Colin Perkins
Isidor Kouvelas
Orion Hodson
Vicky Hardman
University College London

Mark Handley
ISI

Jean-Chrysostome Bolot
Andres Vega-Garcia
Sacha Fosse-Parisis
INRIA Sophia Antipolis

# RTP Payload for Redundant Audio Data

draft-perkins-rtp-redundancy-00.ps

## Status of this Memo



Comments are solicited and should be addressed to the authors and/or the AVT working group's mailing list at rem-conf@es.net.

**Abstract**

This document describes a payload format for use with the real-time transport protocol (RTP), version 2, for encoding redundant audio data. The primary motivation for the scheme described herein is the development of audio conferencing tools for use with lossy packet networks such as the Internet Mbone, although this scheme is not limited to such applications.

# 1 Introduction

If multimedia conferencing is to become widely used by the Internet Mbone community, users must perceive the quality to be sufficiently good for most applications. We have identified a number of problems which impair the quality of conferences, the most significant of which is packet loss. This is a persistent problem, particularly given the increasing popularity, and therefore increasing load, of the Internet. The disruption of speech intelligibility even at low loss rates which is currently experienced may convince a whole generation of users that multimedia conferencing over the Internet is not viable. The addition of redundancy to the data stream is offered as a solution [1]. If a packet is lost then the missing information may be reconstructed at the receiver from the redundant data that arrives in the following packet(s), provided that the average number of consecutively lost packets is small. Recent work [4,5] shows that packet loss patterns in the Internet are such that this scheme typically functions well.

This document describes an RTP payload format for the transmission of audio data encoded in such a redundant fashion. Section 2 presents the requirements and motivation leading to the definition of this payload format, and does not form part of the payload format definition. Sections 3 onwards define the RTP payload format for redundant audio data.

# 2 Requirements/Motivation

The requirements for a redundant encoding scheme under RTP are as follows:

- Packets have to carry a primary encoding and one or more redundant encodings.
- As a multitude of encodings may be used for redundant information, each block of redundant encoding has to have an encoding type identifier.
- As the use of variable size encodings is desirable, each encoded block in the packet has to have a length indicator.
- The RTP header provides a timestamp field that corresponds to the time of creation of the encoded data. When redundant encodings are used this timestamp field can refer to the time of creation of the primary encoding data. Redundant blocks of data will correspond to different time intervals than the primary data, and hence each block of redundant encoding will require its own timestamp. To reduce the number of bytes needed to carry the timestamp, it can be encoded as the difference of the timestamp for the redundant encoding and the timestamp of the primary.

There are two essential means by which redundant audio may be added to the standard RTP specification: a header extension may hold the redundancy, or one, or more, additional payload types may be defined.

Including all the redundancy information for a packet in a header extension would make it easy for applications that do not implement redundancy to discard it and just process the primary encoding data. There are, however, a number of disadvantages with this scheme:

- There is a large overhead from the number of bytes needed for the extension header (4) and the possible padding that is needed at the end of the extension to round up to a four byte boundary (up to 3 bytes). For many applications this overhead is unacceptable.
- Use of the header extension limits applications to a single redundant encoding, unless further structure is introduced into the extension. This would result in further overhead.

For these reasons, the use of RTP header extension to hold redundant audio encodings is disregarded.

The RTP profile for audio and video conferences [3] lists a set of payload types and provides for a dynamic range of 32 encodings that may be defined through a conference control protocol. This leads to two possible schemes for assigning additional RTP payload types for redundant audio applications:

1. A dynamic encoding scheme may be defined, for each combination of primary/redundant payload types, using the RTP dynamic payload type range.
2. A single fixed payload type may be defined to represent a packet with redundancy. This may then be assigned to either a static RTP payload type, or the payload type for this may be assigned dynamically.

