AVT Working Group
Internet-Draft
Expires: April 24, 2006

L. Barbato
Xiph.Org

October 21, 2005

draft-kerr-avt-vorbis-rtp-05
RTP Payload Format for Vorbis Encoded Audio

Status of this Memo

Copyright Notice

Abstract

This document describes an RTP payload format for transporting Vorbis encoded audio. It details the RTP encapsulation mechanism for raw Vorbis data and details the delivery mechanisms for the decoder probability model, referred to as a codebook and other setup information.

Also included within the document are the necessary details for the use of Vorbis with MIME and Session Description Protocol (SDP).

Editors Note

All references to RFC XXXX are to be replaced by references to the RFC number of this memo, when published.

Table of Contents

1. Introduction

Vorbis is a general purpose perceptual audio codec intended to allow maximum encoder flexibility, thus allowing it to scale competitively over an exceptionally wide range of bitrates. At the high quality/ bitrate end of the scale (CD or DAT rate stereo, 16/24 bits), it is in the same league as MPEG-2 and MPC. Similarly, the version 1.1 reference encoder can encode high-quality CD and DAT rate stereo at below 48k bits/sec without resampling to a lower rate. Vorbis is also intended for lower and higher sample rates (from 8kHz telephony to 192kHz digital masters) and a range of channel representations (monaural, polyphonic, stereo, quadraphonic, 5.1, ambisonic, or up to 255 discrete channels).

Vorbis encoded audio is generally encapsulated within an Ogg format bitstream [1], which provides framing and synchronization. For the purposes of RTP transport, this layer is unnecessary, and so raw Vorbis packets are used in the payload.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [2].

2. Payload Format

For RTP based transportation of Vorbis encoded audio the standard RTP header is followed by a 4 octet payload header, then the payload data. The payload headers are used to associate the Vorbis data with its associated decoding codebooks as well as indicating if the following packet contains fragmented Vorbis data and/or the the number of whole Vorbis data frames. The payload data contains the raw Vorbis bitstream information.

2.1. RTP Header

The format of the RTP header is specified in [3] and shown in Figure Figure 1. This payload format uses the fields of the header in a manner consistent with that specification.

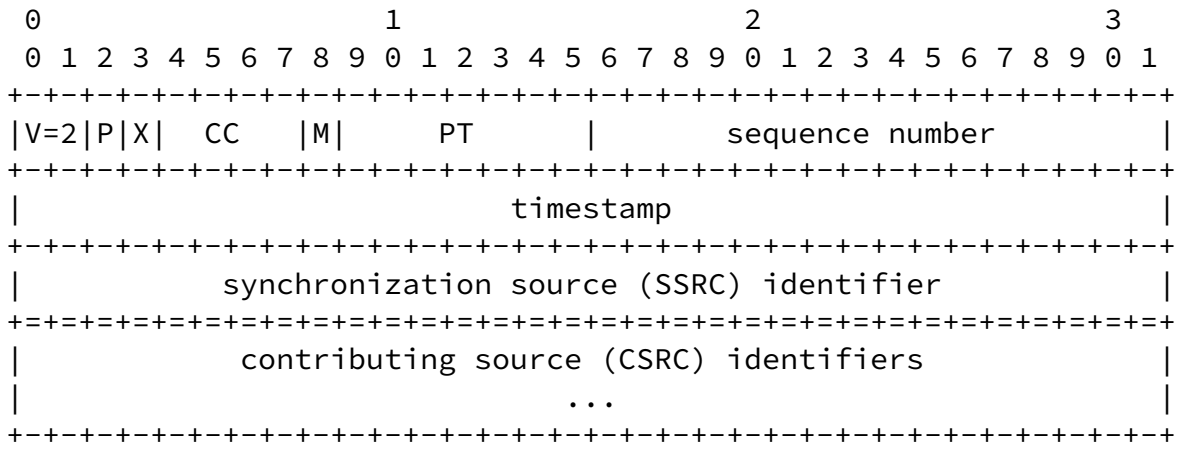


Figure 1: RTP Header

The RTP header begins with an octet of fields (V, P, X, and CC) to support specialized RTP uses (see [3] and [4] for details). For Vorbis RTP, the following values are used.

Version (V): 2 bits

This field identifies the version of RTP. The version used by this specification is two (2).

Padding (P): 1 bit

Padding MAY be used with this payload format according to section 5.1 of [3].

Extension (X): 1 bit

The Extension bit is used in accordance with [3].

CSRC count (CC): 4 bits

The CSRC count is used in accordance with [3].

Marker (M): 1 bit

Set to zero. Audio silence suppression not used. This conforms to section 4.1 of [12].

Payload Type (PT): 7 bits

An RTP profile for a class of applications is expected to assign a payload type for this format, or a dynamically allocated payload type SHOULD be chosen which designates the payload as Vorbis.

Sequence number: 16 bits

The sequence number increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence. This field is detailed further in [3].

Timestamp: 32 bits

A timestamp representing the sampling time of the first sample of the first Vorbis packet in the RTP packet. The clock frequency MUST be set to the sample rate of the encoded audio data and is conveyed out-of-band as a SDP attribute.

SSRC/CSRC identifiers:

These two fields, 32 bits each with one SSRC field and a maximum of 16 CSRC fields, are as defined in [3].

2.2. Payload Header

After the RTP Header section the following 4 octets are the Payload Header. This header is split into a number of bitfields detailing

the format of the following payload data packets.

