

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: July 17, 2015

J. Spittka  
  
K. Vos  
vocTone  
JM. Valin  
Mozilla  
January 13, 2015

**RTP Payload Format for Opus Speech and Audio Codec**  
**draft-ietf-payload-rtp-opus-07**

**Abstract**

This document defines the Real-time Transport Protocol (RTP) payload format for packetization of Opus encoded speech and audio data necessary to integrate the codec in the most compatible way. Further, it describes media type registrations for the RTP payload format.

**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 July 17, 2015.

**Copyright Notice**

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	<a href="#">Introduction . . . . .</a>	<a href="#">2</a>
<a href="#">2.</a>	<a href="#">Conventions, Definitions and Acronyms used in this document .</a>	<a href="#">3</a>
<a href="#">3.</a>	<a href="#">Opus Codec . . . . .</a>	<a href="#">3</a>
<a href="#">3.1.</a>	<a href="#">Network Bandwidth . . . . .</a>	<a href="#">4</a>
<a href="#">3.1.1.</a>	<a href="#">Recommended Bitrate . . . . .</a>	<a href="#">4</a>
<a href="#">3.1.2.</a>	<a href="#">Variable versus Constant Bitrate . . . . .</a>	<a href="#">4</a>
<a href="#">3.1.3.</a>	<a href="#">Discontinuous Transmission (DTX) . . . . .</a>	<a href="#">4</a>
<a href="#">3.2.</a>	<a href="#">Complexity . . . . .</a>	<a href="#">5</a>
<a href="#">3.3.</a>	<a href="#">Forward Error Correction (FEC) . . . . .</a>	<a href="#">5</a>
<a href="#">3.4.</a>	<a href="#">Stereo Operation . . . . .</a>	<a href="#">6</a>
<a href="#">4.</a>	<a href="#">Opus RTP Payload Format . . . . .</a>	<a href="#">6</a>
<a href="#">4.1.</a>	<a href="#">RTP Header Usage . . . . .</a>	<a href="#">6</a>
<a href="#">4.2.</a>	<a href="#">Payload Structure . . . . .</a>	<a href="#">7</a>
<a href="#">5.</a>	<a href="#">Congestion Control . . . . .</a>	<a href="#">8</a>
<a href="#">6.</a>	<a href="#">IANA Considerations . . . . .</a>	<a href="#">8</a>
<a href="#">6.1.</a>	<a href="#">Opus Media Type Registration . . . . .</a>	<a href="#">8</a>
<a href="#">6.2.</a>	<a href="#">Mapping to SDP Parameters . . . . .</a>	<a href="#">12</a>
<a href="#">6.2.1.</a>	<a href="#">Offer-Answer Model Considerations for Opus . . . . .</a>	<a href="#">13</a>
<a href="#">6.2.2.</a>	<a href="#">Declarative SDP Considerations for Opus . . . . .</a>	<a href="#">15</a>
<a href="#">7.</a>	<a href="#">Security Considerations . . . . .</a>	<a href="#">15</a>
<a href="#">8.</a>	<a href="#">Acknowledgements . . . . .</a>	<a href="#">15</a>
<a href="#">9.</a>	<a href="#">References . . . . .</a>	<a href="#">16</a>
<a href="#">9.1.</a>	<a href="#">Normative References . . . . .</a>	<a href="#">16</a>
<a href="#">9.2.</a>	<a href="#">Informative References . . . . .</a>	<a href="#">17</a>
	<a href="#">Authors' Addresses . . . . .</a>	<a href="#">17</a>

## [1.](#) Introduction

The Opus codec is a speech and audio codec developed within the IETF Internet Wideband Audio Codec working group. The codec has a very low algorithmic delay and it is highly scalable in terms of audio bandwidth, bitrate, and complexity. Further, it provides different modes to efficiently encode speech signals as well as music signals, thus making it the codec of choice for various applications using the Internet or similar networks.

This document defines the Real-time Transport Protocol (RTP) [[RFC3550](#)] payload format for packetization of Opus encoded speech and audio data necessary to integrate the Opus codec in the most compatible way. Further, it describes media type registrations for the RTP payload format. More information on the Opus codec can be obtained from [[RFC6716](#)].



