Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 11, 2011

M. Ramalho, Ed.
P. Jones
Cisco Systems
N. Harada
NTT
M. Perumal
Cisco Systems
L. Miao
Huawei Technologies
June 9, 2011

RTP Payload Format for G.711.0
draft-ramalho-payload-g7110-00

Abstract

This document specifies the Real-Time Transport Protocol (RTP) payload format for ITU-T Recommendation G.711.0. ITU-T Rec. G.711.0 defines a lossless and stateless compression for G.711 packet payloads typically used in IP networks. This document also defines two storage mode formats for G.711.0. A media type registration for this RTP payload format is also included.

Status of this Memo

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

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

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

This Internet-Draft will expire on December 11, 2011.

Copyright Notice

Table of Contents

1. Introduction

The International Telecommunication Union (ITU-T) Recommendation G.711.0 [G.711.0] specifies a stateless and lossless compression for G.711 packet payloads typically used in VoIP networks. This document specifies the Real-Time Transport Protocol (RTP) RFC 3550 [RFC3550] payload format and storage modes for this compression.

2. Requirements Language

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].

3. G.711.0 Codec Background

ITU-T Recommendation G.711.0 [G.711.0] is a lossless and stateless compression mechanism for ITU-T Recommendation G.711 [G.711] and thus is not a "codec" in the sense of "lossy" codecs typically carried by RTP. When negotiated end-to-end ITU-T Rec. G.711.0 is negotiated as if it were a codec, with the understanding that ITU-T Rec. G.711.0 losslessly encoded the underlying (lossy) G.711 pulse code modulation (PCM) sample representation of an audio signal. For this reason ITU-T Rec. G.711.0 will be interchangeably referred to in this document as a "lossless data compression algorithm" or a "codec", depending on context. Within this document, individual G.711 PCM samples will be referred to as "G.711 symbols" or just "symbols" for brevity.

This section describes the ITU-T Recommendation G.711 [G.711] codec, its properties, typical uses cases and its key design properties.

3.1. General Information and Use of the ITU-T G.711.0 Codec

ITU-T Recommendation G.711 is the benchmark standard for narrowband telephony. It has been successful for many decades because of its proven voice quality, ubiquity and utility. A new ITU-T recommendation, G.711.0, has been established for defining a stateless and lossless compression for G.711 packet payloads typically used in VoIP networks. ITU-T Rec. G.711.0 is also known as ITU-T Rec. G.711 Annex A [G.711-A1], as ITU-T Rec. G.711 Annex A is effectively a pointer ITU-T Rec. G.711.0. Henceforth in this document, ITU-T Rec. G.711.0 will simply be referred to as "G.711.0" and ITU-T Rec. G.711 simply as "G.711".

G.711.0 may be employed end to end; in which case the RTP payload format specification and use will be nearly identical to the G.711 RTP specification found in RFC 3550 [RFC3550]. The only significant difference other than the payload type (which will be a dynamically assigned payload type) will be the recommendation not to use Voice Activity Detection (as G.711.0 achieves its greatest compression during "VAD silence intervals"). SDP signaling elements are proposed for this use case of G.711.0 herein.

G.711.0, being both lossless and stateless, may also be employed as a LOSSLESS compression mechanism somewhere in between end systems which have negotiated use of G.711. For this case, the G.711 payloads and the corresponding G.711 RTP headers should appear to the end systems as having been transported transparently. This use case will be referred to as "G.711.0 in the Middle" and will be described in detail in Section 6 (Section 6). G.711.0, being both lossless and stateless, can be employed multiple times (e.g., on multiple,

individual hops or series of hops) of a given flow with no degradation of quality relative to end-to-end G.711. Stated another way, multiple "lossless transcodes" from/to G.711.0/G.711 do not negatively affect voice quality as may occur with lossy transcodes to/from dissimilar codecs. Since the use of G.711.0 as a compression mechanism can be used on any hop or hops of an end-to-end G.711 flow, neither Session Description Protocol (SDP) signaling elements nor G.711.0 negotiation mechanisms will be mandated in this document for this particular use case (although SDP descriptions in this document MAY be used for such G.711.0 negotiation).

Lastly, it is expected that G.711.0 will be used as an archival format for recorded G.711 streams. Therefore, a G.711.0 Storage Mode Format is also included in this document.

3.2. Key Properties of G.711.0 Design

The fundamental design of G.711.0 resulted from the desire losslessly encode and compress frames of G.711 symbols independent of what types of signals those G.711 frames contained. The primary G.711.0 use case is for G.711 encoded, zero-mean, acoustic signals (such as speech and music).

G.711.0 attributes are below:

A1 Compression for zero-mean acoustic signals: G.711.0 was designed as its primary use case for the compression of G.711 payloads which contained "speech" or other zero-mean acoustic signals. G.711.0 obtains greater than 50% average compression in service provider environments [ICASSP].

A2 Lossless for any G.711 payload: G.711.0 was designed to be lossless for any valid G.711 payload - even if the payload consisted of apparently random G.711 symbols (e.g., a modem or FAX payload). G.711.0 could be used for "aggregate 64 kbps G.711 channels" carried over IP without explicit concern if a subset of these channels happened to be carrying something other than voice or general audio. To the extent that a particular channel carried something than voice or general audio, G.711.0 ensured that it was carried losslessly, if not significantly compressed.

