only contain one
encapsulation frame. In this example, the original timestamp is the
timestamp in the RTP header.

```
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----------------------------------------------+ RTP Header
+-----------------------------------------------+ EPT 0| SN Offset |
+-----------------------------------------------+
|                                                           |
| Original RTP payload                                    |
+-----------------------------------------------+...OPTIONAL RTP padding
|                                                                 |
```

Figure 5 An interleaving RTP packet
including only one encapsulation frame

Figure 6 shows an example of an packet that contains 3 encapsulation
frame, labeled as 1, 2 and 3 in the figure.

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

<Author> Expires January 7, 2016 [Page 8]
Figure 5 An interleaving RTP packet including 3 encapsulation frames
9.2 Informative References

Authors’ Addresses

Rachel Huang
Huawei
101 Software Avenue, Yuhua District
Nanjing 210012
China

Email: rachel.huang@huawei.com