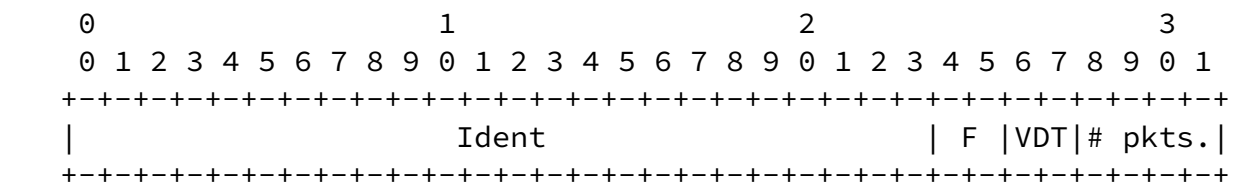


Figure 2: Payload Header

Ident: 24 bits

This 24 bit field is used to associate the Vorbis data to a decoding Configuration.

Fragment type (F): 2 bits

This field is set accordingly the following list

0 = Not Fragmented
1 = Start Fragment
2 = Continuation Fragment
3 = End Fragment

Vorbis Data Type (VDT): 2 bits

This field sets the packet payload type for the Vorbis data. There are currently three type of Vorbis payloads.

0 = Raw Vorbis payload
1 = Vorbis Packed Configuration payload
2 = Legacy Vorbis Comment payload
3 = Reserved

The last 4 bits are the number of complete packets in this payload. This provides for a maximum number of 15 Vorbis packets in the payload. If the packet contains fragmented data the number of packets MUST be set to 0.

2.3. Payload Data

Raw Vorbis packets are unbounded in length currently, although at some future point there will likely be a practical limit placed on them. Typical Vorbis packet sizes are from very small (2-3 bytes) to quite large (8-12 kilobytes). The reference implementation [11] typically produces packets less than ~800 bytes, except for the setup header packets which are ~4-12 kilobytes. Within an RTP context the maximum packet size, including the RTP and payload headers, SHOULD be kept below the path MTU to avoid packet fragmentation.

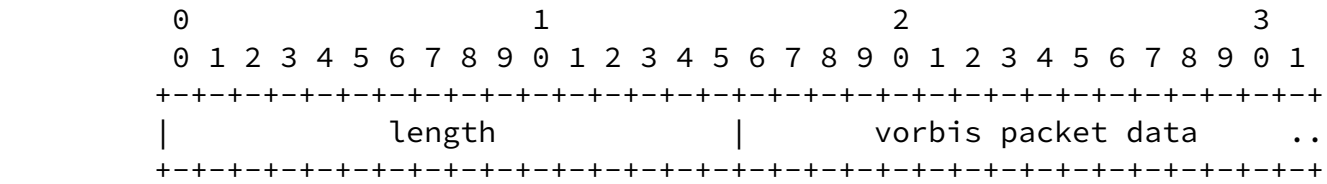


Figure 3: Payload Data Header

Each Vorbis payload packet starts with a two octet length header, which is used to represent the size of the following data payload, followed by the raw Vorbis data padded to the nearest byte boundary.

For payloads which consist of multiple Vorbis packets the payload data consists of the packet length followed by the packet data for each of the Vorbis packets in the payload.

The Vorbis packet length header is the length of the Vorbis data block only and does not count the length field.

The payload packing of the Vorbis data packets MUST follow the guidelines set-out in [4] where the oldest packet occurs immediately after the RTP packet header.

Channel mapping of the audio is in accordance with the Vorbis I

Specification [12].

2.4. Example RTP Packet

Here is an example RTP packet containing two Vorbis packets.

RTP Packet Header:

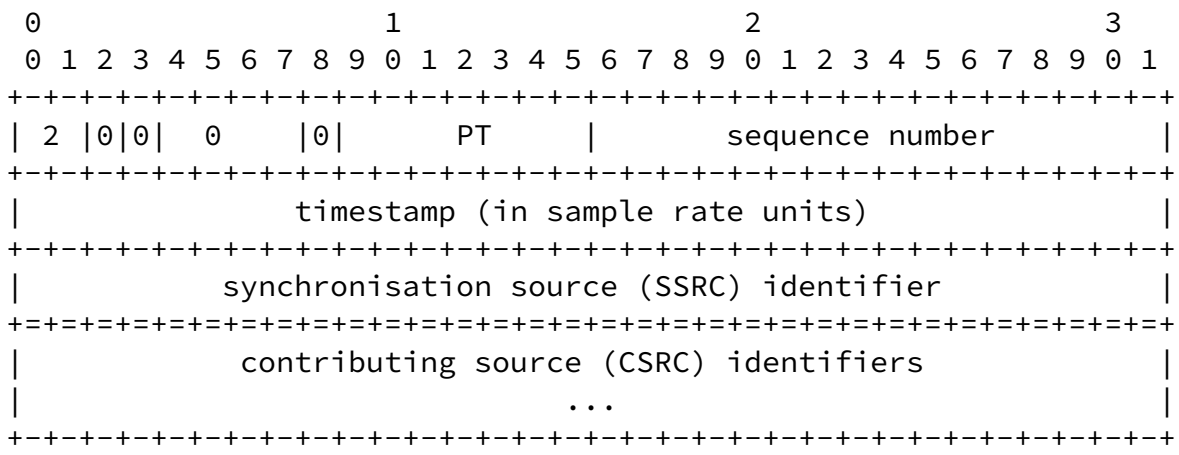


Figure 4: Example Packet (RTP Headers)

Payload Data:

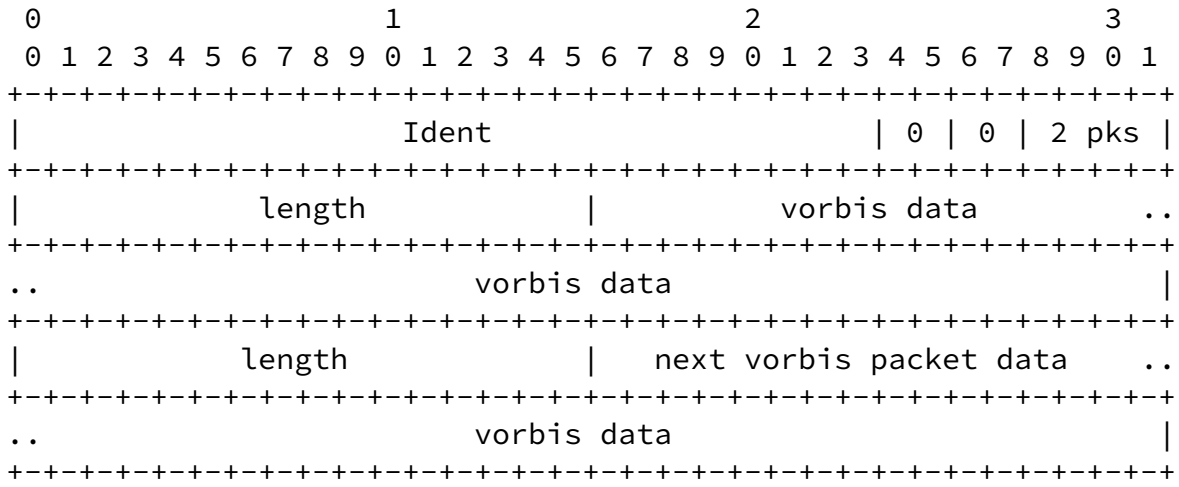


Figure 5: Example Packet (Payload Data)

The payload data section of the RTP packet starts with the 24 bit Ident field followed by the one octet bitfield header, which has the number of Vorbis frames set to 2. Each of the Vorbis data frames is prefixed by the two octet length field. The Packet Type and Fragment Type are set to 0. The decode Configuration that will be used to decode the packets is the one indexed by the ident value.

3. Configuration Headers

Unlike other mainstream audio codecs Vorbis has no statically configured probability model. Instead, it packs all entropy decoding configuration, VQ and Huffman models into a data block that must be transmitted to the decoder along with the compressed data. A decoder also requires identification information detailing the number of audio channels, bitrates and other information to configure itself for a particular compressed data stream. These two blocks of information are often referred to collectively as the "codebooks" for a Vorbis stream, and are nominally included as special "header" packets at the start of the compressed data.

Thus these two codebook header packets must be received by the decoder before any audio data can be interpreted. In addition, the Vorbis I specification [12] requires the presense of a comment header packet which gives simple metadata about the stream. This requirement poses problems in RTP, which is often used over unreliable transports.

Since this information must be transmitted reliably and, as the RTP stream may change certain configuration data mid-session, there are different methods for delivering this configuration data to a client, both in-band and out-of-band which is detailed below. SDP delivery is used to setup an initial state for the client application. The changes may be due to different codebooks as well as different bitrates of the stream.

The delivery vectors in use are specified by an SDP attribute to indicate the method and the optional URI where the Vorbis Packed Configuration (Section 3.1.1) Packets could be fetched. Different delivery methods MAY be advertised for the same session. The in-band Configuration delivery SHOULD be considered as baseline, out-of-band delivery methods that don't use RTP will not be described in this document. For non chained streams, the Configuration delivery method RECOMMENDED is inline the Packed Configuration (Section 3.1.1) in the SDP as explained in the IANA considerations (Section 6.1) section.

The 24 bit Ident field is used to map which Configuration will be used to decodea packet. When the Ident field changes, it indicates that a change in the stream has taken place. The client application MUST have in advance the correct configuration and if the client detects a change in the Ident value and does not have this information it MUST NOT decode the raw Vorbis data associated until it fetches the correct Configuration.

3.1. In-band Header Transmission

The Packed Configuration (Section 3.1.1) Payload is sent in-band with the packet type bits set to match the payload type. Clients MUST be capable of dealing with fragmentation and periodic re-transmission of the configuration headers.

3.1.1. Packed Configuration

A Vorbis Packed Configuration is indicated with the payload type field set to 1. Of the three headers, defined in the Vorbis I specification [12], the identification and the setup will be packed together, the comment header is completely suppressed. Is up to the client provide a minimal size comment header to the decoder if required by the implementation.