## 2. Conventions, Definitions and Acronyms used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [\[RFC2119\]](#).

audio bandwidth: The range of audio frequencies being coded

CBR: Constant bitrate

CPU: Central Processing Unit

DTX: Discontinuous transmission

FEC: Forward error correction

IP: Internet Protocol

samples: Speech or audio samples (per channel)

SDP: Session Description Protocol

VBR: Variable bitrate

Throughout this document, we refer to the following definitions:

Abbreviation	Name	Audio Bandwidth (Hz)	Sampling Rate (Hz)
NB	Narrowband	0 - 4000	8000
MB	Mediumband	0 - 6000	12000
WB	Wideband	0 - 8000	16000
SWB	Super-wideband	0 - 12000	24000
FB	Fullband	0 - 20000	48000

Audio bandwidth naming

Table 1

## 3. Opus Codec

The Opus [\[RFC6716\]](#) codec encodes speech signals as well as general audio signals. Two different modes can be chosen, a voice mode or an audio mode, to allow the most efficient coding depending on the type of the input signal, the sampling frequency of the input signal, and the intended application.

The voice mode allows efficient encoding of voice signals at lower bit rates while the audio mode is optimized for general audio signals at medium and higher bitrates.



The Opus speech and audio codec is highly scalable in terms of audio bandwidth, bitrate, and complexity. Further, Opus allows transmitting stereo signals.

### **3.1. Network Bandwidth**

Opus supports bitrates from 6 kb/s to 510 kb/s. The bitrate can be changed dynamically within that range. All other parameters being equal, higher bitrates result in higher quality.

#### **3.1.1. Recommended Bitrate**

For a frame size of 20 ms, these are the bitrate "sweet spots" for Opus in various configurations:

- o 8-12 kb/s for NB speech,
- o 16-20 kb/s for WB speech,
- o 28-40 kb/s for FB speech,
- o 48-64 kb/s for FB mono music, and
- o 64-128 kb/s for FB stereo music.

#### **3.1.2. Variable versus Constant Bitrate**

For the same average bitrate, variable bitrate (VBR) can achieve higher quality than constant bitrate (CBR). For the majority of voice transmission applications, VBR is the best choice. One reason for choosing CBR is the potential information leak that *\_might\_* occur when encrypting the compressed stream. See [[RFC6562](#)] for guidelines on when VBR is appropriate for encrypted audio communications. In the case where an existing VBR stream needs to be converted to CBR for security reasons, then the Opus padding mechanism described in [[RFC6716](#)] is the RECOMMENDED way to achieve padding because the RTP padding bit is unencrypted.

The bitrate can be adjusted at any point in time. To avoid congestion, the average bitrate SHOULD NOT exceed the available network bandwidth. If no target bitrate is specified, the bitrates specified in [Section 3.1.1](#) are RECOMMENDED.

#### **3.1.3. Discontinuous Transmission (DTX)**

The Opus codec can, as described in [Section 3.1.2](#), be operated with a variable bitrate. In that case, the encoder will automatically reduce the bitrate for certain input signals, like periods of silence. When using continuous transmission, it will reduce the bitrate when the characteristics of the input signal permit, but will never interrupt the transmission to the receiver. Therefore, the received signal will maintain the same high level of quality over the



full duration of a transmission while minimizing the average bit rate over time.

In cases where the bitrate of Opus needs to be reduced even further or in cases where only constant bitrate is available, the Opus encoder can use discontinuous transmission (DTX), where parts of the encoded signal that correspond to periods of silence in the input speech or audio signal are not transmitted to the receiver. A receiver can distinguish between DTX and packet loss by looking for gaps in the sequence number, as described by [Section 4.1 of \[RFC3551\]](#).

On the receiving side, the non-transmitted parts will be handled by a frame loss concealment unit in the Opus decoder which generates a comfort noise signal to replace the non transmitted parts of the speech or audio signal. Use of [\[RFC3389\]](#) Comfort Noise (CN) with Opus is discouraged. The transmitter MUST drop whole frames only, based on the size of the last transmitted frame, to ensure successive RTP timestamps differ by a multiple of 120 and to allow the receiver to use whole frames for concealment.

