Payload Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 4, 2015

P. Westin
H. Lundin
M. Glover
J. Uberti
F. Galligan
Google

March 3, 2015

RTP Payload Format for VP8 Video
draft-ietf-payload-vp8-14

Abstract

This memo describes an RTP payload format for the VP8 video codec. The payload format has wide applicability, as it supports applications from low bit-rate peer-to-peer usage, to high bit-rate video conferences.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This memo describes an RTP payload specification applicable to the transmission of video streams encoded using the VP8 video codec [RFC6386]. The format described in this document can be used both in peer-to-peer and video conferencing applications.

VP8 is based on decomposition of frames into square sub-blocks of pixels, prediction of such sub-blocks using previously constructed blocks, and adjustment of such predictions (as well as synthesis of unpredicted blocks) using a discrete cosine transform (hereafter

abbreviated as DCT). In one special case, however, VP8 uses a "Walsh-Hadamard" (hereafter abbreviated as WHT) transform instead of a DCT. An encoded VP8 frame is divided into two or more partitions, as described in [RFC6386]. The first partition (prediction or mode) contains prediction mode parameters and motion vectors for all macroblocks. The remaining partitions all contain the quantized DCT/WHT coefficients for the residuals. There can be 1, 2, 4, or 8 DCT/WHT partitions per frame, depending on encoder settings.

In summary, the payload format described in this document enables a number of features in VP8, including:

o Taking partition boundaries into consideration, to improve loss robustness and facilitate efficient packet loss concealment at the decoder.

o Temporal scalability.

o Advanced use of reference frames to enable efficient error recovery.

o Marking of frames that have no impact on the decoding of any other frame, so that these non-reference frames can be discarded in a server or media-aware network element if needed.

2. Conventions, Definitions and Acronyms

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 [RFC2119].

3. Media Format Description

The VP8 codec uses three different reference frames for interframe prediction: the previous frame, the golden frame, and the altref frame. Blocks in an interframe may be predicted using blocks in the immediately previous frame as well as the most recent golden frame or altref frame. Every key frame is automatically golden and altref, and any interframe may optionally replace the most recent golden or altref frame. Golden frames and altref frames may also be used to increase the tolerance to dropped frames. The payload specification in this memo has elements that enable advanced use of the reference frames, e.g., for improved loss robustness.

One specific use case of the three reference frame types is temporal scalability. By setting up the reference hierarchy in the appropriate way, up to five temporal layers can be encoded. (How to

set up the reference hierarchy for temporal scalability is not within the scope of this memo.)

Another property of the VP8 codec is that it applies data partitioning to the encoded data. Thus, an encoded VP8 frame can be divided into two or more partitions, as described in "VP8 Data Format and Decoding Guide" [RFC6386]. The first partition (prediction or mode) contains prediction mode parameters and motion vectors for all macroblocks. The remaining partitions all contain the transform coefficients for the residuals. The first partition is decodable without the remaining residual partitions. The subsequent partitions may be useful even if some part of the frame is lost. This memo allows the partitions to be sent separately or in the same RTP packet. It may be beneficial for decoder error-concealment to send the partitions in different packets, even though it is not mandatory according to this specification.

The format specification is described in Section 4. In Section 5, a method to acknowledge receipt of reference frames using RTCP techniques is described.

The payload partitioning and the acknowledging method both serve as motivation for three of the fields included in the payload format: the "PID", "1st partition size" and "PictureID" fields. The ability to encode a temporally scalable stream motivates the "TL0PICIDX" and "TID" fields.

4. Payload Format

This section describes how the encoded VP8 bitstream is encapsulated in RTP. To handle network losses usage of RTP/AVPF [RFC4585] is RECOMMENDED. All integer fields in the specifications are encoded as unsigned integers in network octet order.

