Internet Engineering Task Force
INTERNET DRAFT
File:draft-ietf-avt-rtp-mpeg2aac-02.txt
Kretschmer-AT&T/Basso-AT&T
Civanlar-AT&T/Quackenbush-AT&T
Snyder-AT&T
July 20, 2001
Expires: January 20, 2002

RTP Payload Format for MPEG-2 and MPEG-4 AAC Streams

STATUS OF THIS MEMO

Abstract

This document describes a payload format for transporting MPEG-2/MPEG-4 AAC encoded data using RTP. MPEG-2/MPEG-4 AAC is a recent standard from ISO/IEC [1] [2] [3] for coding multi-channel audio data. This payload format increases the packet loss resilience of AAC coded audio transport above that of 'RTP Payload Format for MPEG1/MPEG2 Video (RFC 2250)' [7] by incorporating AAC properties into the payload format. Supported features comprise fragmentation, interleaving, grouping, repair information and a predictability vector. The MPEG-2/MPEG-4 AAC bitstream format is not backwards compatible with other MPEG-2 audio formats (e.g. MP3). Several services provided by RTP are beneficial for MPEG-2/MPEG-4 AAC encoded data transport over the Internet. Additionally, the use of RTP allows for the synchronization of MPEG-2/MPEG-4 AAC with other real-time streams.

In this version of the draft:

- The fragmentation section has been revised to allow for the fragmentation of elements (CPE, etc.) to better support transmission over small MTU links

1. Introduction

The ISO/IEC MPEG-2/MPEG-4 Advanced Audio Coding (AAC) [1] [2] [3] technology delivers CD-like or better multichannel audio quality at rates around 64 kBit/s per channel. It has a flexible bitstream syntax that supports from 1 to 48 audio channels, up to 16 subwoofer channels and up to 16 embedded data channels. AAC supports a wide range of sampling frequencies (from 16 kHz to 96 kHz) and an extremely wide range of bitrates. AAC can support applications ranging from professional or home theater sound systems to Internet music broadcast systems.

The syntax of MPEG-2 AAC compressed data streams is identical to that of MPEG-4 AAC main, AAC LC and AAC SSR General Audio compressed data streams, so that MPEG-2 AAC is fully forward compatible with MPEG-4 AAC. Both MPEG-2 AAC and MPEG-4 AAC provide the same level of compression performance. However, the semantics of MPEG-4 AAC is different in one small respect, precluding a full backward compatibility of MPEG-4 AAC to MPEG-2 AAC.

Benefits of using RTP for MPEG-2/MPEG-4 AAC stream transport include:

i. Providing increased packet loss resilience based on application layer framing.

ii. The ability to synchronize AAC streams with other RTP payloads

iii. Monitoring MPEG-2/MPEG-4 AAC delivery performance through RTCP

iv. Combining MPEG-2/MPEG-4 AAC and other real-time data streams received from multiple end-systems into a set of consolidated streams through RTP mixers

v. Converting data types, etc. through the use of RTP translators.

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 [5].

1.1 Overview of MPEG-2/MPEG-4 AAC

AAC combines the coding efficiencies of a high resolution filter bank, a powerful model of audio perception, backward-adaptive prediction, joint channel coding, and Huffman coding to achieve high-quality signal compression. In 1998 the MPEG Audio subgroup tested the family of MPEG audio coders (see http://www.tnt.uni-hannover.de/project/mpeg/audio/public/w2006.pdf). The test results indicate that for a stereo signal, AAC at 96 kBit/s has audio quality comparable to MPEG-2 Layer 3 ("mp3") at 128 kBit/s.

AAC is a block oriented, variable rate coding algorithm. An AAC encoder takes 1024 samples per channel at a time (a 'block') as input and the compressed representation is variable in size.

Rate control can be used at the encoder to generate a constant-rate bitstream. Each block of AAC compressed bits is called a "raw data block", and can be decoded "stand-alone", that is, without information from prior raw data blocks. This feature is particularly useful for the delivery of AAC over lossy packet networks since the loss of a packet does not directly affect the decodability of the adjacent packets.