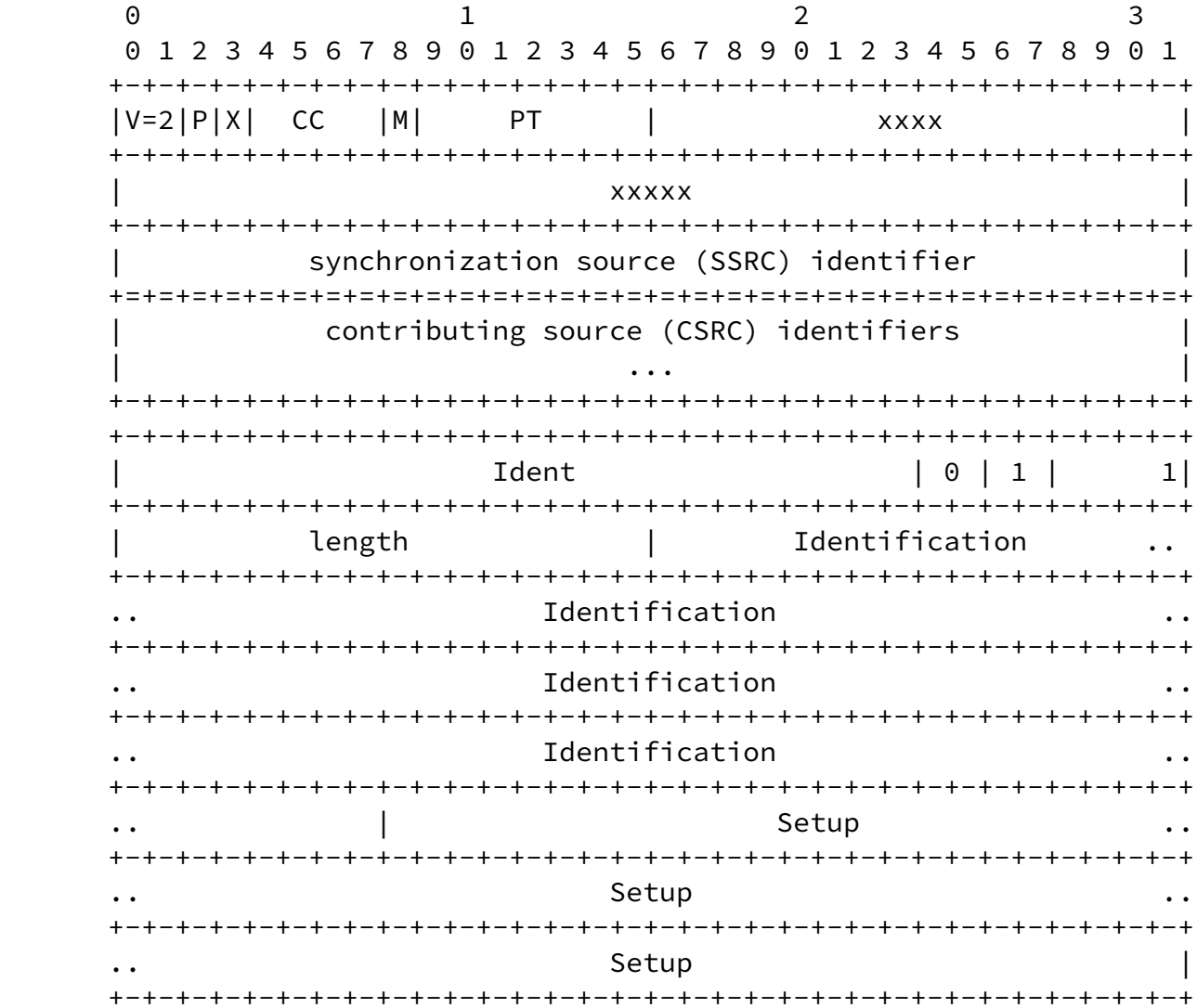


Figure 6: Packed Configuration Figure

The Ident field is set with the value that will be used by the Raw Payload Packets to address this Configuration. The Fragment type is set to 0 since the packet bears the full Packed configuration, the number of packet is set to 1.

3.2. Out of Band Transmission

This section, as stated before, won't cover all the possible out-of-band delivery methods since they rely to different protocols and be linked to a specific application. The following packet definition SHOULD be used in out-of-band delivery and MUST be used when Configuration is inlined in the SDP.

3.2.1. Packed Headers

As mentioned above the RECOMMENDED delivery vector for Vorbis configuration data is via a retrieval method that can be performed using a reliable transport protocol. As the RTP headers are not required for this method of delivery the structure of the configuration data is slightly different. The packed header starts with a 32 bit count field which details the number of packed headers that are contained in the bundle. Next is the Packed header payload for each chained Vorbis stream.

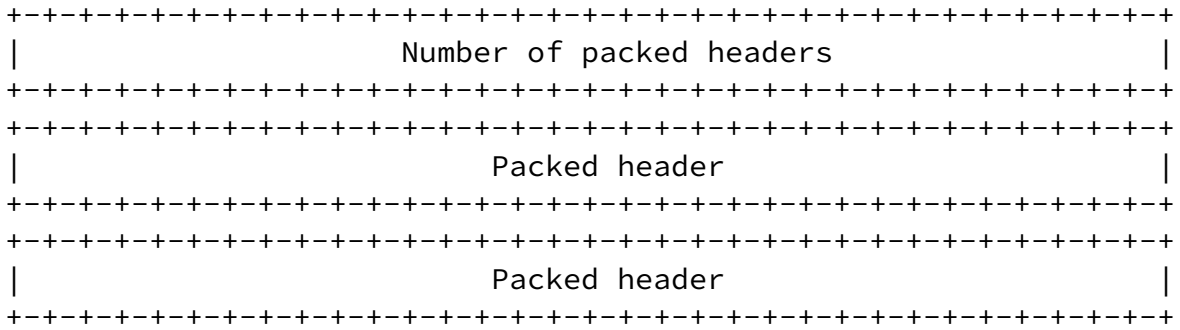


Figure 7: Packed Headers Overview

Since the Configuration Ident and the Identification Header are fixed length there is only a 2 byte length tag to define the length of the packed headers.

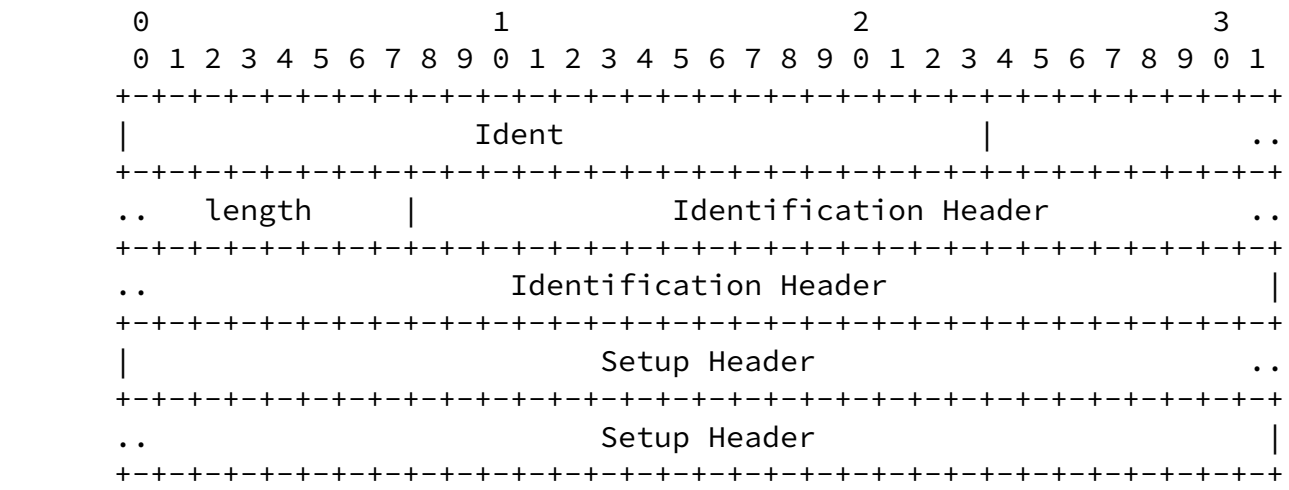


Figure 8: Packed Headers Detail

The key difference between the in-band format and this one, is there is no need for the payload header octet.