DTX can be used with both variable and constant bitrate. It will have a slightly lower speech or audio quality than continuous transmission. Therefore, using continuous transmission is RECOMMENDED unless restraints on available network bandwidth are severe.

### [3.2.](#) Complexity

Complexity of the encoder can be scaled to optimize for CPU resources in real-time, mostly as a trade-off between audio quality and bitrate. Also, different modes of Opus have different complexity.

### [3.3.](#) Forward Error Correction (FEC)

The voice mode of Opus allows for embedding "in-band" forward error correction (FEC) data into the Opus bit stream. This FEC scheme adds redundant information about the previous packet (N-1) to the current output packet N. For each frame, the encoder decides whether to use FEC based on (1) an externally-provided estimate of the channel's packet loss rate; (2) an externally-provided estimate of the channel's capacity; (3) the sensitivity of the audio or speech signal to packet loss; (4) whether the receiving decoder has indicated it can take advantage of "in-band" FEC information. The decision to send "in-band" FEC information is entirely controlled by the encoder and therefore no special precautions for the payload have to be taken.





On the receiving side, the decoder can take advantage of this additional information when it loses a packet and the next packet is available. In order to use the FEC data, the jitter buffer needs to provide access to payloads with the FEC data. Instead of performing loss concealment for a missing packet, the receiver can then configure its decoder to decode the FEC data from the next packet.

Any compliant Opus decoder is capable of ignoring FEC information when it is not needed, so encoding with FEC cannot cause interoperability problems. However, if FEC cannot be used on the receiving side, then FEC SHOULD NOT be used, as it leads to an inefficient usage of network resources. Decoder support for FEC SHOULD be indicated at the time a session is set up.

### **3.4. Stereo Operation**

Opus allows for transmission of stereo audio signals. This operation is signaled in-band in the Opus payload and no special arrangement is needed in the payload format. An Opus decoder is capable of handling a stereo encoding, but an application might only be capable of consuming a single audio channel.

If a decoder cannot take advantage of the benefits of a stereo signal this SHOULD be indicated at the time a session is set up. In that case the sending side SHOULD NOT send stereo signals as it leads to an inefficient usage of network resources.

## **4. Opus RTP Payload Format**

The payload format for Opus consists of the RTP header and Opus payload data.

### **4.1. RTP Header Usage**

The format of the RTP header is specified in [[RFC3550](#)]. The use of the fields of the RTP header by the Opus payload format is consistent with that specification.

The payload length of Opus is an integer number of octets and therefore no padding is necessary. The payload MAY be padded by an integer number of octets according to [[RFC3550](#)], although the Opus internal padding is preferred.

The timestamp, sequence number, and marker bit (M) of the RTP header are used in accordance with [Section 4.1 of \[\[RFC3551\]\(#\)\]](#).

The RTP payload type for Opus is to be assigned dynamically.



The receiving side **MUST** be prepared to receive duplicate RTP packets. The receiver **MUST** provide at most one of those payloads to the Opus decoder for decoding, and **MUST** discard the others.

Opus supports 5 different audio bandwidths, which can be adjusted during a call. The RTP timestamp is incremented with a 48000 Hz clock rate for all modes of Opus and all sampling rates. The unit for the timestamp is samples per single (mono) channel. The RTP timestamp corresponds to the sample time of the first encoded sample in the encoded frame. For data encoded with sampling rates other than 48000 Hz, the sampling rate has to be adjusted to 48000 Hz.

#### **4.2. Payload Structure**

The Opus encoder can output encoded frames representing 2.5, 5, 10, 20, 40, or 60 ms of speech or audio data. Further, an arbitrary number of frames can be combined into a packet, up to a maximum packet duration representing 120 ms of speech or audio data. The grouping of one or more Opus frames into a single Opus packet is defined in [Section 3 of \[RFC6716\]](#). An RTP payload **MUST** contain exactly one Opus packet as defined by that document.

Figure 1 shows the structure combined with the RTP header.