A3 Stateless: Compression of a frame of G.711 symbols was only to be dependent on that frame and not on any prior frame. Although greater compression is usually available by observing a longer history of past G.711 symbols, it was decided to for the compression design would be stateless to completely eliminate error propagation common in many lossy codec designs

(e.g., ITU-T Rec. G.729 [G.729], ITU-T Rec. G.722 [G.722]). That is, the decoding process need not be concerned about lost prior packets because the decompression of a given G.711.0 frame is not dependent on potentially lost prior frames. Owing to this stateless property, the frames input to the G.711.0 encoder may be changed "on-the-fly" (a 5 ms encoding could be followed by a 20 ms encoding).

A4 Self-describing: This property is defined as the ability to determine how many source G.711 samples are contained within the G.711.0 frame solely by information contained within the G.711.0 frame. Generally, the number of source G.711 symbols can be determined by decoding the initial octets of the compressed G.711.0 frame (these octets are called "prefix codes" in the standard)[ICASSP]. A G.711.0 decoder need not know what ptime is, as it is able to decompress the G.711.0 frame presented to it without signaling knowledge.

A5 Accommodate G.711 payload sizes typically used in IP: G.711 input frames of length typically found in VoIP applications represent SDP ptimes (see RFC 4566 [RFC4566]) of 5 ms, 10 ms, 20 ms, 30 ms or 40 ms. Since the dominant sampling frequency for G.711 is 8000 samples per second, G.711.0 was designed to compress G.711 input frames of 40, 80, 160, 240 or 320 samples.

A6 Bounded expansion: Since attribute A2 above requires G.711.0 to be lossless for any payload, by definition there exists at least one potential G.711 payload which must be "uncompressible". Since the quantum of compression is an octet, the minimum expansion of such an uncompressible payload was designed to be the minimum possible of one octet. Thus G.711.0 "compressed" frames can be of length one octet to X+1 octets, where X is the size of the input G.711 frame in octets. G.711.0 can therefore be viewed as a Variable Bit Rate (VBR) encoding in which the size of the G.711.0 output frame is a function of the G.711 symbols input to it.

A7 Algorithmic delay: G.711.0 was designed to have the algorithmic delay equal to the time represented by the number of samples in the G.711 input frame (i.e., no "look-ahead").

A8 Low Complexity: Less than 1.0 WMOPS average and low memory footprint (˜5k octets RAM, ˜5.7k octets ROM and ˜3.6 basic operations) [ICASSP] [G.711].

A9 Both A-law and Mu-law supported: G.711 has two operating laws, A-law and Mu-law. These two laws are also known as PCMA and PCMU in RFC 3550 [RFC3550]. The use of A-law or Mu-law should be signaled in SDP for IP applications.

These attributes generally make it trivial to compress a G.711 input frame consisting of 40, 80, 160, 240 or 320 samples. After the input frame is presented to a G.711.0 encoder, a G.711.0 "self-describing" output frame is produced. The number of samples contained within this frame is easily determined at the G.711.0 decoder by virtue of attribute A4. The G.711.0 decoder can decode the G.711.0 frame back to a G.711 frame by using only data within the G.711.0 frame.

Lastly we note that losing a G.711.0 encoded packet is identical in effect of losing a G.711 packet (when using RTP); this is because a G.711.0 payload, like the corresponding G.711 payload, is stateless. Thus, it is anticipated that existing G.711 PLC mechanisms will be employed when a G.711.0 packet is lost and an identical MOS degradation relative to G.711 loss will be achieved.

3.3. G.711 Input Frames to G.711.0 Output Frames

G.711.0 is a lossless and stateless compression of G.711 frames. The following figure depicts this where "A" is the process of G.711.0 encoding and "B" is the process of G.711.0 decoding.

1:1 Mapping from G.711 Input Frame to G.711.0 Output Frame

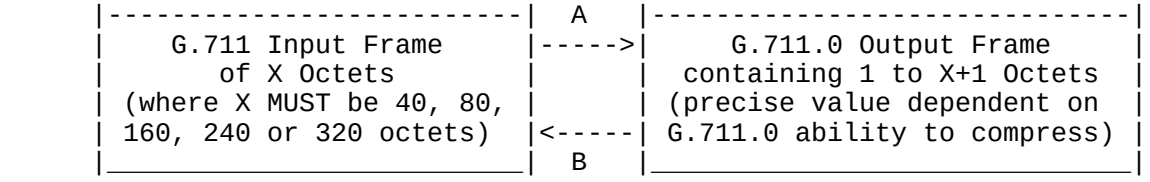

Figure 1

Note that the mapping is 1:1 (lossless) in both directions, subject to two constraints. The first constraint is that the input frame provided to the G.711.0 encoder (process "A") has a specific number of input G.711 symbols consistent with attribute A5 (40, 80, 160, 240 or 320 octets). The second constraint is that the compression law used to create the G.711 input frame (A-law or Mu-law) must be known, consistent with attribute A9.

Subject to these two constraints, the input G.711 frame is processed

by the G.711.0 encoder ("A") and produces a "self-describing" G.711.0 output frame, consistent with attribute A4. Depending on the source G.711 symbols, the G.711.0 output frame can contain anywhere from 1 to X+1 octets, where X is the number of input G.711 symbols. For virtually every use of G.711.0 resulting from zero-mean acoustic signal capture, we expect compression.

