

Network Working Group  
Internet-Draft  
Intended status: Informational  
Expires: June 3, 2013

J. Spittka  
  
K. Vos  
Skype Technologies S.A.  
JM. Valin  
Mozilla  
November 30, 2012

**RTP Payload Format for Opus Speech and Audio Codec**  
**draft-spittka-payload-rtp-opus-03**

**Abstract**

This document defines the Real-time Transport Protocol (RTP) payload format for packetization of Opus encoded speech and audio data that is essential to integrate the codec in the most compatible way. Further, media type registrations are described 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 June 3, 2013.

**Copyright Notice**

Copyright (c) 2012 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>	Introduction . . . . .	<a href="#">3</a>
<a href="#">2.</a>	Conventions, Definitions and Acronyms used in this document .	<a href="#">4</a>
<a href="#">2.1.</a>	Audio Bandwidth . . . . .	<a href="#">4</a>
<a href="#">3.</a>	Opus Codec . . . . .	<a href="#">5</a>
<a href="#">3.1.</a>	Network Bandwidth . . . . .	<a href="#">5</a>
<a href="#">3.1.1.</a>	Recommended Bitrate . . . . .	<a href="#">5</a>
<a href="#">3.1.2.</a>	Variable versus Constant Bit Rate . . . . .	<a href="#">5</a>
<a href="#">3.1.3.</a>	Discontinuous Transmission (DTX) . . . . .	<a href="#">6</a>
<a href="#">3.2.</a>	Complexity . . . . .	<a href="#">6</a>
<a href="#">3.3.</a>	Forward Error Correction (FEC) . . . . .	<a href="#">6</a>
<a href="#">3.4.</a>	Stereo Operation . . . . .	<a href="#">7</a>
<a href="#">4.</a>	Opus RTP Payload Format . . . . .	<a href="#">8</a>
<a href="#">4.1.</a>	RTP Header Usage . . . . .	<a href="#">8</a>
<a href="#">4.2.</a>	Payload Structure . . . . .	<a href="#">9</a>
<a href="#">5.</a>	Congestion Control . . . . .	<a href="#">11</a>
<a href="#">6.</a>	IANA Considerations . . . . .	<a href="#">12</a>
<a href="#">6.1.</a>	Opus Media Type Registration . . . . .	<a href="#">12</a>
<a href="#">6.2.</a>	Mapping to SDP Parameters . . . . .	<a href="#">15</a>
<a href="#">6.2.1.</a>	Offer-Answer Model Considerations for Opus . . . . .	<a href="#">17</a>
<a href="#">6.2.2.</a>	Declarative SDP Considerations for Opus . . . . .	<a href="#">18</a>
<a href="#">7.</a>	Security Considerations . . . . .	<a href="#">19</a>
<a href="#">8.</a>	Acknowledgements . . . . .	<a href="#">20</a>
<a href="#">9.</a>	Normative References . . . . .	<a href="#">21</a>
	Authors' Addresses . . . . .	<a href="#">22</a>



## **1. Introduction**

The Opus codec is a speech and audio codec developed within the IETF Internet Wideband Audio Codec working group (codec). 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 that is essential to integrate the Opus codec in the most compatible way. Further, media type registrations are described 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](#)].

CBR: Constant bitrate

CPU: Central Processing Unit

DTX: Discontinuous transmission

FEC: Forward error correction

IP: Internet Protocol

samples: Speech or audio samples (usually per channel)

SDP: Session Description Protocol

VBR: Variable bitrate

### 2.1. Audio Bandwidth

Throughout this document, we refer to the following definitions:

Abbreviation	Name	Bandwidth	Sampling
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](#)] speech and audio codec has been developed to encode speech signals as well as audio signals. Two different modes, a voice mode or an audio mode, may be chosen to allow the most efficient coding dependent on the type of input signal, the sampling frequency of the input signal, and the specific application.