3.2.1.1. Packed Headers IANA Considerations

The following IANA considerations MUST only be applied to the packed headers.

MIME media type name: audio

MIME subtype: vorbis-config

Required Parameters:

None.

Optional Parameters:

None.

Encoding considerations:

This type is only defined for transfer via non RTP protocols as specified in RFC XXXX.

Security Considerations:

See Section 6 of RFC 3047.

Interoperability considerations: none

Published specification:

See RFC XXXX for details.

Applications which use this media type:

Vorbis encoded audio, configuration data.

Additional information: none

Person & email address to contact for further information:

Luca Barbato: <lu_zero@gentoo.org>

Intended usage: COMMON

Author/Change controller:

Author: Luca Barbato

Change controller: IETF AVT Working Group

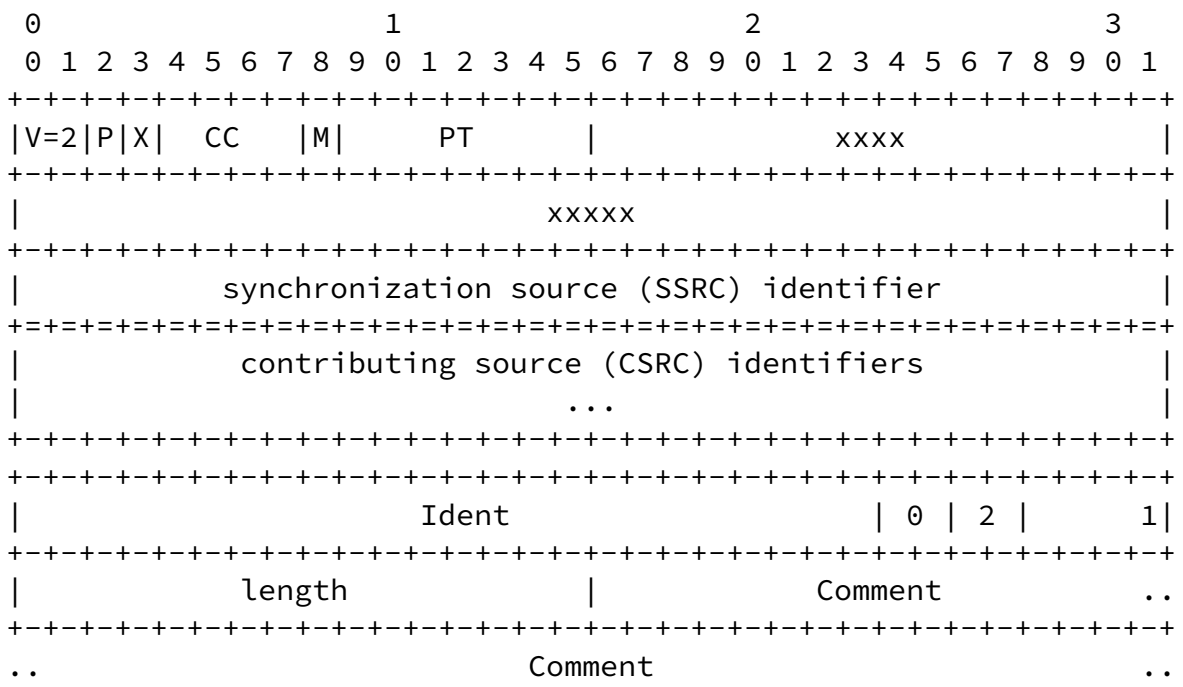
3.3. Loss of Configuration Headers

Unlike the loss of raw Vorbis payload data, loss of a configuration header can lead to a situation where it will not be possible to successfully decode the stream.

Loss of Configuration Packet results in the halting of stream decoding and SHOULD be reported to the client as well as a loss report sent via RTCP.

4. Comment Headers

With the payload type flag set to 2, this indicates that the packet contain the comment metadata, such as artist name, track title and so on. These metadata messages are not intended to be fully descriptive but to offer basic track/song information. Clients MAY ignore it completely. The details on the format of the comments can be found in the Vorbis documentation [12].



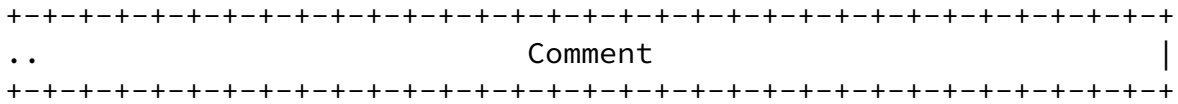


Figure 9: Comment Packet

The 2 bytes length field is necessary since this packet could be fragmented.

5. Frame Packetizing

Each RTP packet contains either one Vorbis packet fragment, or an integer number of complete Vorbis packets (up to a max of 15 packets, since the number of packets is defined by a 4 bit value).

Any Vorbis data packet that is less than path MTU SHOULD be bundled in the RTP packet with as many Vorbis packets as will fit, up to a maximum of 15. Path MTU is detailed in [6] and [7].

If a Vorbis packet, not only data but also Configuration and Comment, is larger than 65535 octets it MUST be fragmented. A fragmented packet has a zero in the last four bits of the payload header. The first fragment will set the Fragment type to 1. Each fragment after the first will set the Fragment type to 2 in the payload header. The RTP packet containing the last fragment of the Vorbis packet will have the Fragment type set to 3. To maintain the correct sequence for fragmented packet reception the timestamp field of fragmented packets MUST be the same as the first packet sent, with the sequence number incremented as normal for the subsequent RTP packets. The

length field shows the fragment length.