Since the G.711.0 output frame is "self-describing", a G.711.0 decoder (process "B") can losslessly reproduce the original G.711 input frame with only the knowledge of which companding law was used (A-law or Mu-law). The G.711.0 frame, being "self-describing", allows for the G.711.0 decoder ("B") to know precisely how many G.711 symbols to create.

Since G.711.0 was designed with typical G.711 payload lengths as a design constraint (attribute A5), this lossless encoding can be performed only with knowledge of the companding law being used. This information is anticipated to be signaled in SDP and will be described later in this document.

If the original inputs were known to be from a zero-mean acoustic signal coded by G.711, an intelligent G.711.0 encoder could infer the G.711 companding law in use (via G.711 input signal histogram). Likewise, an intelligent G.711.0 decoder producing G.711 from the G.711.0 frames could also infer the encoding law in use. Thus G.711.0 could be designed for use in applications that have limited stream signaling between the G.711 endpoints (i.e., they only know "G.711 at 8k sampling is being used", but nothing more). Such usage is not further described in this document. Additionally, if the original inputs were known to come from zero-mean acoustic signals, an intelligent G.711.0 encoder could tell if the G.711.0 payload had been encrypted - as the symbols would not have the distribution expected in either companding law and would appear random. Such determination is also not further discussed in this document.

It is easily seen that this process is 1:1 and that G.711.0 based lossless compression can be employed multiple times, as the original G.711 input symbols are always reproduced with 100% fidelity.

G.711.0 frames containing more source G.711 symbols compress more as a general rule, but there are exceptions. For example, an intelligent G.711.0 encoder may choose to encode 20 ms of G.711 as two individual 10 ms G.711.0 frames if a higher overall compression will result (this might occur if the first 10 ms was "silence" and two, 10 ms G.711.0 frames contained fewer octets than one 20 ms G.711.0 frame). For this reason, we will explicitly allow multiple G.711.0 encoded frames in the G.711.0 RTP payload in Section 4.2.2 (Section 3.3) below even though the usual case is anticipated to be

only one G.711.0 frame per RTP payload.

4. RTP Header and Payload

In this section we describe the precise format for G.711.0 frames carried via RTP. We begin with RTP header description relative to G.711, then provide two G.711.0 payload examples.

4.1. G.711.0 RTP Header

Relative to G.711 RTP headers, the utilization of G.711.0 does not create any special requirements with respect to the contents of the RTP packet header. The only significant difference is that the payload type (PT) RTP header field will have a value corresponding to the dynamic payload type assigned to the flow (whereas G.711 PCMU has a static PT = 0 and G.711 PCMA has a static PT = 8 [RFC3551]).

Voice Activity Detection (VAD) SHOULD NOT be used when G.711.0 is negotiated because G.711.0 obtains high compression during "VAD silence intervals" and one of the advantages of G.711.0 over G.711 with VAD is the lack of any VAD-inducing artifacts in the received signal. However, if VAD is employed, the Marker bit (M) MUST be set in the first packet of a talkspurt, that is, the first packet after a silence period which packets have not been transmitted contiguously as per rules specified in [RFC3550] for G.711 payloads.

With this introduction, the RTP packet header fields are defined as follows:

V - As per [RFC3550]

P - As per [RFC3550]

X - As per [RFC3550]

CC - As per [RFC3550]

M - As per [RFC3550]

PT- Dynamic PT assigned, consistent with MIME allocation for G711.0 defined in Media Type Definition (Section 5.1 (Section 5.1)).

SN - As per [RFC3550]

timestamp - As per [RFC3550]

SSRC - As per [RFC3550]

CSRC - As per [RFC3550]

Where V (version bits), P (padding bit), X (extension bit), CC (CSRC count), M (marker bit), PT (payload type), SN (sequence number), timestamp, SSRC (synchronizing source) and CSRC (contributing sources) are as defined in [RFC3550] and as typically used with G.711. PT (payload type) is as defined in [RFC3550].

4.2. G.711.0 RTP Payload

In this section we provide two examples for carrying G.711.0 frames in RTP payloads. The first example is used when it is desired to carry only one G.711.0 frame is in the payload. This example is a subset of the second and shown separately for clarity.

4.2.1. Single G.711.0 Frame per RTP Payload Example

This example depicts a single G.711.0 frame in the RTP payload. This is expected to be the dominant RTP payload case for G.711.0, as the G.711.0 encoding process supports the SDP packet times (ptime and maxptime, see [RFC4566]) commonly used when G.711 is transported in RTP. Additionally, as mentioned previously, larger G.711.0 frames generally compress more effectively than a multiplicity of smaller G.711.0 frames.

The following Figure illustrates the single G.711.0 frame per RTP payload case.

Single G.711.0 Frame in RTP Payload Case

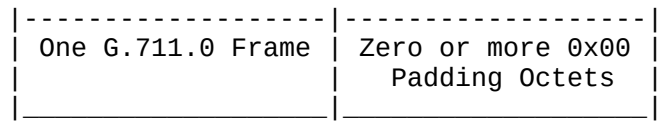

Figure 2

Encoding Process: A single G.711.0 frame is inserted into the RTP payload. The amount of time represented by the G.711 symbols compressed in the G.711.0 frame MUST correspond to the ptime signaled for applications using SDP. Although generally not desired, padding desired in the RTP payload after the G.711.0 frame MAY be created by placing one or more 0x00 octets after the G.711.0 frame. Such padding may be desired based on security considerations (see Section 11 (Section 11)).