It is possible to define a set of payload types that signify a particular combination of primary and secondary encodings for each of the 32 dynamic payload types provided. This would be a slightly restrictive yet feasible solution for packets with a single block of redundancy as the number of possible combinations is not too large. However the need for multiple blocks of redundancy greatly increases the number of encoding combinations and makes this solution not viable.

A modified version of the above solution could be to decide prior to the beginning of a conference on a set a 32 encoding combinations that will be used for the duration of the conference. All tools in the conference can be initialized with this working set of encoding combinations. Communication of the working set could be made through the use of an external, out of band, mechanism. Setup is complicated as great care needs to be taken in starting tools with identical parameters. This scheme is more efficient as only one byte is used to identify combinations of encodings.

It is felt that the complication inherent in distributing the mapping of payload types onto combinations of redundant data preclude the use of this mechanism.

A more flexible solution is to have a single payload type which signifies a packet with redundancy. That packet then becomes a container, encapsulating multiple payloads into a single RTP packet. Such a scheme is flexible, since any amount of redundancy may be encapsulated within a single packet. There is, however, a small overhead since each encapsulated payload must be preceded by a header indicating the type of data enclosed. This is the preferred solution, since it is both flexible, extensible, and has a relatively low overhead. The remainder of this document describes this solution.

# 3 Payload Format Specification

The assignment of an RTP payload type for this new packet format is outside the scope of this document, and will not be specified here. It is expected that the RTP profile for a particular class of applications will assign a payload type for this encoding, or if that is not done then a payload type in the dynamic range shall be chosen.

An RTP packet containing redundant data shall have a standard RTP header, with payload type indicating redundancy. The other fields of the RTP header relate to the primary data block of the redundant data.

Following the RTP header are a number of additional headers, defined in the figure below, which specify the contents of each of the encodings carried by the packet. Following these additional headers are a number of data blocks, which contain the standard RTP payload data for these encodings. It is noted that all the headers are aligned to a 32 bit boundary, but that the payload data will typically not be aligned. If multiple redundant encodings are carried in a packet, they should correspond to different time intervals: there is no reason to include multiple copies of data for a single time interval within a packet.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|F|   block PT  |  timestamp offset         |   block length    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The bits in the header are specified as follows:

**F: 1 bit** First bit in header indicates whether another header block follows. If 1 further header blocks follow, if 0 this is the last header block.

**block PT: 7 bits** RTP payload type for this block.

**timestamp offset: 14 bits** Unsigned offset of timestamp of this block relative to timestamp given in RTP header. The use of an unsigned offset implies that redundant data must be sent after the primary data, and is hence a time to be subtracted from the current timestamp to determine the timestamp of the data for which this block is the redundancy.

**block length: 10 bits** Length in bytes of the corresponding data block excluding header.

It is noted that the use of an unsigned timestamp offset limits the use of redundant data slightly: it is not possible to send redundancy before the primary encoding. This may affect schemes where a low bandwidth coding suitable for redundancy is produced early in the encoding process, and hence could feasibly be transmitted early. However, the addition of a sign bit would unacceptably reduce the range of the timestamp offset, and increasing the size of the field above 14 bits limits the block length field. It seems that limiting redundancy to be transmitted after the primary will cause fewer problems than limiting the size of the other fields.

The timestamp offset for a redundant block is measured in the same units as the timestamp of the primary encoding (ie: audio samples, with the same clock rate as the primary). The implication of this is that the redundant encoding MUST be sampled at the same rate as the primary.

It is further noted that the block length and timestamp offset are 10 bits, and 14 bits respectively; rather than the more obvious 8 and 16 bits. Whilst such an encoding complicates parsing the header information slightly, and adds some additional processing overhead, there are a number of problems involved with the more obvious choice: An 8 bit block length field is sufficient for most, but not all, possible encodings: for example 80ms PCM and DVI audio packets comprise more than 256 bytes, and cannot be encoded with a single byte length field. It is possible to impose additional structure on the block length field (for example the high bit set could imply the lower 7 bits code a length in words, rather than bytes), however such schemes are complex. The use of a 10 bit block length field retains simplicity and provides an enlarged range, at the expense of a reduced range of timestamp values.