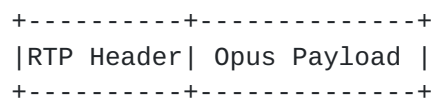


Figure 1: Packet structure with RTP header

Table 2 shows supported frame sizes in milliseconds of encoded speech or audio data for the speech and audio modes (Mode) and sampling rates (fs) of Opus and shows how the timestamp is incremented for packetization (ts incr). If the Opus encoder outputs multiple encoded frames into a single packet, the timestamp increment is the sum of the increments for the individual frames.



Mode	fs	2.5	5	10	20	40	60
ts incr	all	120	240	480	960	1920	2880
voice	NB/MB/WB/SWB/FB	x	x	o	o	o	o
audio	NB/WB/SWB/FB	o	o	o	o	x	x

Table 2: Supported Opus frame sizes and timestamp increments marked with an o. Unsupported marked with an x.

## 5. Congestion Control

The target bitrate of Opus can be adjusted at any point in time, thus allowing efficient congestion control. Furthermore, the amount of encoded speech or audio data encoded in a single packet can be used for congestion control, since the transmission rate is inversely proportional to the packet duration. A lower packet transmission rate reduces the amount of header overhead, but at the same time increases latency and loss sensitivity, so it ought to be used with care.

It is RECOMMENDED that senders of Opus encoded data apply congestion control.

## 6. IANA Considerations

One media subtype (audio/opus) has been defined and registered as described in the following section.

### 6.1. Opus Media Type Registration

Media type registration is done according to [\[RFC6838\]](#) and [\[RFC4855\]](#).

Type name: audio

Subtype name: opus

Required parameters:

rate: the RTP timestamp is incremented with a 48000 Hz clock rate for all modes of Opus and all sampling rates. For data encoded



with sampling rates other than 48000 Hz, the sampling rate has to be adjusted to 48000 Hz.

Optional parameters:

**maxplaybackrate:** a hint about the maximum output sampling rate that the receiver is capable of rendering in Hz. The decoder **MUST** be capable of decoding any audio bandwidth but due to hardware limitations only signals up to the specified sampling rate can be played back. Sending signals with higher audio bandwidth results in higher than necessary network usage and encoding complexity, so an encoder **SHOULD NOT** encode frequencies above the audio bandwidth specified by maxplaybackrate. This parameter can take any value between 8000 and 48000, although commonly the value will match one of the Opus bandwidths (Table 1). By default, the receiver is assumed to have no limitations, i.e. 48000.

**sprop-maxcapture:** a hint about the maximum input sampling rate that the sender is likely to produce. This is not a guarantee that the sender will never send any higher bandwidth (e.g. it could send a pre-recorded prompt that uses a higher bandwidth), but it indicates to the receiver that frequencies above this maximum can safely be discarded. This parameter is useful to avoid wasting receiver resources by operating the audio processing pipeline (e.g. echo cancellation) at a higher rate than necessary. This parameter can take any value between 8000 and 48000, although commonly the value will match one of the Opus bandwidths (Table 1). By default, the sender is assumed to have no limitations, i.e. 48000.

**maxptime:** the maximum duration of media represented by a packet (according to [Section 6 of \[RFC4566\]](#)) that a decoder wants to receive, in milliseconds rounded up to the next full integer value. Possible values are 3, 5, 10, 20, 40, 60, or an arbitrary multiple of an Opus frame size rounded up to the next full integer value, up to a maximum value of 120, as defined in [Section 4](#). If no value is specified, the default is 120.

**ptime:** the preferred duration of media represented by a packet (according to [Section 6 of \[RFC4566\]](#)) that a decoder wants to receive, in milliseconds rounded up to the next full integer value. Possible values are 3, 5, 10, 20, 40, 60, or an arbitrary multiple of an Opus frame size rounded up to the next full integer value, up to a maximum value of 120, as defined in [Section 4](#). If no value is specified, the default is 20.





**maxaveragebitrate:** specifies the maximum average receive bitrate of a session in bits per second (b/s). The actual value of the bitrate can vary, as it is dependent on the characteristics of the media in a packet. Note that the maximum average bitrate MAY be modified dynamically during a session. Any positive integer is allowed, but values outside the range 6000 to 510000 SHOULD be ignored. If no value is specified, the maximum value specified in [Section 3.1.1](#) for the corresponding mode of Opus and corresponding **maxplaybackrate** is the default.