Decoding Process: Passing the entire RTP payload to the G.711.0 decoder is sufficient for the G.711.0 decoder to create the source G.711 symbols. Any padding inserted after the G.711.0 frame (i.e., the 0x00 octets) present in the RTP payload is silently ignored by the G.711.0 decoding process. The decoding process is fully described in Section Section 4.2.3 below.

4.2.2. Multiple G.711.0 Frames per RTP Payload Example

This example depicts the case where multiple G.711.0 frames are desired in the RTP payload.

As described in Section 3.3 (Section 3.3), an "intelligent G.711.0 encoder" can decide to encode, let's say, 20 ms of G.711 symbols as two, 10 ms G.711.0 frames because a greater compression is attained for that particular 20 ms segment. Thus such "smart encoding" of such inputs is accommodated by the ability to have multiple G.711.0 frames in the RTP payload.

Note that since each G.711.0 frame is self-describing (see Attribute A4 in Section 3.2 (Section 3.2)), the individual G.711.0 frames in the RTP payload need not represent the same duration of time (i.e., a 5 ms G.711.0 frame could be followed by a 20 ms G.711.0 frame). Owing to this, the amount of time represented in the RTP payload MAY be any integer multiple of 5 ms (as 5 ms is the smallest interval of time that can be represented in a G.711.0 frame).

The following Figure illustrates the multiple G.711.0 frame per RTP payload case where the number of G.711.0 frames placed in the RTP payload is N.

Multiple G.711.0 Frames in RTP Payload Case

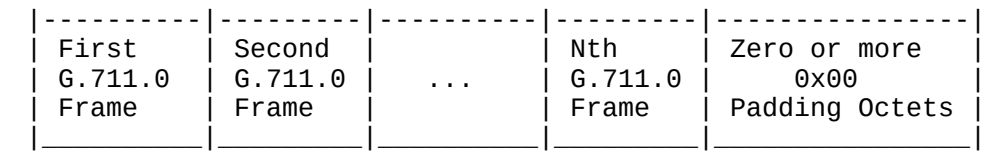

Figure 3

We note here that the individual G.711.0 frames can be, and generally are, of different lengths. The decoding process in the following section is used to determine the frame boundaries.

Encoding Process: One or more G.711.0 frames are placed in the RTP

payload simply by concatenating the G.711.0 frames together. The amount of time represented by the G.711 symbols compressed in all the G.711.0 frames in the RTP payload MUST correspond to the ptime signaled for applications using SDP. Although not generally desired, padding desired in the RTP payload SHOULD be placed after the last G.711.0 frame in the payload and MAY be created by placing one or more 0x00 octets after the last G.711.0 frame. Such padding may be desired based on security considerations (see Section 11 (Section 11)).

Decoding Process: As G.711.0 frames can be of varying length, the payload decoding process described in the following section is used to determine where the individual G.711.0 frame boundaries are.

4.2.3. G.711.0 RTP Payload Decoding Process

This decoding process is a standard part of G.711.0 bit stream decoding and is implemented in the ITU-T Rec. G.711.0 reference code.

Before describing the decoding, we note here that the largest possible G.711.0 frame is created whenever the largest number of G.711 symbols is encoded (320 from Section 3.2 (Section 3.2), property A5) and these 320 symbols are "uncompressible" by the G.711.0 encoder. In this case (via property A6 in Section 3.2 (Section 3.2)) the G.711.0 output frame will be 321 octets long. We also note that the value 0x00 chosen for the optional padding cannot be the first octet of a valid G.711.0 frame (see [G.711.0]). We also note that whenever more than one G.711.0 frame is contained in the RTP payload, the decoding of the individual G.711.0 frames will occur multiple times.

For the decoding heuristic below, let N be the number of octets in the RTP payload (i.e., excluding any RTP padding, but including any RTP payload padding), let P equal the number of RTP payload octets processed by the G.711.0 decoding process, let J represent the present G.711.0 frame being decoded, let K be the number of G.711 symbols in the output buffer, let Q be the number of octets contained in the present G.711.0 frame being processed and let "!=" represent not equal to. The keyword "STOP" is used below to indicate the end of the processing of G.711.0 frames in the RTP payload. The heuristic below assumes an output buffer for the decoded G.711 source symbols of length sufficient to accommodate the expected number of G.711 symbols and an input buffer of length 321 octets.

G.711.0 RTP Decoding Heuristic:

H1 Initialize the number of processed octets to zero (P = 0). Initialize the G.711.0 frame counter J to zero (J = 0). Initialize the counter for how many G.711 symbols are in the output buffer to zero (K = 0).

H2 Read min{320+1, (N-P)} octets into the internal buffer from the (P+1) octet of the RTP payload. We note at this point, N-P octets have yet to be processed and that 320+1 octets is the largest possible G.711.0 frame.

H3 Analyze the first octet in the internal buffer. If this octet is other than 0x00 (a padding octet), increment the G.711.0 frame counter (set J = J + 1) and continue to Step H4. Otherwise increment the processed packets counter by one (set P = P + 1). If the result of this increment results in P = N then STOP (as all RTP Payload octet have been processed), otherwise go to Step H2.