4.1. RTP Header Usage

The general RTP payload format for VP8 is depicted below.

```
  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
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |V=2|P|X|  CC   |M|     PT      |       sequence number         |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                           timestamp                           |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |           synchronization source (SSRC) identifier            |
 +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
 |            contributing source (CSRC) identifiers             |
 |                             ....                              |
 +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
 |            VP8 payload descriptor (integer #bytes)            |
 :                                                               :
 |                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                               : VP8 payload header (3 octets) |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | VP8 pyld hdr  :                                               |
 +-+-+-+-+-+-+-+-+                                               |
 :                   Bytes 4..N of VP8 payload                   :
 |                                                               |
 |                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                               :    OPTIONAL RTP padding       |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The VP8 payload descriptor and VP8 payload header will be described in the sequel. OPTIONAL RTP padding MUST NOT be included unless the P bit is set.

Figure 1

Marker bit (M): MUST be set for the very last packet of each encoded frame in line with the normal use of the M bit in video formats. This enables a decoder to finish decoding the picture, where it otherwise may need to wait for the next packet to explicitly know that the frame is complete.

Timestamp: The RTP timestamp indicates the time when the frame was sampled at a clock rate of 90 kHz.

Sequence number: The sequence numbers are monotonically increasing and set as packets are sent.

The remaining RTP header fields are used as specified in [RFC3550].

4.2. VP8 Payload Descriptor

The first octets after the RTP header are the VP8 payload descriptor, with the following structure. The single-octet version of the PictureID is illustrated to the left (M bit set to zero), while the dual-octet version (M bit set to one) is show to the right.