**stereo:** specifies whether the decoder prefers receiving stereo or mono signals. Possible values are 1 and 0 where 1 specifies that stereo signals are preferred, and 0 specifies that only mono signals are preferred. Independent of the stereo parameter every receiver MUST be able to receive and decode stereo signals but sending stereo signals to a receiver that signaled a preference for mono signals may result in higher than necessary network utilization and encoding complexity. If no value is specified, the default is 0 (mono).

**sprop-stereo:** specifies whether the sender is likely to produce stereo audio. Possible values are 1 and 0, where 1 specifies that stereo signals are likely to be sent, and 0 specifies that the sender will likely only send mono. This is not a guarantee that the sender will never send stereo audio (e.g. it could send a pre-recorded prompt that uses stereo), but it indicates to the receiver that the received signal can be safely downmixed to mono. This parameter is useful to avoid wasting receiver resources by operating the audio processing pipeline (e.g. echo cancellation) in stereo when not necessary. If no value is specified, the default is 0 (mono).

**cbr:** specifies if the decoder prefers the use of a constant bitrate versus variable bitrate. Possible values are 1 and 0, where 1 specifies constant bitrate and 0 specifies variable bitrate. If no value is specified, the default is 0 (vbr). When cbr is 1, the maximum average bitrate can still change, e.g. to adapt to changing network conditions.

**useinbandfec:** specifies that the decoder has the capability to take advantage of the Opus in-band FEC. Possible values are 1 and 0. Providing 0 when FEC cannot be used on the receiving side is RECOMMENDED. If no value is specified, **useinbandfec** is assumed to be 0. This parameter is only a preference and the receiver MUST



be able to process packets that include FEC information, even if it means the FEC part is discarded.

usedtx: specifies if the decoder prefers the use of DTX. Possible values are 1 and 0. If no value is specified, the default is 0.

Encoding considerations:

The Opus media type is framed and consists of binary data according to [Section 4.8 in \[RFC6838\]](#).

Security considerations:

See [Section 7](#) of this document.

Interoperability considerations: none

Published specification: RFC [XXXX]

Note to the RFC Editor: Replace [XXXX] with the number of the published RFC.

Applications that use this media type:

Any application that requires the transport of speech or audio data can use this media type. Some examples are, but not limited to, audio and video conferencing, Voice over IP, media streaming.

Fragment identifier considerations: N/A

Person & email address to contact for further information:

SILK Support [silksupport@skype.net](mailto:silksupport@skype.net)  
Jean-Marc Valin [jmvalin@jmvalin.ca](mailto:jmvalin@jmvalin.ca)

Intended usage: COMMON

Restrictions on usage:

For transfer over RTP, the RTP payload format ([Section 4](#) of this document) SHALL be used.



Author:

Julian Spittka [jspittka@gmail.com](mailto:jspittka@gmail.com)

Koen Vos [koenvos74@gmail.com](mailto:koenvos74@gmail.com)

Jean-Marc Valin [jmvalin@jmvalin.ca](mailto:jmvalin@jmvalin.ca)

Change controller: IETF Payload Working Group delegated from the IESG

## **6.2. Mapping to SDP Parameters**

The information described in the media type specification has a specific mapping to fields in the Session Description Protocol (SDP) [[RFC4566](#)], which is commonly used to describe RTP sessions. When SDP is used to specify sessions employing the Opus codec, the mapping is as follows:

- o The media type ("audio") goes in SDP "m=" as the media name.
- o The media subtype ("opus") goes in SDP "a=rtpmap" as the encoding name. The RTP clock rate in "a=rtpmap" MUST be 48000 and the number of channels MUST be 2.
- o The OPTIONAL media type parameters "ptime" and "maxptime" are mapped to "a=ptime" and "a=maxptime" attributes, respectively, in the SDP.
- o The OPTIONAL media type parameters "maxaveragebitrate", "maxplaybackrate", "stereo", "cbr", "useinbandfec", and "usedtx", when present, MUST be included in the "a=fmtp" attribute in the SDP, expressed as a media type string in the form of a semicolon-separated list of parameter=value pairs (e.g., maxplaybackrate=48000). They MUST NOT be specified in an SSRC-specific "fmtp" source-level attribute (as defined in [Section 6.3 of \[RFC5576\]](#)).
- o The OPTIONAL media type parameters "sprop-maxcapture", and "sprop-stereo" MAY be mapped to the "a=fmtp" SDP attribute by copying them directly from the media type parameter string as part of the semicolon-separated list of parameter=value pairs (e.g., sprop-stereo=1). These same OPTIONAL media type parameters MAY also be specified using an SSRC-specific "fmtp" source-level attribute as described in [Section 6.3 of \[RFC5576\]](#). They MAY be specified in both places, in which case the parameter in the source-level attribute overrides the one found on the "a=fmtp" line. The value of any parameter which is not specified in a source-level source attribute MUST be taken from the "a=fmtp" line, if it is present there.

Below are some examples of SDP session descriptions for Opus:



Example 1: Standard mono session with 48000 Hz clock rate

```
m=audio 54312 RTP/AVP 101
a=rtpmap:101 opus/48000/2
```

Example 2: 16000 Hz clock rate, maximum packet size of 40 ms, recommended packet size of 40 ms, maximum average bitrate of 20000 bps, prefers to receive stereo but only plans to send mono, FEC is desired, DTX is not desired

```
m=audio 54312 RTP/AVP 101
a=rtpmap:101 opus/48000/2
a=fmtp:101 maxplaybackrate=16000; sprop-maxcapture=16000;
maxaveragebitrate=20000; stereo=1; useinbandfec=1; usedtx=0
a=ptime:40
a=maxptime:40
```

Example 3: Two-way full-band stereo preferred

```
m=audio 54312 RTP/AVP 101
a=rtpmap:101 opus/48000/2
a=fmtp:101 stereo=1; sprop-stereo=1
```

#### **6.2.1. Offer-Answer Model Considerations for Opus**

When using the offer-answer procedure described in [[RFC3264](#)] to negotiate the use of Opus, the following considerations apply:

- o Opus supports several clock rates. For signaling purposes only the highest, i.e. 48000, is used. The actual clock rate of the corresponding media is signaled inside the payload and is not restricted by this payload format description. The decoder **MUST** be capable of decoding every received clock rate. An example is shown below:

```
m=audio 54312 RTP/AVP 100
a=rtpmap:100 opus/48000/2
```

- o The "ptime" and "maxptime" parameters are unidirectional receive-only parameters and typically will not compromise interoperability; however, some values might cause application





performance to suffer. [RFC3264] defines the SDP offer-answer handling of the "ptime" parameter. The "maxptime" parameter MUST be handled in the same way.

- o The "maxplaybackrate" parameter is a unidirectional receive-only parameter that reflects limitations of the local receiver. When sending to a single destination, a sender MUST NOT use an audio bandwidth higher than necessary to make full use of audio sampled at a sampling rate of "maxplaybackrate". Gateways or senders that are sending the same encoded audio to multiple destinations SHOULD NOT use an audio bandwidth higher than necessary to represent audio sampled at "maxplaybackrate", as this would lead to inefficient use of network resources. The "maxplaybackrate" parameter does not affect interoperability. Also, this parameter SHOULD NOT be used to adjust the audio bandwidth as a function of the bitrate, as this is the responsibility of the Opus encoder implementation.
- o The "maxaveragebitrate" parameter is a unidirectional receive-only parameter that reflects limitations of the local receiver. The sender of the other side MUST NOT send with an average bitrate higher than "maxaveragebitrate" as it might overload the network and/or receiver. The "maxaveragebitrate" parameter typically will not compromise interoperability; however, some values might cause application performance to suffer, and ought to be set with care.
- o The "sprop-maxcapture" and "sprop-stereo" parameters are unidirectional sender-only parameters that reflect limitations of the sender side. They allow the receiver to set up a reduced-complexity audio processing pipeline if the sender is not planning to use the full range of Opus's capabilities. Neither "sprop-maxcapture" nor "sprop-stereo" affect interoperability and the receiver MUST be capable of receiving any signal.
- o The "stereo" parameter is a unidirectional receive-only parameter. When sending to a single destination, a sender MUST NOT use stereo when "stereo" is 0. Gateways or senders that are sending the same encoded audio to multiple destinations SHOULD NOT use stereo when "stereo" is 0, as this would lead to inefficient use of network resources. The "stereo" parameter does not affect interoperability.
- o The "cbr" parameter is a unidirectional receive-only parameter.
- o The "useinbandfec" parameter is a unidirectional receive-only parameter.
- o The "usedtx" parameter is a unidirectional receive-only parameter.
- o Any unknown parameter in an offer MUST be ignored by the receiver and MUST be removed from the answer.