H4 Pass the internal buffer to the G.711.0 decoder. The G.711.0 decoder will read the first octet (called the "prefix code" octet in [G.711.0]) to determine the number of source G.711 samples M are contained in this G.711.0 frame.

H5 The G.711.0 decoder will produce exactly M G.711 source symbols. If J = 1, these M symbols will be the first in the output buffer and are placed at the beginning of the output buffer. If J != 1, these M symbols are concatenated with the prior symbols in the output buffer. Set K = K + M (as there are now this many G.711 source symbols in the output buffer).

H6 In process H5, the G.711.0 decoder will have consumed some number of packets, Q, in the internal buffer to produce the M G.711 symbols. Increment the number of processed octets by this quantity; that is set P = P + Q.

H7 If P < N there are more octets in the RTP payload left to process, go to Step H2. If P >= N, STOP (as all RTP payload octets have been processed).

At this point, the output buffer will contain precisely K G.711 source symbols which should correspond to the ptime signaled if SDP was used and the encoding process was without error. We also note, as an aside, that the heuristic above (and the ITU-T G.711.0 reference code) accommodates padding octets (0x00) placed anywhere in the RTP payload.

If the decoder is at a playout endpoint location, this G.711 buffer SHOULD be used in the same manner as a received G.711 payload would

have been used (passed to a playout buffer, to a PLC implementation. etc.). If not, then the instructions in Section 6 (Section 6) (G.711.0 "In The Middle") should be followed.

5. Payload Format Parameters

This section defines the parameters that may be used to configure optional features in the G.711.0 RTP transmission.

The parameters defined here as a part of the media subtype registration for the G.711.0 codec. Mapping of the parameters into Session Description Protocol (SDP) RFC 4566 [RFC4566] is also provided for those applications that use SDP.

5.1. Media Type Registration

Type name: audio

Subtype name: G7110

Required Parameters:

rate: The RTP timestamp clock rate, which is equal to the sampling rate. The typical rate is 8000, but other rates may be specified.

complaw: Indicates the companding law (A-law or mu-law) employed. The case-insensitive values are "a" or "mu".

Optional parameters:

channels: how many audio streams are represented in the G.711.0 payload - defaults to 1; stereo would be 2, etc.

[Editor's Note: We are considering specifying more than one channel for multiplexing or conference switching applications. One option for the delimiting the channels with the RTP payload would be to use one of the few not allowed G.711.0 frame prefix codes to delineate the channel data and appropriate modification to the RTP decoding heuristic in Section 4.2.3 (Section 4.2.3). Note that the channel order is already well specified in RFC 3551 [RFC3551].]

ptime, maxptime: see RFC 4566 [RFC4566]

Encoding considerations:

This media type is framed binary data (see Section 4.8 in RFC 4288 [RFC4288]) compressed as per ITU-T Rec. G.711.0.

Security considerations:

This media type does not carry active content. It does transfer compressed data. See Section 4 of RFC 4856 [RFC4856].

Interoperability considerations: none

Published specification:

ITU-T Rec. G.711.0 and RFC QQQQ.

[RFC Editor: please replace QQQQ with a reference to this RFC]

Applications that use this media type:

Audio and video streaming and conferencing tools.

Additional information: none

Person & email address to contact for further information:

Michael Ramalho <mramalho@cisco.com> or <mar42@cornell.edu>

Intended usage: COMMON

Restrictions on usage:

This media type depends on RTP framing, and hence is only defined for transfer via RTP [RFC3550]. Transport within other framing protocols is not defined at this time.

Author: Michael Ramalho

Change controller:

IETF Audio/Video Transport working group delegated from the IESG.

5.2. Mapping to SDP Parameters

The information carried in the media type specification has a specific mapping to fields in the Session Description Protocol (SDP), which is commonly used to describe RTP sessions. When SDP is used to specify sessions employing G.711.0, the mapping is as follows:

- The media type ("audio") goes in SDP "m=" as the media name.
- The media subtype ("G7110") goes in SDP "a=rtpmap" as the encoding name.

- The required parameter "rate" also goes in "a=rtpmap" as the clock rate.
- The parameters "ptime" and "maxptime" go in the SDP "a=ptime" and "a=maxptime" attributes, respectively.
- Remaining parameters go in the SDP "a=fmtp" attribute by copying them directly from the media type string as a semicolon-separated list of parameter=value pairs.

5.3. Offer/Answer Considerations

There are no special considerations when using the SDP offer/answer RFC 3264 [RFC3264] as all the SDP parameters are declaritive.

[EDITOR'S NOTE: This may change in a future revision if the channel parameter is negotiated. This could happen when the offer desires channels = N and the answerer can only support a number less than N. We would then insert an offer/answer example in a future revision of this draft.]

5.4. SDP Example

The following examples illustrate how to signal G.711.0 via SDP:

```
m=audio RTP/AVP 98

a=rtpmap:  98 G7110/8000

a=prime:  20

a=fmtp:98 complaw = mu
```

In the above example, the dynamic payload type 98 is mapped to G.711.0 via the "a=rtpmap" parameter. The packetization time (ptime) is indicated to be 20 ms of audio. The mandatory "complaw" is on the "a=fmtp" parameter line.