5.1. Example Fragmented Vorbis Packet

Here is an example fragmented Vorbis packet split over three RTP packets. Each packet contains the standard RTP headers as well as the 4 octet Vorbis headers.

Packet 1:

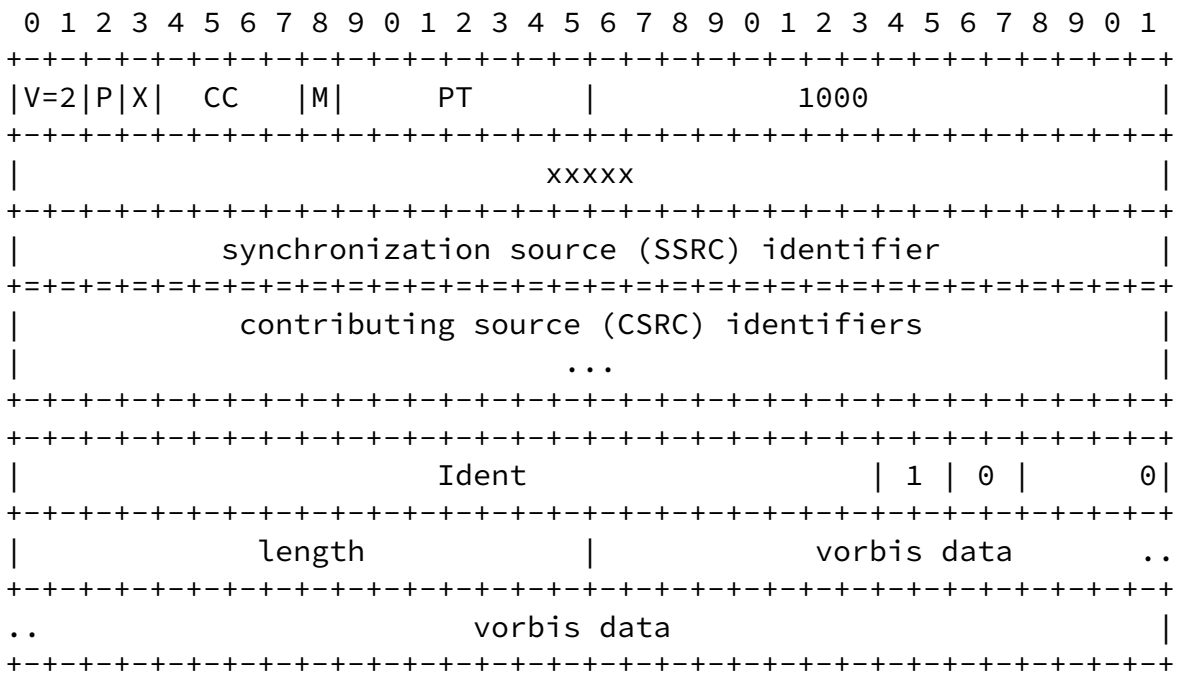
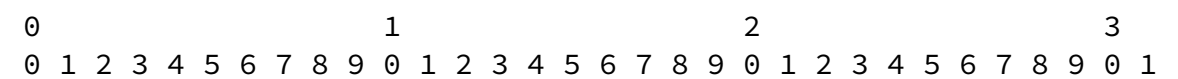


Figure 10: Example Fragmented Packet (Packet 1)

In this packet the initial sequence number is 1000 and the timestamp is xxxxx. The Fragment type is set to 1, the number of packets field is set to 0, and as the payload is raw Vorbis data the VDT field is set to 0.

Packet 2:



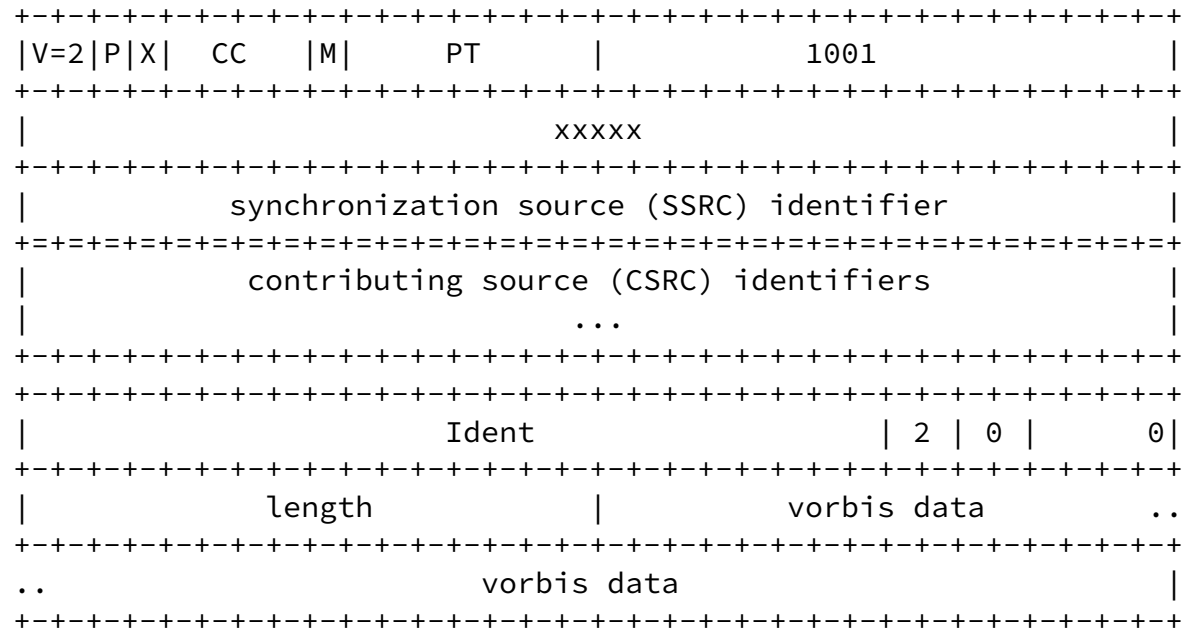


Figure 11: Example Fragmented Packet (Packet 2)

The Fragment type field is set to 2 and the number of packets field is set to 0. For large Vorbis fragments there can be several of these type of payload packets. The maximum packet size SHOULD be no greater than the path MTU, including all RTP and payload headers. The sequence number has been incremented by one but the timestamp field remains the same as the initial packet.