```
      0 1 2 3 4 5 6 7                      0 1 2 3 4 5 6 7
     +-+-+-+-+-+-+-+-+                    +-+-+-+-+-+-+-+-+
     |X|R|N|S|R| PID | (REQUIRED)         |X|R|N|S|R| PID | (REQUIRED)
     +-+-+-+-+-+-+-+-+                    +-+-+-+-+-+-+-+-+
X:   |I|L|T|K| RSV   | (OPTIONAL)   X:    |I|L|T|K| RSV   | (OPTIONAL)
     +-+-+-+-+-+-+-+-+                    +-+-+-+-+-+-+-+-+
I:   |M| PictureID   | (OPTIONAL)   I:    |M| PictureID   | (OPTIONAL)
     +-+-+-+-+-+-+-+-+                    +-+-+-+-+-+-+-+-+
L:   |   TL0PICIDX   | (OPTIONAL)         |   PictureID   |
     +-+-+-+-+-+-+-+-+                    +-+-+-+-+-+-+-+-+
T/K: |TID|Y| KEYIDX  | (OPTIONAL)   L:    |   TL0PICIDX   | (OPTIONAL)
     +-+-+-+-+-+-+-+-+                    +-+-+-+-+-+-+-+-+
                                    T/K:  |TID|Y| KEYIDX  | (OPTIONAL)
                                          +-+-+-+-+-+-+-+-+
```

Figure 2

X: Extended control bits present. When set to one, the extension octet MUST be provided immediately after the mandatory first octet. If the bit is zero, all optional fields MUST be omitted.

R: Bit reserved for future use. MUST be set to zero and MUST be ignored by the receiver.

N: Non-reference frame. When set to one, the frame can be discarded without affecting any other future or past frames. If the reference status of the frame is unknown, this bit SHOULD be set to zero to avoid discarding frames needed for reference.

> Informative note: This document does not describe how to determine if an encoded frame is non-reference. The reference status of an encoded frame is preferably provided from the encoder implementation.

S: Start of VP8 partition. SHOULD be set to 1 when the first payload octet of the RTP packet is the beginning of a new VP8 partition, and MUST NOT be 1 otherwise. The S bit MUST be set to 1 for the first packet of each encoded frame.

PID: Partition index. Denotes which VP8 partition the first payload octet of the packet belongs to. The first VP8 partition

(containing modes and motion vectors) MUST be labeled with PID = 0. PID SHOULD be incremented for each subsequent partition, but MAY be kept at 0 for all packets. PID MUST NOT be larger than 7. If more than one packet in an encoded frame contains the same PID, the S bit MUST NOT be set for any other packet than the first packet with that PID.

When the X bit is set to 1 in the first octet, the extension bit field octet MUST be provided as the second octet. If the X bit is 0, the extension bit field octet MUST NOT be present, and all bits below MUST be implicitly interpreted as 0.

I: PictureID present. When set to one, the OPTIONAL PictureID MUST be present after the extension bit field and specified as below. Otherwise, PictureID MUST NOT be present.

L: TL0PICIDX present. When set to one, the OPTIONAL TL0PICIDX MUST be present and specified as below, and the T bit MUST be set to 1. Otherwise, TL0PICIDX MUST NOT be present.

T: TID present. When set to one, the OPTIONAL TID/KEYIDX octet MUST be present. The TID|Y part of the octet MUST be specified as below. If K (below) is set to one but T is set to zero, the TID/KEYIDX octet MUST be present, but the TID|Y field MUST be ignored. If neither T nor K is set to one, the TID/KEYIDX octet MUST NOT be present.

K: KEYIDX present. When set to one, the OPTIONAL TID/KEYIDX octet MUST be present. The KEYIDX part of the octet MUST be specified as below. If T (above) is set to one but K is set to zero, the TID/KEYIDX octet MUST be present, but the KEYIDX field MUST be ignored. If neither T nor K is set to one, the TID/KEYIDX octet MUST NOT be present.

RSV: Bits reserved for future use. MUST be set to zero and MUST be ignored by the receiver.

After the extension bit field follow the extension data fields that are enabled.

M: The most significant bit of the first octet is an extension flag. The field MUST be present if the I bit is equal to one. If set the PictureID field MUST contain 16 bits else it MUST contain 8 bits including this MSB, see PictureID.

PictureID: 8 or 16 bits (shown left and right, respectively, in Figure 2) including the M bit. This is a running index of the frames. The field MUST be present if the I bit is equal to one.

The 7 following bits carry (parts of) the PictureID. If the extension flag is one, the PictureID continues in the next octet forming a 15 bit index, where the 8 bits in the second octet are the least significant bits of the PictureID. If the extension flag is zero, there is no extension, and the PictureID is the 7 remaining bits of the first (and only) octet. The sender may choose 7 or 15 bits index. The PictureID SHOULD start on a random number, and MUST wrap after reaching the maximum ID. The receiver MUST NOT assume that the number of bits in PictureID stay the same through the session.

TL0PICIDX: 8 bits temporal level zero index. The field MUST be present if the L bit is equal to 1, and MUST NOT be present otherwise. TL0PICIDX is a running index for the temporal base layer frames, i.e., the frames with TID set to 0. If TID is larger than 0, TL0PICIDX indicates which base layer frame the current image depends on. TL0PICIDX MUST be incremented when TID is 0. The index SHOULD start on a random number, and MUST restart at 0 after reaching the maximum number 255.

TID: 2 bits temporal layer index. The TID/KEYIDX octet MUST be present when either the T bit or the K bit or both are equal to 1, and MUST NOT be present otherwise. The TID field MUST be ignored by the receiver when the T bit is set equal to 0. The TID field indicates which temporal layer the packet represents. The lowest layer, i.e., the base layer, MUST have TID set to 0. Higher layers SHOULD increment the TID according to their position in the layer hierarchy.

Y: 1 layer sync bit. The TID/KEYIDX octet MUST be present when either the T bit or the K bit or both are equal to 1, and MUST NOT be present otherwise. The Y bit SHOULD be set to 1 if the current frame depends only on the base layer (TID = 0) frame with TL0PICIDX equal to that of the current frame. The Y bit MUST be set to 0 if the current frame depends any other frame than the base layer (TID = 0) frame with TL0PICIDX equal to that of the current frame. If the Y bit is set when the T bit is equal to 0 the current frame MUST only depend on a past base layer (TID=0) key frame as signaled by a change in the KEYIDX field. Additionally this frame MUST NOT depend on any of the three codec buffers (as defined by [RFC6386]) that have been updated since the last time the KEYIDX field was changed.

Informative note: This document does not describe how to determine the dependence status for a frame; this information is preferably provided from the encoder implementation. In the case of unknown status, the Y bit can safely be set to 0.

KEYIDX: 5 bits temporal key frame index. The TID/KEYIDX octet MUST be present when either the T bit or the K bit or both are equal to 1, and MUST NOT be present otherwise. The KEYIDX field MUST be ignored by the receiver when the K bit is set equal to 0. The KEYIDX field is a running index for key frames. KEYIDX MAY start on a random number, and MUST restart at 0 after reaching the maximum number 31. When in use, the KEYIDX SHOULD be present for both key frames and interframes. The sender MUST increment KEYIDX for key frames which convey parameter updates critical to the interpretation of subsequent frames, and SHOULD leave the KEYIDX unchanged for key frames that do not contain these critical updates. If the KEYIDX is present, a receiver SHOULD NOT decode an interframe if it has not received and decoded a key frame with the same KEYIDX after the last KEYIDX wrap-around.

Informative note: This document does not describe how to determine if a key frame updates critical parameters; this information is preferably provided from the encoder implementation. A sender that does not have this information may either omit the KEYIDX field (set K equal to 0), or increment the KEYIDX on every key frame. The benefit with the latter is that any key frame loss will be detected by the receiver, which can signal for re-transmission or request a new key frame.

Informative note: Implementations doing splicing of VP8 streams will have to make sure the rules for incrementing TL0PICIDX and KEYIDX are obeyed across the splice. This will likely require rewriting values of TL0PICIDX and KEYIDX after the splice.

4.3. VP8 Payload Header

The beginning of an encoded VP8 frame is referred to as an "uncompressed data chunk" in [RFC6386], and co-serve as payload header in this RTP format. The codec bitstream format specifies two different variants of the uncompressed data chunk: a 3 octet version for interframes and a 10 octet version for key frames. The first 3 octets are common to both variants. In the case of a key frame the remaining 7 octets are considered to be part of the remaining payload in this RTP format. Note that the header is present only in packets which have the S bit equal to one and the PID equal to zero in the payload descriptor. Subsequent packets for the same frame do not carry the payload header.