1.2 Bitstream Syntax

The syntax of an AAC bitstream is as follows:

```
<bitstream>        => <raw_data_block><bitstream>
<raw_data_block>   => [<element>]<END><PAD>
```

where <bitstream> indicates the AAC bitstream, <lowercase> indicates intermediate tokens, <UPPERCASE> indicates terminal tokens and [] indicates one or more occurrence. <END> is a token that indicates the end of a raw_data_block and <PAD> is a variable length token that forces the total length of a raw_data_block to be an integral number of bytes. In general, intermediate tokens are not an integral number of bytes in length.

The <element> tokens are a string of bits of variable length, and they can be any of the following:

```
<single_channel_element>     a single audio channel
<channel_pair_element>       a stereo presentation (2 channels)
<coupling_channel_element>   a mechanism for multi-channel compression
<lfe_channel_element>        a special effects channel
<data_stream_element>        "user data"
<program_config_element>     a mechanism for describing the bitstream
                             content
<fill_element>               a mechanism to use bits (for constant rate
                             channels)
```

The <elements> can occur several times in a single raw_data_block. For example, the raw_data_block for a 5.1 surround sound signal would be:

```
<single_channel_element><channel_pair_element>...
<channel_pair_element><lfe_channel_element><END>
```

corresponding to the center, left and right, left surround and right surround and effects channels. Occurances of the <channel_pair_element> are dis-ambiguated by means of a unique 4-bit id inside the <channel_pair_element>.

2. Issues covered by this Payload Format

2.1 Repair Information to reconstruct lost AAC Frames (Unequal FEC)

A smart AAC decoder can mitigate the effects of lost packets using techniques such as interpolation in the spectral domain. However if the raw_data_block in a packet is perceptually significant and also highly unpredictable (e.g. the onset of a cymbal crash) then the sender may choose to add repair information associated with that raw_data_block. This form of unequal FEC allows the encoder/sender to protect a stream depending on known loss characteristics and/or frame predictability. A given repair information block (AAC DATA chunk with TYPE > 0) is typically associated with a raw_data_block (TYPE = 0). The association between the raw_data_block and the repair information is obtained by means of the SEQ field.

Repair Information as defined here is a valid AAC raw_data_block. As an example, the Repair Information can be a highly compressed monophonic version of a subset of the signal being transmitted. An AAC stereo signal coded at a sampling rate of 44100 samples/s and a bit rate of 96 kBit/s corresponds to an average raw_data_block size of 279 bytes. A RepairData version of that block, compressed to 16 kBit/s would be 46 bytes in length. Given that perceptually critical blocks might occur only once per 100 or more blocks, the average rate increase associated with this type of RepairData can be very low. Generally, the Repair Information for a given AAC frame X SHALL be carried by a different RTP packet then the one that carries **X. Generally, the Repair Information MUST be computed at the same** sampling rate as the stream being repaired.

The usage of the Repair Information is similar to the one proposed in [6]. The OPTIONAL Repair Information MAY be provided for every frame. RepairData can be generated in many ways including using two encoders, decoding followed by coding or processing the original bitstream.

2.2 Fragmentation of AAC Frames

It is desirable to limit the size of an AAC frame to less than the path-MTU. If this is not possible, the frame can be fragmented across several RTP packets. Fragmentation SHOULD occur at <element> boundaries. If further fragmentation is needed <elements> MAY be fragmented, as well. In that case the decoder must be able to handle partial <elements>.

An RTP packet contains either an integer number of complete AAC frames or fragments of a single AAC frame. Subsequent RTP packets containing subsequent fragments of an AAC frame have a much simpler header that is just two bytes long. They can be identified by the F-bit set to **1. The S-bit signals the first or only fragment of an <element>. The** same ELEMENT ID is shared by all fragments belonging to the same <element>. ELEMENT ID zero is assigned to the fragments of the first <element> in the frame and is increased by one for each following <element>. FBITS indicates the number of unused bits in the first byte and LBITS the number of unused bits in the last byte. FBITS and/or LBITS must be set to zero if the fragment starts and/or ends at a byte-aligned boundary. The length of a fragment can be determined from the total length of the packet excluding the headers.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|X|X|F|S|ELEMNT ID|FBITS|LBITS|AAC FRAGMENT                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.3 Predictability of AAC Frames

AAC frame predictability allows adaptive handling of packet losses and/or bandwidth constraints by signalling the need for an action a receiver may take when an associated AAC frame is lost. Every AAC frame will be assigned to one of the following three predictability classes:

- 0: not predictable
- 1: one side predictable (either L-predictable or R-predictable)
- 2: two side predictable

(- 3: reserved)

An AAC frame that belongs to class 0 cannot be easily concealed using any other AAC frame(s) in the bitstream.