Packet 3:

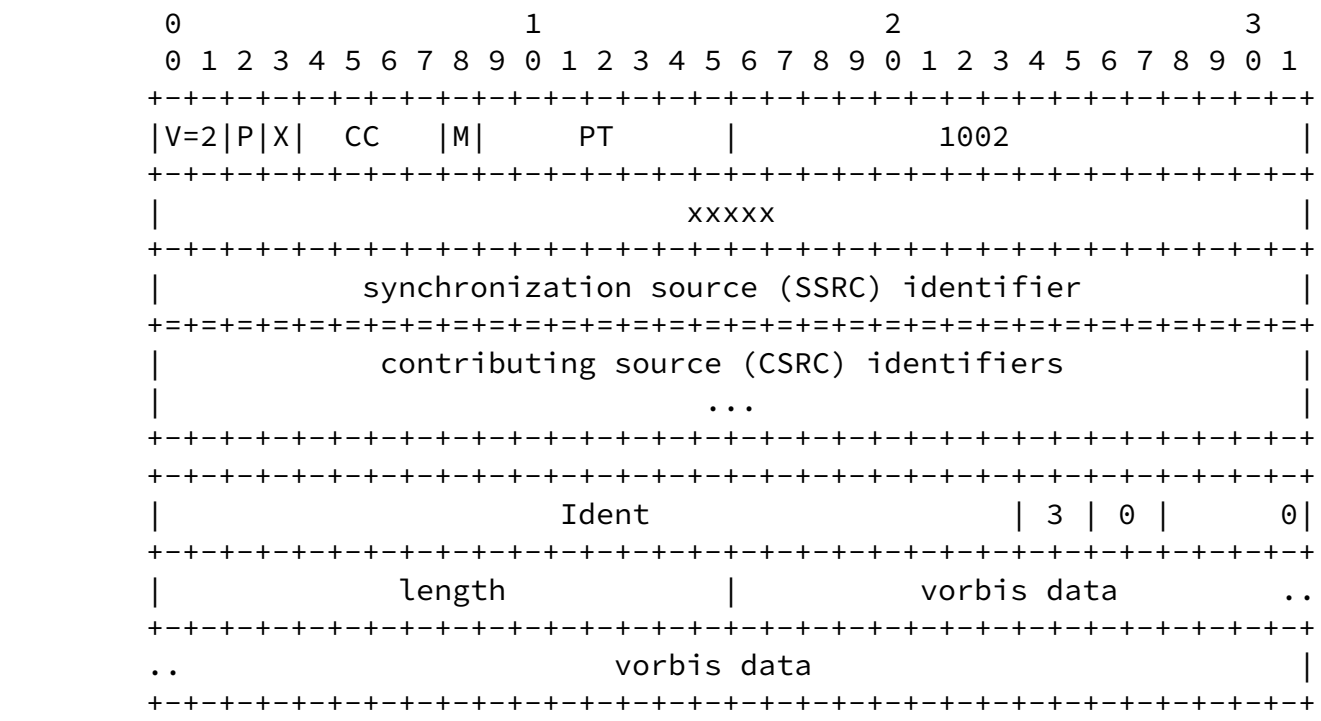


Figure 12: Example Fragmented Packet (Packet 3)

This is the last Vorbis fragment packet. The Fragment type is set to 3 and the packet count remains set to 0. As in the previous packets the timestamp remains set to the first packet in the sequence and the sequence number has been incremented.

5.2. Packet Loss

As there is no error correction within the Vorbis stream, packet loss will result in a loss of signal. Packet loss is more of an issue for fragmented Vorbis packets as the client will have to cope with the handling of the Fragment Type. In case of loss of fragments the client MUST discard all the remaining fragments and decode the incomplete packet. If we use the fragmented Vorbis packet example above and the first packet is lost the client MUST detect that the next packet has the packet count field set to 0 and the Fragment type 2 and MUST drop it. The next packet, which is the final fragmented packet, MUST be dropped in the same manner. If the missing packet is the last, the received two fragments will be kept and the incomplete vorbis packet decoded. Feedback reports on lost and dropped packets MUST be sent back via RTCP.

If a particular multicast session has a large number of participants care must be taken to prevent an RTCP feedback implosion, [10], in

the event of packet loss from a large number of participants.

Loss of any of the Configuration fragment will result in the loss of the full Configuration packet with the result detailed in the Loss of Configuration Headers (Section 3.3) section.

6. IANA Considerations

MIME media type name: audio

MIME subtype: vorbis

Required Parameters:

delivery-method: indicates the delivery methods in use, the possible values are:inline, in_band, out_band

configuration: the base16 [9] (hexadecimal) representation of the Packed Headers (Section 3.2.1).

Optional Parameters:

configuration-uri: the URI of the configuration headers in case of out of band transmission. In the form of "protocol://path/to/resource/". Depending on the specific method the single ident packet could be retrived by their number, or aggregated in a single stream.

Encoding considerations:

This type is only defined for transfer via RTP as specified in RFC XXXX.

Security Considerations:

See Section 6 of RFC 3047.

Interoperability considerations: none

Published specification:

See the Vorbis documentation [12] for details.