The voice mode allows efficient encoding of voice signals at lower bit rates while the audio mode is optimized for 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 all bitrates from 6 kb/s to 510 kb/s. The bitrate can be changed dynamically within that range. All other parameters being equal, higher bitrate results 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 Bit Rate**

For the same average bitrate, variable bitrate (VBR) can achieve higher quality than constant bitrate (CBR). For the majority of voice transmission application, VBR is the best choice. One potential reason for choosing CBR is the potential information leak that may 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 be adjusted to the available network capacity. 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 may, as described in [Section 3.1.2](#), be operated with an adaptive bitrate. In that case, the bitrate will automatically be reduced for certain input signals like periods of silence. During continuous transmission the bitrate will be reduced, when the input signal allows to do so, but the transmission to the receiver itself will never be interrupted. 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 may be set to 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.

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.

The DTX mode of Opus will have a slightly lower speech or audio quality than the continuous mode. Therefore, it is RECOMMENDED to use Opus in the continuous mode unless restraints on network capacity are severe. The DTX mode can be engaged for operation in both adaptive or constant bitrate.

### **3.2. Complexity**

Complexity 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 "in-band" forward error correction (FEC) data to be embedded into the bit stream of Opus. 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, in case of a packet loss, 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. The decoder API function has a flag to indicate that a FEC frame rather than a regular frame should be decoded. If no FEC data is available for the current frame, the decoder will consider the frame lost and invokes the frame loss concealment.

If the FEC scheme is not implemented on the receiving side, 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 required in the payload format. Any implementation of the Opus decoder MUST be capable of receiving stereo signals, although it MAY decode those signals as mono.

If a decoder can not 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 the network.



## **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 Opus payload format uses the fields of the RTP header consistent with this specification.

The payload length of Opus is a multiple number of octets and therefore no padding is required. The payload MAY be padded by an integer number of octets according to [[RFC3550](#)].

The marker bit (M) of the RTP header is used in accordance with [Section 4.1 of \[RFC3551\]](#).

The RTP payload type for Opus has not been assigned statically and is expected to be assigned dynamically.

The receiving side MUST be prepared to receive duplicates of RTP packets. Only one of those payloads MUST be provided to the Opus decoder for decoding and others MUST be discarded.

Opus supports 5 different audio bandwidths which may be adjusted during the duration of a call. The RTP timestamp clock frequency is defined as the highest supported sampling frequency of Opus, i.e. 48000 Hz, for all modes and sampling rates of Opus. 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 sampling rates lower than 48000 Hz the number of samples has to be multiplied with a multiplier according to Table 2 to determine the RTP timestamp.



fs (Hz)	Multiplier
8000	6
12000	4
16000	3
24000	2
48000	1

Table 2: Timestamp multiplier

#### 4.2. Payload Structure

The Opus encoder can be set to 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. The maximum packet length is limited to the amount of encoded data representing 120 ms of speech or audio data. The packetization of encoded data is purely done by the Opus encoder and therefore only one packet output from the Opus encoder MUST be used as a payload.

Figure 1 shows the structure combined with the RTP header.

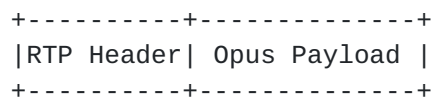


Figure 1: Payload Structure with RTP header

Table 3 shows supported frame sizes in milliseconds of encoded speech or audio data for speech and audio mode (Mode) and sampling rates (fs) of Opus and how the timestamp needs to be incremented for packetization (ts incr). If the Opus encoder outputs multiple encoded frames into a single packet the timestamps have to be added up according to the combined 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	x	x
audio	nb/wb/swb/fb	x	x	x	x		

Table 3: Supported Opus frame sizes and timestamp increments



## **5. Congestion Control**

The adaptive nature of the Opus codec allows for an efficient congestion control.

The target bitrate of Opus can be adjusted at any point in time and thus allowing for an 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 these frame sizes. A lower packet transmission rate reduces the amount of header overhead but at the same time increases latency and error sensitivity and should be done with care.

It is RECOMMENDED that congestion control is applied during the transmission of Opus encoded data.



## **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 [\[RFC4288\]](#) and [\[RFC4855\]](#).

Type name: audio

Subtype name: opus

Required parameters:

rate: RTP timestamp clock rate is incremented with 48000 Hz clock rate for all modes of Opus and all sampling frequencies. For audio sampling rates other than 48000 Hz the rate has to be adjusted to 48000 Hz according to Table 2.