### **6.2.2. Declarative SDP Considerations for Opus**

For declarative use of SDP such as in Session Announcement Protocol (SAP), [[RFC2974](#)], and RTSP, [[RFC2326](#)], for Opus, the following needs to be considered:

- o The values for "maxptime", "ptime", "maxplaybackrate", and "maxaveragebitrate" ought to be selected carefully to ensure that a reasonable performance can be achieved for the participants of a session.
- o The values for "maxptime", "ptime", and of the payload format configuration are recommendations by the decoding side to ensure the best performance for the decoder.
- o All other parameters of the payload format configuration are declarative and a participant **MUST** use the configurations that are provided for the session. More than one configuration can be provided if necessary by declaring multiple RTP payload types; however, the number of types ought to be kept small.

## **7. Security Considerations**

All RTP packets using the payload format defined in this specification are subject to the general security considerations discussed in the RTP specification [[RFC3550](#)] and any profile from, e.g., [[RFC3711](#)] or [[RFC3551](#)].

This payload format transports Opus encoded speech or audio data. Hence, security issues include confidentiality, integrity protection, and authentication of the speech or audio itself. Opus does not provide any confidentiality or integrity protection. Any suitable external mechanisms, such as SRTP [[RFC3711](#)], **MAY** be used.

This payload format and the Opus encoding do not exhibit any significant non-uniformity in the receiver-end computational load and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological datagrams.

## **8. Acknowledgements**

Many people have made useful comments and suggestions contributing to this document. In particular, we would like to thank Tina le Grand, Cullen Jennings, Jonathan Lennox, Gregory Maxwell, Colin Perkins, Jan Skoglund, Timothy B. Terriberry, Martin Thompson, Justin Uberti, Magnus Westerlund, and Mo Zanaty.



## **9. References**

### **9.1. Normative References**

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2326] Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time Streaming Protocol (RTSP)", [RFC 2326](#), April 1998.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", [RFC 3264](#), June 2002.
- [RFC3389] Zopf, R., "Real-time Transport Protocol (RTP) Payload for Comfort Noise (CN)", [RFC 3389](#), September 2002.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, [RFC 3550](#), July 2003.
- [RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, [RFC 3551](#), July 2003.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", [RFC 3711](#), March 2004.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", [RFC 4566](#), July 2006.
- [RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", [RFC 4855](#), February 2007.
- [RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", [RFC 5576](#), June 2009.
- [RFC6562] Perkins, C. and JM. Valin, "Guidelines for the Use of Variable Bit Rate Audio with Secure RTP", [RFC 6562](#), March 2012.
- [RFC6716] Valin, JM., Vos, K., and T. Terriberry, "Definition of the Opus Audio Codec", [RFC 6716](#), September 2012.



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

## **[9.2.](#) Informative References**

[RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", [RFC 2974](#), October 2000.

### Authors' Addresses

Julian Spittka

Email: [jspittka@gmail.com](mailto:jspittka@gmail.com)

Koen Vos

vocTone

Email: [koenvos74@gmail.com](mailto:koenvos74@gmail.com)

Jean-Marc Valin

Mozilla

331 E. Evelyn Avenue

Mountain View, CA 94041

USA

Email: [jmvalin@jmvalin.ca](mailto:jmvalin@jmvalin.ca)