Applications which use this media type:

Audio streaming and conferencing tools

Additional information: none

Person & email address to contact for further information:

Luca Barbato: <lu_zero@gentoo.org>

Intended usage: COMMON

Author/Change controller:

Author: Luca Barbato

Change controller: IETF AVT Working Group

6.1. Mapping MIME Parameters into SDP

The information carried in the MIME media type specification has a specific mapping to fields in the Session Description Protocol (SDP) [5], which is commonly used to describe RTP sessions. When SDP is used to specify sessions the mapping are as follows:

- The MIME type ("audio") goes in SDP "m=" as the media name.
- The MIME subtype ("vorbis") goes in SDP "a=rtpmap" as the encoding name.
- The parameter "rate" also goes in "a=rtpmap" as clock rate.
- The parameter "channels" also goes in "a=rtpmap" as channel count.
- The mandated parameters "delivery-method" and "configuration" MUST be included in the SDP "a=fmpt" attribute.
- The optional parameter "configuration-uri", when present, MUST be included in the SDP "a=fmpt" attribute.

If the stream comprises chained Vorbis files and all of them are known in advance, the Configuration Packet for each file SHOULD be passed to the client using the configuration attribute.

The URI specified in the configuration-uri attribute MUST point to a location where all of the Configuration Packets needed for the life of the session reside.

The port value is specified by the server application bound to the address specified in the c attribute. The bitrate value and channels specified in the rtpmap attribute MUST match the Vorbis sample rate value. An example is found below.

The answer to any offer, [8], MUST NOT change the URI specified in

the configuration-uri attribute. The Configuration inlined in the configuration parameter MAY change.

```
c=IN IP4/6
m=audio RTP/AVP 98
a=rtpmap:98 VORBIS/44100/2
a=delivery:out_band/http
a=fmtp:98 delivery-method:in_band,out_band/http;
configuration=base16string1;
configuration-uri=http://path/to/the/resource
```

Note that the payload format (encoding) names are commonly shown in upper case. MIME subtypes are commonly shown in lower case. These names are case-insensitive in both places. Similarly, parameter names are case-insensitive both in MIME types and in the default mapping to the SDP a=fmtp attribute. The exception regarding case sensitivity is the configuration-uri URI which MUST be regarded as being case sensitive.

7. Congestion Control

Vorbis clients SHOULD send regular receiver reports detailing congestion. A mechanism for dynamically downgrading the stream, known as bitrate peeling, will allow for a graceful backing off of the stream bitrate. This feature is not available at present so an

alternative would be to redirect the client to a lower bitrate stream if one is available.

If a particular multicast session has a large number of participants care must be taken to prevent an RTCP feedback implosion, [10], in the event of congestion.

8. Examples

The following examples are common usage patterns that MAY be applied in such situations, the main scope of this section is to explain better usage of the transmission vectors.

8.1. Stream Radio

That is one of the most common situation: one single server streaming content in multicast, the clients may start a session at random time. The content itself could be a mix of live stream as the dj's speech and stored streams as the music she plays.

In this situation we don't know in advance how many codebooks we will

use and. The clients can join anytime and users expect to start listening to the content in a short time

On join the client will receive the current Configuration necessary to decode the current stream inlined in the SDP. And can start decoding the current stream.

When the streamed content changes the new Configuration is sent in-band befoe the actual stream, and the Configuration that has to be sent inline in the SDP updated.

A serverside optimization would be keep an hash list of the Configurations per session to avoid packing them and send the same Configuration with different Ident tags

A clientside optimization would be keep a tag list of the Configurations per session and don't process configuration packets already known.

Let's assume that the client playout buffer can store at least 7 packets and that is the maximum latency.

9. Security Considerations

RTP packets using this payload format are subject to the security considerations discussed in the RTP specification [3]. This implies that the confidentiality of the media stream is achieved by using encryption. Because the data compression used with this payload format is applied end-to-end, encryption may be performed on the compressed data. Where the size of a data block is set care MUST be taken to prevent buffer overflows in the client applications.

10. Acknowledgments

This document is a continuation of draft-moffitt-vorbis-rtp-00.txt and draft-kerr-avt-vorbis-rtp-04.txt. The MIME type section is a continuation of draft-short-avt-rtp-vorbis-mime-00.txt.

Thanks to the AVT, Ogg Vorbis Communities / Xiph.org including Steve Casner, Aaron Colwell, Ross Finlayson, Fluendo, Ramon Garcia, Pascal Hennequin, Ralph Giles, Tor-Einar Jarnbjo, Colin Law, John Lazzaro, Jack Moffitt, Christopher Montgomery, Colin Perkins, Barry Short, Mike Smith, Phil Kerr, Michael Sparks, Magnus Westerlund, David Barrett, Silvia Pfeiffer, Politecnico di Torino (LS)^3/IMG Group in particular Federico Ridolfo, Francesco Varano, Giampaolo Mancini, Juan Carlos De Martin.

11. References

11.1. Normative References

[1] Pfeiffer, S., "The Ogg Encapsulation Format Version 0", RFC 3533.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119.

[3] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for real-time applications",

RFC 3550.

[4] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control.", RFC 3551.

[5] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327.

[6] Mogul et al., J., "Path MTU Discovery", RFC 1063.

[7] McCann et al., J., "Path MTU Discovery for IP version 6", RFC 1981.

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

[9] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 3548.

[10] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)", Internet Draft (draft-ietf-avt-rtcp-feedback-11: Work in progress).

11.2. Informative References

[11] "libvorbis: Available from the Xiph website, http://www.xiph.org".

[12] "Ogg Vorbis I specification: Codec setup and packet decode. Available from the Xiph website, http://www.xiph.org".

[13] "Ogg Vorbis I specification: Comment field and header specification. Available from the Xiph website, http://www.xiph.org".

Author's Address

Luca Barbato
Xiph.Org

Email: lu_zero@gentoo.org
URI: http://www.xiph.org/

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.