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 decoder's maximum length of time in milliseconds rounded up to the next full integer value represented by the media in a packet that can be encapsulated in a received packet according to [Section 6 of \[RFC4566\]](#). Possible values are 3, 5, 10, 20, 40, and 60 or an arbitrary multiple of Opus frame sizes 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, 120 is assumed as default. This value is a recommendation by the decoding side to ensure the best performance for the decoder. The decoder **MUST** be capable of accepting any allowed packet sizes to ensure maximum compatibility.

**ptime:** the decoder's recommended length of time in milliseconds rounded up to the next full integer value represented by the media in a packet according to [Section 6 of \[RFC4566\]](#). Possible values are 3, 5, 10, 20, 40, or 60 or an arbitrary multiple of Opus frame sizes 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, 20 is assumed as default. If ptime is greater than maxptime, ptime **MUST** be ignored. This parameter **MAY** be changed during a session. This value is a recommendation by the decoding side to ensure the best performance for the decoder. The decoder **MUST** be capable of accepting any allowed packet sizes to ensure maximum compatibility.

**minptime:** the decoder's minimum length of time in milliseconds rounded up to the next full integer value represented by the media in a packet that **SHOULD** be encapsulated in a received packet according to [Section 6 of \[RFC4566\]](#). Possible values are 3, 5, 10, 20, 40, and 60 or an arbitrary multiple of Opus frame sizes 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, 3 is assumed as default. This value is a recommendation by the decoding side to ensure the best performance for the decoder. The decoder **MUST** be capable to accept any allowed packet sizes to ensure maximum compatibility.

**maxaveragebitrate:** specifies the maximum average receive bitrate of a session in bits per second (b/s). The actual value of the bitrate may 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 between 6000 and 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: will be 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 utilisation and encoding complexity. If no value is specified, mono is assumed (stereo=0).

**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, mono is assumed (sprop-stereo=0).

**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, cbr is assumed to be 0. Note that the maximum average bitrate may still be changed, 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. It is **RECOMMENDED** to provide 0 in case FEC cannot be utilized on the receiving side. 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, usedtx is assumed to be 0.

Encoding considerations:



Opus media type is framed and consists of binary data according to [Section 4.8 in \[RFC4288\]](#).

Security considerations:

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

Interoperability considerations: none

Published specification: none

Applications that use this media type:

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

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: TBD

## **[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", "minptime", "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., maxaveragebitrate=20000). 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 allowed, DTX is not allowed



```
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 subject to this payload format description. The decoder **MUST** be capable to decode 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, dependent on the set values of the parameters the performance of the application may suffer. [[RFC3264](#)] defines the SDP offer-answer handling of the "ptime" parameter. The "maxptime" parameter **MUST** be handled in the same way.
- o The "minptime" parameter is a unidirectional receive-only parameters and typically will not compromise interoperability; however, dependent on the set values of the parameter the performance of the application may suffer and should be set with care.
- o The "maxplaybackrate" parameter is a unidirectional receive-only parameter that reflects limitations of the local receiver. The sender of the other side **SHOULD NOT** send with an audio bandwidth





higher than "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 bitrates, 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, dependent on the set value of the parameter the performance of the application may suffer and should 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.
- 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", "minptime", "maxplaybackrate", and "maxaveragebitrate" should 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 "minptime" of the payload format configuration are recommendations by the decoding side to ensure the best performance for the decoder. The decoder MUST be capable to accept any allowed packet sizes to ensure maximum compatibility.
- 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 may be provided if necessary by declaring multiple RTP payload types; however, the number of types should 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. The Opus payload format does not have any built-in security mechanisms. 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**

TBD

## 9. 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.
- [RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", [RFC 2974](#), October 2000.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", [RFC 3264](#), June 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.
- [RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", [BCP 13](#), [RFC 4288](#), December 2005.
- [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.





Authors' Addresses

Julian Spittka

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

Koen Vos

Skype Technologies S.A.

3210 Porter Drive

Palo Alto, CA 94304

USA

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

Jean-Marc Valin

Mozilla

650 Castro Street

Mountain View, CA 94041

USA

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