The primary encoding block header is placed last in the packet. It is therefore possible to omit the timestamp and block-length fields from the header of this block, since they may be determined from the RTP header and overall packet length. The header for the primary (final) block comprises only a zero F bit, and the block payload type information, a total of 8 bits. This is illustrated in the figure below:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|0|   Block PT  |
+-+-+-+-+-+-+-+-+
```

The final header is followed, immediately, by the data blocks, stored in the same order as the headers. There is no padding or other delimiter between the data blocks, and they are typically not 32 bit aligned. Again, this choice was made to reduce bandwidth overheads, at the expense of additional decoding time.

The choice of encodings used should reflect the bandwidth requirements of those encodings. It is expected that the redundant encoding shall use significantly less bandwidth that the primary encoding: the exception being the case where the primary is very low-bandwidth and has high processing requirement, in which case a copy of the primary MAY be used as the redundancy. The redundant encoding MUST NOT be higher bandwidth than the primary.

The use of multiple levels of redundancy is rarely necessary. However, in those cases which require it, the bandwidth required by each level of redundancy is expected to be significantly less than that of the previous level.

# 4 Limitations

The RTP marker bit is not preserved for redundant data blocks. Hence if the primary (containing this marker) is lost, the marker is lost. It is believed that this will not cause undue problems: even if the marker bit was transmitted with the redundant information, there would still be the possibility of its loss, so applications would still have to be written with this in mind.

In addition, CSRC information is not preserved for redundant data. The CSRC data in the RTP header of a redundant audio packet relates to the primary only. Since CSRC data in an audio stream is expected to change relatively infrequently, it is recommended that applications which require this information assume that the CSRC data in the RTP header may be applied to the reconstructed redundant data.

# 5 Relation to SDP

When a redundant payload is used, it may need to be bound to an RTP dynamic payload type. This may be achieved through any out-of-band mechanism, but one common way is to communicate this binding using the Session Description Protocol (SDP) [6]. SDP has a mechanism for binding a dynamic payload types to particular codec, sample rate, and number of channels using the "rtpmap" attribute. An example of its use (using the RTP audio/video profile [3]) is:

```
m=audio 12345 RTP/AVP 121 0 5
a=rtpmap:121 red/8000/1
```

This specifies that an audio stream using RTP is using payload types 121 (a dynamic payload type), 0 (PCM u-law) and 5 (DVI). The "rtpmap" attribute is used to bind payload type 121 to codec "red" indicating this codec is actually a redundancy frame, 8KHz, and monaural. When used with SDP, the term "red" is used to indicate the redundancy format discussed in this document.

In this case the additional formats of PCM and DVI are specified. The receiver must therefore be prepared to use these formats. Such a specification means the sender will send redundancy by default, but also may send PCM or DVI. However, with a redundant payload we additionally take this to mean that no codec other than PCM or DVI will be used in the redundant encodings. Note that the additional payload formats defined in the "m=" field may themselves be dynamic payload types, and if so a number of additional "a=" attributes may be required to describe these dynamic payload types.

To receive a redundant stream, this is all that is required. However to send a redundant stream, the sender needs to know which codecs are recommended for the primary and secondary (and tertiary, etc) encodings. This information is specific to the redundancy format, and is specified using an additional attribute "fmtp" which conveys format-specific information. A session directory does not parse the values specified in an fmtp attribute but merely hands it to the media tool unchanged. For redundancy, we define the format parameters to be a slash "/" separated list of RTP payload types.

Thus a complete example is:

```
m=audio 12345 RTP/AVP 121 0 5
a=rtpmap:121 red/8000/1
a=fmtp:121 0/5
```

This specifies that the default format for senders is redundancy with PCM as the primary encoding and DVI as the secondary encoding. Encodings cannot be specified in the fmtp attribute unless they are also specified as valid encodings on the media ("m=") line.

# 6 Security Considerations

RTP packets containing redundant information are subject to the security considerations discussed in the RTP specification [2], and any appropriate RTP profile (for example [3]). This implies that confidentiality of the media streams is achieved by encryption. Encryption of a redundant data stream may occur in two ways:

1. The entire stream is to be secured, and all participants are expected to have keys to decode the entire stream. In this case, nothing special need be done, and encryption is performed in the usual manner.
2. A portion of the stream is to be encrypted with a different key to the remainder. In this case a redundant copy of the last packet of that portion cannot be sent, since there is no following packet which is encrypted with the correct key in which to send it. Similar limitations may occur when enabling/disabling encryption.

The choice between these two is a matter for the encoder only. Decoders can decrypt either form without modification.

Whilst the addition of low-bandwidth redundancy to an audio stream is an effective means by which that stream may be protected against packet loss, application designers should be aware that the addition of large amounts of redundancy will increase network congestion, and hence packet loss, leading to a worsening of the problem which the use of redundancy was intended to solve. At its worst, this can lead to excessive network congestion and may constitute a denial of service attack.

# 7 Example Packet

An RTP audio data packet containing a DVI4 (8KHz) primary, and a single block of redundancy encoded using 8KHz LPC (both 20ms packets), as defined in the RTP audio/video profile [3] is illustrated:

```
 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=0  |M|      PT     |  sequence number of primary   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              timestamp  of  primary encoding                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1| block PT=7  |  timestamp offset         |   block length    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0| block PT=5  |                                               |
+-+-+-+-+-+-+-+-+                                               +
|                                                               |
+               LPC encoded redundant data (PT=7)               +
|               (14 bytes)                                      |
+                                               +---------------+
|                                               |               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               +
|                                                               |
+                                                               +
|                                                               |
+                                                               +
|                                                               |
+                                                               +
|               DVI4 encoded primary data (PT=5)                |
+               (84 bytes, not to scale)                        +
/                                                               /
+                                                               +
|                                                               |
+                                                               +
|                                                               |
+                                               +---------------+
|                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

# 8 Author's Addresses

Colin Perkins/Isidor Kouvelas/Orion Hodson/Vicky Hardman
Department of Computer Science
University College London
London WC1E 6BT
United Kingdom
Email: {c.perkins|i.kouvelas|o.hodson|v.hardman}@cs.ucl.ac.uk

Mark Handley
USC Information Sciences Institute
c/o MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139, USA
Email: mjh@isi.edu

Jean-Chrysostome Bolot/Andres Vega-Garcia/Sacha Fosse-Parisis
INRIA Sophia Antipolis
2004 Route des Lucioles, BP 93
06902 Sophia Antipolis
France
Email: {bolot|avega|sfosse}@sophia.inria.fr

# 9 References

[1] V.J. Hardman, M.A. Sasse, M. Handley and A. Watson; Reliable Audio for Use over the Internet; Proceedings INET'95, Honalulu, Oahu, Hawaii, September 1995. http://www.isoc.org/in95prc/

[2] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson; RTP: A Transport Protocol for Real-Time Applications; RFC 1889, January 1996

[3] H. Schulzrinne; RTP Profile for Audio and Video Conferences with Minimal Control; RFC 1890, January 1996

[4] M. Yajnik, J. Kurose and D. Towsley; Packet loss correlation in the MBone multicast network; IEEE Globecom Internet workshop, London, November 1996

[5] J.-C. Bolot and A. Vega-Garcia; The case for FEC-based error control for packet audio in the Internet; ACM Multimedia Systems, 1997

[6] M. Handley and V. Jacobson; SDP: Session Description Protocol (draft 03.2) draft-ietf-mmusic-sdp-03.txt, November 1996