```
  0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+
 |Size0|H| VER |P|
 +-+-+-+-+-+-+-+-+
 |     Size1     |
 +-+-+-+-+-+-+-+-+
 |     Size2     |
 +-+-+-+-+-+-+-+-+
 | Bytes 4..N of |
 | VP8 payload   |
 :               :
 +-+-+-+-+-+-+-+-+
 | OPTIONAL RTP  |
 | padding       |
 :               :
 +-+-+-+-+-+-+-+-+
```

Figure 3

H: Show frame bit as defined in [RFC6386].

VER: A version number as defined in [RFC6386].

P: Inverse key frame flag. When set to 0 the current frame is a key frame. When set to 1 the current frame is an interframe. Defined in [RFC6386]

SizeN: The size of the first partition in bytes is calculated from the 19 bits in Size0, Size1, and Size2 as 1stPartitionSize = Size0 + 8 * Size1 + 2048 * Size2. [RFC6386].

4.4. Aggregated and Fragmented Payloads

An encoded VP8 frame can be divided into two or more partitions, as described in Section 1. One packet can contain a fragment of a partition, a complete partition, or an aggregate of fragments and partitions. In the preferred use case, the S bit and PID fields described in Section 4.2 should be used to indicate what the packet contains. The PID field should indicate which partition the first octet of the payload belongs to, and the S bit indicates that the packet starts on a new partition. Aggregation of encoded partitions is done without explicit signaling. Partitions MUST be aggregated in decoding order. Two fragments from different partitions MAY be aggregated into the same packet. An aggregation MUST have exactly one payload descriptor. Aggregated partitions MUST represent parts of one and the same video frame. Consequently, an aggregated packet will have one or no payload header, depending on whether the aggregate contains the beginning of the first partition of a frame or not, respectively. Note that the length of the first partition can always be obtained from the first partition size parameter in the VP8 payload header.

The VP8 bitstream format [RFC6386] specifies that if multiple DCT/WHT partitions are produced, the location of each partition start is found at the end of the first (prediction/mode) partition. In this RTP payload specification, the location offsets are considered to be part of the first partition.

It is OPTIONAL for a packetizer implementing this RTP specification to pay attention to the partition boundaries within an encoded frame. If packetization of a frame is done without considering the partition boundaries, the PID field MAY be set to zero for all packets, and the S bit MUST NOT be set to one for any other packet than the first.

4.5. Frame reconstruction algorithm

Example of frame reconstruction algorithm.

1: Collect all packets with a given RTP timestamp.

2: Go through packets in order, sorted by sequence numbers, if packets are missing, send NACK as defined in [RFC4585] or decode with missing partitions, see Section 4.5.1 below.

3: A frame is complete if the frame has no missing sequence numbers, the first packet in the frame contains S=1 with partId=0 and the last packet in the frame has the marker bit set.

4.5.1. Partition reconstruction algorithm

Example of partition reconstruction algorithm.

1: Scan for the start of a new partition; S=1.

2: Continue scan to detect end of partition; hence a new S=1 (previous packet was the end of the partition) is found or the marker bit is set. If a loss is detected before the end of the partition, abandon all packets in this partition and continue the scan repeating from step 1.

3: Store the packets in the complete partition, continue the scan repeating from step 1 until end of frame is reached.

4: Send all complete partitions to the decoder. If no complete partition is found discard the whole frame.

4.6. Examples of VP8 RTP Stream

A few examples of how the VP8 RTP payload can be used are included below.

4.6.1. Key frame in a single RTP packet

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|  RTP header   |
|  M = 1        |
+-+-+-+-+-+-+-+-+
|1|0|0|1|0|0 0 0| X = 1; S = 1; PID = 0
+-+-+-+-+-+-+-+-+
|1|0|0|0|0 0 0 0| I = 1
+-+-+-+-+-+-+-+-+
|0 0 0 1 0 0 0 1| PictureID = 17
+-+-+-+-+-+-+-+-+
|Size0|1| VER |0| P = 0
+-+-+-+-+-+-+-+-+
|     Size1     |
+-+-+-+-+-+-+-+-+
|     Size2     |
+-+-+-+-+-+-+-+-+
| VP8 payload   |
+-+-+-+-+-+-+-+-+
```