6. G.711.0 "In The Middle"

When G.711 has been negotiated end-to-end, G.711.0 compression can be employed by entities in the middle of the end-to-end G.711 flow as a compression mechanism. When used in this manner, it can be used with or without compression of the RTP header. In either case, the G.711 payloads AND the corresponding G.711 RTP headers MUST appear to the end systems as having been transported transparently.

6.1. G.711.0 "In The Middle" - No RTP Header Compression

This figure below illustrates how the compression could be accomplished without the RTP header compression.

G.711.0 Compression "In The Middle" - No RTP Header Compression Case

Figure 4

Figure 4 depicts G.711.0 compression in Box C and G.711.0 decompression back to G.711 in Box E. It depicts the case where only compression of the G.711 payload is desired; the RTP header (including any extensions) is simply copied with the exception that the G.711 payload type (the usual static PT of 0 or 8 is shown) is

replaced by a PT negotiated between Box C and Box E (depicted as PT = Q).

Note that if there are no hops between Box C and E (i.e, no Box D), this is equivalent to compression over a single link. The compression segment represented by Box C, Box E and Box F is labeled a "G.711.0 compression segment" in the above figure.

Since G.711.0 is a lossless and stateless compression, there can be multiple such segments between the sending and receiving endpoints (not shown).

The G.711.0 compression and decompression (Box C and E) may reside in a variety of network elements such as, but not limited to, switches, routers, middleboxes (NATs/PATs, firewalls, session border controllers, transport acceleration devices) and is purposely not specified here.

There may be many "potential G.711.0 compression/decompression points" along the end-to-end G.711 flow; the mechanisms by which certain entities determine that they should perform G.711.0-based compression and decompression are outside the scope of this document.

The method by which G.711.0 compression segment endpoints negotiate which RTP payload type (Q shown above) is to be used is outside the scope of this document, although the SDP elements described herein MAY be used.

Firewalls, NATs, SBCs, etc. that may exist in the path of the G.711.0 packets (Box D) and who may drop packets of unexpected payload types may need additional configuration and/or intelligence to let the compressed G.711.0 packets through. Mechanisms to do this are also outside the scope of this document.

[EDITOR'S NOTE: Because many boxes of this type inspect signaling to determine which RTP packets are allowed to progress, we are considering a hint to be place in the G.711 (not G.711.0) SDP that says, in essence, if you do compress this packet, please use PT = Q. In this way middleboxes may also know to pass PT = Q packets as well as PT = [0 | 8]. Such a G.711 SDP entry may look like:

Example of G.711 SDP with hint for G.711.0 PT

```
m=audio RTP/AVP 0
a=rtpmap: 0 PCMU
a=fmtp:0 G7110 = Q   <<< the G.711 SDP hint
```

Here, the last line provides the hint that this G.711 Mu-law flow may be represented anywhere on the end-to-end G.711 flow as PT = 0 (uncompressed) or PT = Q (compressed) RTP packets. Then candidate compression entities MAY choose to honor that hint in their respective dynamic payload type negotiation for G.711.0. Of course, this hint would require the registration of an optional parameter for the PCMU and PCMA media registrations. We hereby solicit feedback on this concept.]

6.2. G.711.0 "In The Middle" - With RTP Header Compression

When it is desired to compress the G.711 header as well, the G.711.0 compression segment endpoints of the previous section have further functionality by which they also compress the headers. However, this functionality is outside of the scope of this document.

We simply note here that if such functionality is employed, that the G.711 payloads AND the corresponding G.711 RTP headers MUST appear to the end systems as having been transported transparently. Such RTP header compression functionality SHOULD be stateless so as to minimize error propagation for lost packets to be consistent with G.711.0 design goal attribute A3.

6.3. G.711.0 "In The Middle" - Implications for Voice Quality and Added Delay

As described in the sections immediately previous, G.711.0 can be employed multiple times (e.g., on multiple, individual hops or series of hops) of a given G.711 flow. Owing to the stateless design of G.711.0 and any RTP header compression scheme recommended above, there is no error propagation owing to loss of a G.711.0 packet. Thus the impact of an individual packet drop of a G.711.0 RTP packet is identical to the impact of the corresponding equivalent G.711 RTP packet.

Stated another way, multiple "lossless transcodes" from/to G.711.0/ G.711 do not negatively affect voice quality as may occur with lossy transcodes to/from dissimilar codecs.

G.711.0 provides over 50% reduction in average payload size with exactly 0.0000% quality loss relative to G.711 [ICASSP].

For completeness, we note that a G.711.0 encode/decode average complexity is 1 WMOPS (see Section 3.2 (Section 3.2), attribute A8). Given such low complexity, less than 1 ms of compression/ decompression delay per each G.711.0 compression segment is expected in most implementations.

6.4. G.711.0 "In The Middle" - Multiplexing Multiple G.711 Flows

It may also be desired to multiplex the payloads of many G.711 channels into one "G.711.0 payload". As with the previous section, functionality for the mechanism used is outside the scope of this document.

[EDITOR'S NOTE: If we allow the channels parameter to be >1, then a mechanism for multiplexing the G.711.0 payloads into one RTP payload would be specified in this document. However, we don't know of a standardized use case for multiplexing a multiplicity of media streams representing multiple endpoint flows in one RTP media flow. This would be advantageous if, for example, multiple G.711 flows traverse two known endpoints and where the endpoints could "add and delete" G.711 flows "on-the-fly" (perhaps by association of the RTP CSRCs) and then fake out the endpoints with simulated RTP headers in a manner similar to WAN optimization product does today for TCP. Comments are welcome on this point.]