An AAC frame that belongs to class 1 can be predicted either from previous (R-predictable) or following (L-predictable) AAC frame but not from both.

An AAC frame that belongs to class 2 can be predicted from the preceding or following AAC frame or from both.

Predictability information is coded for every RTP AAC packet in the Predictability Quantifier (PQ) which is 2 bits in length. For a given RTP packet such PQs are organized in a predictability vector which represents a sliding window of PQs, starting with the current packet's PQ followed by preceding packets' PQs.

2.4 Grouping and Interleaving of AAC Frames

It is often desirable to group an integer number of AAC frames. The predictability of such an RTP packet is the predictability of the AAC frame in the RTP packet which is least predictable. AAC frames belonging to the same predictability class MAY be grouped into one RTP packet. Note that if frames of different predictabilities are grouped much of the usefulness of the predictability information is lost. The sequence numbers SEQ of the AAC DATA chunks are used to restore the proper order on the receiver side.

Grouping AAC frames into a single RTP packet is OPTIONAL.

2.5 Example RTP Packet Sequence

The example below shows a sequence of AAC frames (a...p) and their assigned predictability classes.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 2 | 2 | 2 | 1 | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 1 | 0 | 1 | 2 | 2 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The AAC frames MAY be grouped according to their predictability. R(x) is the RepairData information sent within the RTP packet:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|a g j|b h k|c i o| d f |  e  | l n |  m  |  p  |           |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|     |     |R(e) |     |     |R(m) |     |     |     |     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3. RTP AAC Payload Format

The AAC specific RTP payload consists of a 8, 32, 64 or 96 bit header, and a variable number of AAC DATA chunks. The type of those chunks is identified by TYPE field. The LENGTH field specifies the length of a chunk in bytes and SEQ is a sequence number which allows grouping, interleaving and association of Repair Info with the frame it repairs.

The header contains a vector of Predictability Quantizers (PQ) which specify the packets' predictability classes, and a set of control bits.

The PVS field specifies if the header contains 12, 28 or 44 PQs. At the beginning of a session, if fewer packets have been transmitted/ received than there are PQs in the header then the extra PQs are invalid and MUST be set to 0 (on the sender side) and MUST be ignored (on the receiver side).

If a sender provides a predictability vector but does not provide frame predictability information it MUST set all PQs to 0. A client can ignore the information provided by PQs since PQs are not required for decoding AAC frames. PQs can be used to decide when to ask for retransmission of lost packets. PQs can also provide hints which help a PQ-aware decoder to improve the audio quality when concealing lost packets.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PVS|M|F|  MBZ  |PRD VECTOR (PVS > 0)                           | Header
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PRD VECTOR (PVS > 1)                                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PRD VECTOR (PVS > 2)                                           |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|TYPE   |SEQ                  |LENGTH (if M==1)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
|AAC DATA 1                                                     | AAC
|                                                               | Data
|                                                               |
|               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |TYPE   |SEQ                  |LENGTH           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
|               |AAC DATA 2                                     |
|-+-+-+-+-+-+-+-+                                               |
|                               .                               |
|                               .                               |
|                               .                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
|TYPE   |SEQ                  |LENGTH                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
|AAC DATA N                                                     |
|                                                               |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PRD VECTOR: Predictability vector. It contains either 12, 28 or 44 Predictability Quantifiers (PQ). The size of a PQ element is 2 bits. The first PQ refers to the predictability class of the current packet. The following PQs refer to the most recent previous packets. Thus, the vector looks like this: {PQ(t), PQ(t-1), PQ(t-2)...}
The predictability class of a packet is that of the least predictable AAC frame that is contained in the packet.

PVS: Predictability Vector Size. Specifies the number of 32bit words used for the Predictability Vector. The first 32bit word contains the flags field. Hence, only the lower significant 24bits belong to the vector. If PVS is set to 0 the predictability vector field does not exist, and the TYPE field is contiguous with the MBZ field.