4.6.2. Non-discardable VP8 interframe in a single RTP packet; no PictureID

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|  RTP header   |
|  M = 1        |
+-+-+-+-+-+-+-+-+
|0|0|0|1|0|0 0 0| X = 0; S = 1; PID = 0
+-+-+-+-+-+-+-+-+
|Size0|1| VER |1| P = 1
+-+-+-+-+-+-+-+-+
|     Size1     |
+-+-+-+-+-+-+-+-+
|     Size2     |
+-+-+-+-+-+-+-+-+
| VP8 payload   |
+-+-+-+-+-+-+-+-+
```

4.6.3. VP8 partitions in separate RTP packets

First RTP packet; complete first partition.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|  RTP header   |
|  M = 0        |
+-+-+-+-+-+-+-+-+
|1|0|0|1|0|0 0 0| X = 1; S = 1; PID = 0
+-+-+-+-+-+-+-+-+
|1|0|0|0|0 0 0 0| I = 1
+-+-+-+-+-+-+-+-+
|0 0 0 1 0 0 0 1| PictureID = 17
+-+-+-+-+-+-+-+-+
|Size0|1| VER |1| P = 1
+-+-+-+-+-+-+-+-+
|     Size1     |
+-+-+-+-+-+-+-+-+
|     Size2     |
+-+-+-+-+-+-+-+-+
| Bytes 4..L of |
| first VP8     |
| partition     |
:               :
+-+-+-+-+-+-+-+-+
```

Second RTP packet; complete second partition.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|  RTP header   |
|  M = 1        |
+-+-+-+-+-+-+-+-+
|1|0|0|1|0|0 0 1| X = 1; S = 1; PID = 1
+-+-+-+-+-+-+-+-+
|1|0|0|0|0 0 0 0| I = 1
+-+-+-+-+-+-+-+-+
|0 0 0 1 0 0 0 1| PictureID = 17
+-+-+-+-+-+-+-+-+
| Remaining VP8 |
| partitions    |
:               :
+-+-+-+-+-+-+-+-+
```

4.6.4. VP8 frame fragmented across RTP packets

First RTP packet; complete first partition.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|  RTP header   |
|  M = 0        |
+-+-+-+-+-+-+-+-+
|1|0|0|1|0|0 0 0| X = 1; S = 1; PID = 0
+-+-+-+-+-+-+-+-+
|1|0|0|0|0 0 0 0| I = 1
+-+-+-+-+-+-+-+-+
|0 0 0 1 0 0 0 1| PictureID = 17
+-+-+-+-+-+-+-+-+
|Size0|1| VER |1| P = 1
+-+-+-+-+-+-+-+-+
|     Size1     |
+-+-+-+-+-+-+-+-+
|     Size2     |
+-+-+-+-+-+-+-+-+
| Complete      |
| first         |
| partition     |
:               :
+-+-+-+-+-+-+-+-+
```

Second RTP packet; first fragment of second partition.

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|  RTP header   |
|  M = 0        |
+-+-+-+-+-+-+-+-+
|1|0|0|1|0|0 0 1| X = 1; S = 1; PID = 1
+-+-+-+-+-+-+-+-+
|1|0|0|0|0 0 0 0| I = 1
+-+-+-+-+-+-+-+-+
|0 0 0 1 0 0 0 1| PictureID = 17
+-+-+-+-+-+-+-+-+
| First fragment|
| of second     |
| partition     |
:               :
+-+-+-+-+-+-+-+-+
```

Third RTP packet; second fragment of second partition.