If such RTP multiplexing compression functionality is designed, the RTP header compression used with the G.711.0 multiplexing SHOULD be stateless so as to minimize error propagation for lost packets to be consistent with G.711.0 design goal attribute A3.

7. G.711.0 Storage Mode

There are two storage modes defined for the G.711.0; one for short recordings and one for long recordings.

For short recordings, the recommendation will be similar to many other IETF codecs (e.g., iLBC, EVRC-NW) and will fundamentally be a concatenation of received G.711.0 frames and erasure frames.

However, since G.711.0 has variable length frames, it is prudent for long recordings (over many minutes long) to use an indexing methodology that enables playout far into the recording without decoding all the G.711.0 frames since the recording began.

For either storage mode, a "G.711.0 erasure frame" is defined in the following section because ITU-T Rec. G.711.0 [G.711.0] does not define one.

7.1. G.711.0 Erasure Frame

A G.711.0 erasure frame is a representation of the amount of time in lost or not received G.711.0 frames. Lost frames are typically determined by unexpected discontinuities observed in the RTP send timestamps at the receiver.

A G.711.0 compressed frame can represent 40, 80, 160, 240 or 320 G.711 symbols (Attribute A5 of Section 3.2 (Section 3.2)). As G.711.0 also does not use the value of 0x01 for the first octet of any valid G.711.0 frame, it can be used as an identifier for an erasure frame. Thus an erasure frame is the two octet quantity shown below.

The Two Octet G.711.0 Erasure Frame Definition

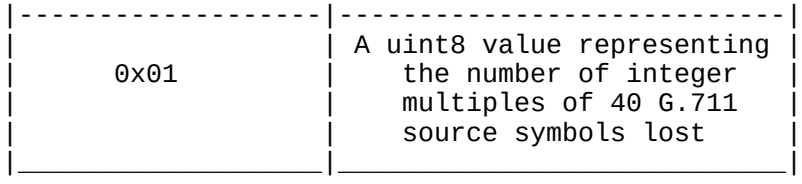

Figure 5

The value of the second octet is representative of how many G.711 source samples were lost. For example, a value of 4 implies 160 samples were lost independently of how many G.711.0 frames those 160

samples were represented by. Erasure frames themselves may be concatenated if it is desired to create one per G.711.0 frame, or one per G.711.0 RTP payload, etc. This erasure frame format can therefore represent 255*40 = 10,200 missing G.711 symbols (i.e., about 1.2 seconds of G.711 at 8000 samples per second).

The erasure frame has been designed to represent precisely what was observed at the receiver for true archival purposes (including potentially law enforcement). Therefore, the use of erasure frames for missing G.711.0 frames is RECOMMENDED for archival purposes.

If such an archival needs are not required, a storage implementation MAY provide packet loss concealment (PLC) or simply "silence" (G.711.0 analog zero representation) for the missing source G.711 data and choose not to use an erasure frame. Such PLC/silence insertion MUST represent the precise amount of time represented by the missing data to maintain synchronization with the original media.

7.2. G.711.0 Storage Mode - Short Recordings

This short recording storage format is used for storing G.711.0 encoded frames. The file begins with a magic number to identify the coder that is used. The magic number for G.711.0 A-law corresponds to the ASCII character string "#!G7110A\n", i.e., "0x23 0x21 0x47 0x37 0x31 0x31 0x30 0x41 0x0A". Likewise, the magic number for G.711.0 MU-law corresponds to the ASCII character string "#!G7110M\n", i.e., "0x23 0x21 0x47 0x37 0x31 0x31 0x4E 0x4D 0x0A". The codec data frames including any necessary erasure frames are stored in consecutive order concatenated together as shown in Section 4.2.2 (Section 4.2.2).

To decode the individual G.711.0 frames, a heuristic similar to the one presented in Section 4.2.2 (Section 4.2.2) modified appropriately to recognize and process erasure frames as legitimate G.711.0 frames in the recording format.

7.3. G.711.0 Storage Mode - Long Recordings

[EDITOR'S NOTE: The long recordings storage mode format is TBD. This storage mode will likely have indexing capability and metadata capabilities.]

8. Acknowledgements

There have been many people contributing to G.711.0 in the course of its development. The people listed here deserve special mention: Takehiro Moriya, Claude Lamblin, Herve Taddei, Simao Campos, Yusuke Hiwasaki, Jacek Stachurski, Lorin Netsch, Paul Coverdale, Patrick Luthi, Paul Barrett, Jari Hagqvist, Pengjun (Jeff) Huang, and Jon Gibbs.

9. Contributors

The authors thank everyone who have comtributed to this document. The people listed here deserve special mention: Ali Begen and Roni Even.

10. IANA Considerations

One media type (audio/G7110) has been defined and requires IANA registration in the media types registry. See Section 5.1 (Section 5.1)

11. 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 appropriate RTP profile (for example RFC 3551 [RFC3551] or [RFC4585]. This implies that confidentiality of the media streams is achieved by encryption; for example, through the application of SRTP [RFC3711]. Because the data compression used with this payload format is applied end-to-end, any encryption needs to be performed after compression.