M: If M is set, then the payload contains more than one AAC frame. Hence, a LENGTH field for the first frame MUST be present. If M is clear, then only one AAC frame is present, and no LENGTH field is present.

F: Fragmented Frame.

MBZ: Must be set to 0.

TYPE: The type of AAC DATA. This field specifies if the AAC DATA is an original AAC frame or contains some form of Error Correction data. The following types are defined for now:
0: Original AAC frame
1: Identical to original frame but sent for redundancy
2: Same AAC configuration but encoded using less bits (Repair Information)
(all three types are valid AAC frames)

SEQ: The sequence number enumerates AAC frames at the stream level. It may be used to support interleaving or association of Repair Information with TYPE==0 AAC frames, etc.

LENGTH: The length of the AAC Data in bytes.

AAC DATA: The actual AAC data chunk. This is either a valid AAC frame (TYPE = 0) or Repair Information belonging to a valid AAC frame (TYPE > 0).

3.1 RTP Header Fields Usage:

The RTP header fields are used as follows:

Payload Type (PT): It is expected that the RTP profile for a particular class of applications will assign a payload type for this encoding, or alternatively a payload type in the dynamic range shall be chosen.

Marker (M) bit: Set to one to mark the last fragment (or only fragment) of an AAC frame.

Extension (X) bit: Defined by the RTP profile used.

Timestamp (TS): 32-bit timestamp representing the sampling time of the first sample of the first AAC frame in the packet. The clock frequency MUST be set to the sample rate of the encoded audio data and is conveyed out-of-band (i.e. through SDP [8]). If N > 1 frames are present in a RTP packet the TS of the frames 2...N can be calculated by computing the sequence number difference between those frames and the first frame since the sample rate and the number of samples per frame are fixed and known. All packets that make up a fragmented AAC frame MUST use the same TS. Timestamps start at a random value to improve security.

SSRC: set as described in RFC1889 [2].

CC and CSRC fields are used as described in RFC 1889 [2].

RTCP SHOULD be used as defined in RFC 1889 [2]

4. Security Considerations

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [2]. This implies that confidentiality of the media streams is achieved by encryption. Because the data compression used with this payload format is applied end-to-end, encryption may be performed on the compressed data so there is no conflict between the two operations.

This payload type does not exhibit any significant non-uniformity in the receiver side computational complexity for packet processing to cause a potential denial-of-service threat.

5. Intellectual Property Disclosure

A US patent application has been filed on the usage and computation of predictability information for transmission over lossy channels.

6. References

[1] ISO/IEC 13818-7 Advanced Audio Coding (AAC).

[2] ISO/IEC 14496-3:1999 "Information technology - Coding of audio-visual objects - Part 3: Audio," December, 1999.

[3] ISO/IEC 14496-3:1999 / AMD1:2000.

[4] Schulzrinne, Casner, Frederick, Jacobson RTP: A Transport Protocol for Real Time Applications RFC 1889, Internet Engineering Task Force, January 1996.

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

[6] Perkins C., Kouvelas I., Hodson O., Hardman V., Bolot J.C, Vega-Garcia A., Fosse-Parisis S. "RTP Payload for Redundant Audio Data", RFC 2198, Internet Engineering Task Force, September 1997.

[7] D. Hoffman, G. Fernando, V. Goyal, M. Civanlar RTP Payload Format for MPEG1/MPEG2 Video RFC 2250, Internet Engineering Task Force, January 1998.

[8] M. Handley, V. Jacobson, SDP: Session Description Protocol

RFC 2327, Internet Engineering Task Force, April 1998.

7. Authors' Addresses

Mathias Kretschmer
AT&T Labs - Research
180 Park Ave.
Florham Park, NJ 07932
USA
e-mail: mathias@research.att.com

Andrea Basso
AT&T Labs - Research
100 Schultz Drive
Red Bank, NJ 07701
USA
e-mail: basso@research.att.com

M. Reha Civanlar
AT&T Labs - Research
100 Schultz Drive
Red Bank, NJ 07701
USA
e-mail: civanlar@research.att.com

Schuyler R. Quackenbush
AT&T Labs - Research
180 Park Ave.
Florham Park, NJ 07932
USA
e-mail: srq@research.att.com

James H. Snyder
AT&T Labs - Research
180 Park Ave.
Florham Park, NJ 07932
USA
e-mail: jhs@research.att.com