```
  0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+
 |  RTP header   |
 |  M = 0        |
 +-+-+-+-+-+-+-+-+
 |1|0|0|0|0|0 0 1| X = 1; S = 0; PID = 1
 +-+-+-+-+-+-+-+-+
 |1|0|0|0|0 0 0 0| I = 1
 +-+-+-+-+-+-+-+-+
 |0 0 0 1 0 0 0 1| PictureID = 17
 +-+-+-+-+-+-+-+-+
 | Mid fragment  |
 | of second     |
 | partition     |
 :               :
 +-+-+-+-+-+-+-+-+
```

Fourth RTP packet; last fragment of second partition.

```
  0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+
 |  RTP header   |
 |  M = 1        |
 +-+-+-+-+-+-+-+-+
 |1|0|0|0|0|0 0 1| X = 1; S = 0; PID = 1
 +-+-+-+-+-+-+-+-+
 |1|0|0|0|0 0 0 0| I = 1
 +-+-+-+-+-+-+-+-+
 |0 0 0 1 0 0 0 1| PictureID = 17
 +-+-+-+-+-+-+-+-+
 | Last fragment |
 | of second     |
 | partition     |
 :               :
 +-+-+-+-+-+-+-+-+
```

4.6.5. VP8 frame with long PictureID

PictureID = 4711 = 001001001100111 binary (first 7 bits: 0010010, last 8 bits: 01100111).

```
  0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+
 |  RTP header   |
 |  M = 1        |
 +-+-+-+-+-+-+-+-+
 |1|0|0|1|0|0 0 0| X = 1; S = 1; PID = 0
 +-+-+-+-+-+-+-+-+
 |1|0|0|0|0 0 0 0| I = 1;
 +-+-+-+-+-+-+-+-+
 |1 0 0 1 0 0 1 0| Long PictureID flag = 1
 |0 1 1 0 0 1 1 1| PictureID = 4711
 +-+-+-+-+-+-+-+-+
 |Size0|1| VER |1|
 +-+-+-+-+-+-+-+-+
 |     Size1     |
 +-+-+-+-+-+-+-+-+
 |     Size2     |
 +-+-+-+-+-+-+-+-+
 | Bytes 4..N of |
 | VP8 payload   |
 :               :
 +-+-+-+-+-+-+-+-+
```

5. Using VP8 with RPSI and SLI Feedback

The VP8 payload descriptor defined in Section 4.2 above contains an optional PictureID parameter. This parameter is included mainly to enable use of reference picture selection index (RPSI) and slice loss indication (SLI), both defined in [RFC4585].

5.1. RPSI

The reference picture selection index is a payload-specific feedback message defined within the RTCP-based feedback format. The RPSI message is generated by a receiver and can be used in two ways. Either it can signal a preferred reference picture when a loss has been detected by the decoder -- preferably then a reference that the decoder knows is perfect -- or, it can be used as positive feedback information to acknowledge correct decoding of certain reference pictures. The positive feedback method is useful for VP8 used as unicast. The use of RPSI for VP8 is preferably combined with a special update pattern of the codec's two special reference frames -- the golden frame and the altref frame -- in which they are updated in an alternating leapfrog fashion. When a receiver has received and correctly decoded a golden or altref frame, and that frame had a

PictureID in the payload descriptor, the receiver can acknowledge this simply by sending an RPSI message back to the sender. The message body (i.e., the "native RPSI bit string" in [RFC4585]) is simply the PictureID of the received frame.

5.2. SLI

The slice loss indication is another payload-specific feedback message defined within the RTCP-based feedback format. The SLI message is generated by the receiver when a loss or corruption is detected in a frame. The format of the SLI message is as follows [RFC4585]:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         First           |        Number             | PictureID |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4

Here, First is the macroblock address (in scan order) of the first lost block and Number is the number of lost blocks. PictureID is the six least significant bits of the codec-specific picture identifier in which the loss or corruption has occurred. For VP8, this codec-specific identifier is naturally the PictureID of the current frame, as read from the payload descriptor. If the payload descriptor of the current frame does not have a PictureID, the receiver MAY send the last received PictureID+1 in the SLI message. The receiver MAY set the First parameter to 0, and the Number parameter to the total number of macroblocks per frame, even though only parts of the frame is corrupted. When the sender receives an SLI message, it can make use of the knowledge from the latest received RPSI message. Knowing that the last golden or altref frame was successfully received, it can encode the next frame with reference to that established reference.

5.3. Example

The use of RPSI and SLI is best illustrated in an example. In this example, the encoder may not update the altref frame until the last sent golden frame has been acknowledged with an RPSI message. If an update is not received within some time, a new golden frame update is sent instead. Once the new golden frame is established and acknowledged, the same rule applies when updating the altref frame.