Note that the appropriate mechanism to ensure confidentiality and integrity of RTP packets and their payloads is very dependent on the application and on the transport and signaling protocols employed. Thus, although SRTP is given as an example above, other possible choices exist.

Note that end-to-end security with either authentication, integrity or confidentiality protection will prevent a network element not within the security context from performing media-aware operations other than discarding complete packets. To allow any (media-aware) intermediate network element to perform its operations, it is required to be a trusted entity which is included in the security context establishment.

G.711.0 has no known denial-of-service attacks due to decoding, as data posing as a desired G711.0 payload will be decoded into something (as per the decoding algorithm) with a finite amount of computation. This is due to the decompression algorithm having a finite worst-case processing path (no infinite computational loops are possible).

G.711.0 is a variable bit rate (VBR) audio codec. There have been recent concerns with VBR speech codecs where a passive observer can identify phrases from a standard speech corpus by means of the lengths produced by the encoder even when the payload is encrypted [IEEE]. In this paper, it was determined that some code excited linear prediction (CELP) codecs would produce discrete packet lengths for some phonemes. And furthermore with the use of appropriately designed Hidden Markov Models (HMMs) that such a system could predict phrases with unexpected accuracy. One CELP codec studied, SPEEX, had the property that it produced 21 different packet lengths in its wideband mode and that these packet lengths probabilistically mapped to phonemes that a HMM system could be trained on. In this paper it was determined that a mitigation technique would be to pad the output of the encoder with random padding lengths to the effect: 1) that more discrete payload sizes would result, and 2) that the probabilistic mapping to phonemes would become less clear. As G.711

is not a speech model based codec, neither is G.711.0. A G.711.0 encoding, during talking periods, produces frames of varying frame lengths which are not likely to have a strong mapping to phonemes. Thus G.711.0 is not expected to have this same vulnerability. It should be noted that "silence" (only one value of G.711 in the entire G.711 input frame)" or "near silence" (only a few G.711 values) is easily detectable as G.711.0 frame lengths or one or a few octets. If one desires to mitigate for silence/non-silence detection, statistically variable padding should be added to G.711.0 frames that resulted in very small G.711.0 frames (less than about 20% of the symbols of the corresponding G.711 input frame). Methods of introducing padding in the G.711.0 payloads have been provided in the G.711.0 RTP payload definitions in Sections 4.2.1 (Section 4.2.1) and 4.2.2 (Section 4.2.2).

12. References

12.1. Normative References

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

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

[RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

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

[RFC4856] Casner, S., "Media Type Registration of Payload Formats in the RTP Profile for Audio and Video Conferences", RFC 4856, February 2007.

[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.

[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.

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

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[G.711.0] ITU-T G.711.0, "Recommendation ITU-T G.711.0 - Lossless Compression of G.711 Pulse Code Modulation", September 2009.

[G.711] ITU-T G.711.0, "Recommendation ITU-T G.711 - Pulse Code Modulation (PCM) of Voice Frequencies", November 1988.

[G.711-A1] ITU-T G.711 Amendment 1, "Recommendation ITU-T G.711 Amendment 1 - Amendment 1: New Annex A on Lossless Encoding of PCM Frames", September 2009.

12.2. Informative References

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[G.729] ITU-T G.729, "Recommendation ITU-T G.729 - Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP)", January 2007.

[G.722] ITU-T G.722, "Recommendation ITU-T G.722 - 7 kHz audio-coding within 64 kbit/s", November 1988.

[ICASSP] N. Harada, Y. Yamamoto, T. Moriya, Y. Hiwasaki, M. A. Ramalho, L. Netsch, Y. Stachurski, Miao Lei, H. Taddei, and Q. Fengyan, "Emerging ITU-T Standard G.711.0 - Lossless Compression of G.711 Pulse Code Modulation, International Conference on Acoustics Speech and Signal Processing (ICASSP), 2010, ISBN 978-1-4244-4244-4295-9", March 2010.

[IEEE] C.V. Wright, L. Ballard, S.E. Coull, F. Monrose, and G.M. Masson, "Spot Me if You Can: Uncovering Spoken Phrases in Encrypted VoIP Conversations, IEEE Symposium on Security and Privacy, 2008, ISBN: 978-0-7695-3168-7", May 2008.

Authors' Addresses

Michael A. Ramalho (editor)
Cisco Systems, Inc.
4563 Tuscana Drive
Sarasota, FL 34241
USA

Phone: +1 919 476 2038
Email: mramalho@cisco.com

Paul E. Jones
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com

Noboru Harada
NTT Communications Science Labs
3-1 Morinosato-Wakamiya
Atsugi, Kanagawa 243-0198
JAPAN

Email: harada.noboru@lab.ntt.co.jp

Muthu Arul Mozhi Perumal
Cisco Systems, Inc.
Cessna Business Park
Sarjapur-Marathahalli Outer Ring Road
Bangalore, Karnataka 560103
India

Phone: +91 9449288768
Email: mpermumal@cisco.com

Miao Lei
Huawei Technologies Co. Ltd
No. 3 Xinxi Rd.
ShangDi, HaiDian District
Beijing, Beijing 100085
Beijing

Phone: +86 1082882759
Email: lei.miao@huawei.com