```
+-------+-------------------+-------------------------+-------------+
| Event | Sender            | Receiver                | Established |
```

```
   |       |                   |                         | reference   |
   +-------+-------------------+-------------------------+-------------+
   | 1000  | Send golden frame |                         |             |
   |       | PictureID = 0     |                         |             |
   |       |                   |                         |             |
   |       |                   | Receive and decode      |             |
   |       |                   | golden frame            |             |
   |       |                   |                         |             |
   | 1001  |                   | Send RPSI(0)            |             |
   |       |                   |                         |             |
   | 1002  | Receive RPSI(0)   |                         | golden      |
   |       |                   |                         |             |
   | ...   | (sending regular  |                         |             |
   |       | frames)           |                         |             |
   |       |                   |                         |             |
   | 1100  | Send altref frame |                         |             |
   |       | PictureID = 100   |                         |             |
   |       |                   |                         |             |
   |       |                   | Altref corrupted or     | golden      |
   |       |                   | lost                    |             |
   |       |                   |                         |             |
   | 1101  |                   | Send SLI(100)           | golden      |
   |       |                   |                         |             |
   | 1102  | Receive SLI(100)  |                         |             |
   |       |                   |                         |             |
   | 1103  | Send frame with   |                         |             |
   |       | reference to      |                         |             |
   |       | golden            |                         |             |
   |       |                   |                         |             |
   |       |                   | Receive and decode      | golden      |
   |       |                   | frame (decoder state    |             |
   |       |                   | restored)               |             |
   |       |                   |                         |             |
   | ...   | (sending regular  |                         |             |
   |       | frames)           |                         |             |
   |       |                   |                         |             |
   | 1200  | Send altref frame |                         |             |
   |       | PictureID = 200   |                         |             |
   |       |                   |                         |             |
   |       |                   | Receive and decode      | golden      |
   |       |                   | altref frame            |             |
   |       |                   |                         |             |
   | 1201  |                   | Send RPSI(200)          |             |
   |       |                   |                         |             |
   | 1202  | Receive RPSI(200) |                         | altref      |
   |       |                   |                         |             |
   | ...   | (sending regular  |                         |             |
   |       | frames)           |                         |             |
```

1300	Send golden frame PictureID = 300		
		Receive and decode golden frame	altref
1301		Send RPSI(300)	altref
1302	RPSI lost		
1400	Send golden frame PictureID = 400		
		Receive and decode golden frame	altref
1401		Send RPSI(400)	
1402	Receive RPSI(400)		golden

Table 1: Exemple signaling between sender and receiver

Note that the scheme is robust to loss of the feedback messages. If the RPSI is lost, the sender will try to update the golden (or altref) again after a while, without releasing the established reference. Also, if an SLI is lost, the receiver can keep sending SLI messages at any interval allowed by the RTCP sending timing restrictions as specified in [RFC4585], as long as the picture is corrupted.

6. Payload Format Parameters

This payload format has two required parameters.

6.1. Media Type Definition

This registration is done using the template defined in [RFC6838] and following [RFC4855].

Type name: video

Subtype name: VP8

Required parameters:

These parameters MUST be used to signal the capabilities of a receiver implementation. These parameters MUST NOT be used for any other purpose.

max-fr: The value of max-fr is an integer indicating the maximum frame rate in units of frames per second that the decoder is capable of decoding.

max-fs: The value of max-fs is an integer indicating the maximum frame size in units of macroblocks that the decoder is capable of decoding.

The decoder is capable of decoding this frame size as long as the width and height of the frame in macroblocks are less than int(sqrt(max-fs * 8)) - for instance, a max-fs of 1200 (capable of supporting 640x480 resolution) will support widths and heights up to 1552 pixels (97 macroblocks).

Optional parameters: none

Encoding considerations:
This media type is framed in RTP and contains binary data; see Section 4.8 of [RFC6838].

Security considerations: See Section 7 of RFC xxxx.
[RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

Interoperability considerations: None.

Published specification: VP8 bitstream format [RFC6386] and RFC XXXX.
[RFC Editor: Upon publication as an RFC, please replace "XXXX" with the number assigned to this document and remove this note.]

Applications which use this media type:
For example: Video over IP, video conferencing.

Additional information: None.

Person & email address to contact for further information:
Patrik Westin, patrik.westin@gmail.com

Intended usage: COMMON

Restrictions on usage:
This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550].

Author: Patrik Westin, patrik.westin@gmail.com

Change controller:
IETF Payload Working Group delegated from the IESG.

6.2. SDP Parameters

The receiver MUST ignore any fmtp parameter unspecified in this memo.

6.2.1. Mapping of Media Subtype Parameters to SDP

The media type video/VP8 string is mapped to fields in the Session Description Protocol (SDP) [RFC4566] as follows:

- The media name in the "m=" line of SDP MUST be video.
- The encoding name in the "a=rtpmap" line of SDP MUST be VP8 (the media subtype).
- The clock rate in the "a=rtpmap" line MUST be 90000.
- The parameters "max-fs", and "max-fr", MUST be included in the "a=fmtp" line if the SDP is used to declare receiver capabilities. These parameters are expressed as a media subtype string, in the form of a semicolon separated list of parameter=value pairs.

6.2.1.1. Example

An example of media representation in SDP is as follows:

```
m=video 49170 RTP/AVPF 98
a=rtpmap:98 VP8/90000
a=fmtp:98 max-fr=30; max-fs=3600;
```

6.2.2. Offer/Answer Considerations

The VP8 codec offers a decode complexity that is roughly linear with the number of pixels encoded. The parameters "max-fr" and "max-fs" are defined in Section 6.1, where the macroblock size is 16x16 pixels as defined in [RFC6386], the max-fs and max-fr parameters MUST be used to establish these limits.

7. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550], and in any applicable RTP profile. The main security considerations for the RTP packet carrying the RTP payload

format defined within this memo are confidentiality, integrity and source authenticity. Confidentiality is achieved by encryption of the RTP payload. Integrity of the RTP packets through suitable cryptographic integrity protection mechanism. Cryptographic system may also allow the authentication of the source of the payload. A suitable security mechanism for this RTP payload format should provide confidentiality, integrity protection and at least source authentication capable of determining if an RTP packet is from a member of the RTP session or not. Note that the appropriate mechanism to provide security to RTP and payloads following this memo may vary. It is dependent on the application, the transport, and the signaling protocol employed. Therefore a single mechanism is not sufficient, although if suitable the usage of SRTP [RFC3711] is recommended. This RTP payload format and its media decoder do not exhibit any significant non-uniformity in the receiver-side computational complexity for packet processing, and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological data. Nor does the RTP payload format contain any active content.

8. Congestion Control

Congestion control for RTP SHALL be used in accordance with RFC 3550 [RFC3550], and with any applicable RTP profile; e.g., RFC 3551 [RFC3551]. The congestion control mechanism can, in a real-time encoding scenario, adapt the transmission rate by instructing the encoder to encode at a certain target rate. Media aware network elements MAY use the information in the VP8 payload descriptor in Section 4.2 to identify non-reference frames and discard them in order to reduce network congestion. Note that discarding of non-reference frames cannot be done if the stream is encrypted (because the non-reference marker is encrypted).

9. IANA Considerations

The IANA is requested to register the following values:
- Media type registration as described in Section 6.1.

10. 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.

[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.

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

[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.

[RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, February 2007.

[RFC6386] Bankoski, J., Koleszar, J., Quillio, L., Salonen, J., Wilkins, P., and Y. Xu, "VP8 Data Format and Decoding Guide", RFC 6386, November 2011.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, January 2013.

Authors' Addresses

Patrik Westin
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
USA

Email: patrik.westin@gmail.com

Henrik F Lundin
Google, Inc.
Kungsbron 2
Stockholm 11122
Sweden

Email: hlundin@google.com

Michael Glover
Google, Inc.
5 Cambridge Center
Cambridge, MA 02142
USA

Justin Uberti
Google, Inc.
747 6th Street South
Kirkland, WA 98033
USA

Frank Galligan
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
USA