Internet Engineering Task Force (IETF)
Request for Comments: 8088
Updates: 2736
Category: Informational
ISSN: 2070-1721

M. Westerlund
Ericsson

May 2017

How to Write an RTP Payload Format

Abstract

This document contains information on how best to write an RTP payload format specification. It provides reading tips, design practices, and practical tips on how to produce an RTP payload format specification quickly and with good results. A template is also included with instructions.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc8088.

Table of Contents

1. Introduction

RTP [RFC3550] payload formats define how a specific real-time data format is structured in the payload of an RTP packet. A real-time data format without a payload format specification cannot be transported using RTP. This creates an interest in many individuals/organizations with media encoders or other types of real-time data to define RTP payload formats. However, the specification of a well-designed RTP payload format is nontrivial and requires knowledge of both RTP and the real-time data format.

This document is intended to help any author of an RTP payload format specification make important design decisions, consider important features of RTP and RTP security, etc. The document is also intended to be a good starting point for any person with little experience in the IETF and/or RTP to learn the necessary steps.

This document extends and updates the information that is available in "Guidelines for Writers of RTP Payload Format Specifications" [RFC2736]. Since that RFC was written, further experience has been gained on the design and specification of RTP payload formats. Several new RTP profiles and robustness tools have been defined, and these need to be considered.

This document also discusses the possible venues for defining an RTP payload format: the IETF, other standards bodies, and proprietary ones.

Note, this document does discuss IETF, IANA, and RFC Editor processes and rules as they were when this document was published. This to make clear how the work to specify an RTP payload formats depends, uses, and interacts with these rules and processes. However, these rules and processes are subject to change and the formal rule and process specifications always takes precedence over what is written here.

1.1. Structure

This document has several different parts discussing different aspects of the creation of an RTP payload format specification. Section 3 discusses the preparations the author(s) should make before starting to write a specification. Section 4 discusses the different processes used when specifying and completing a payload format, with focus on working inside the IETF. Section 5 discusses the design of payload formats themselves in detail. Section 6 discusses current design trends and provides good examples of practices that should be followed when applicable. Following that, Section 7 provides a discussion on important sections in the RTP payload format

specification itself such as Security Considerations and IANA Considerations. This document ends with an appendix containing a template that can be used when writing RTP payload formats specifications.

2. Terminology

2.1. Definitions

RTP Stream: A sequence of RTP packets that together carry part or all of the content of a specific media (audio, video, text, or data whose form and meaning are defined by a specific real-time application) from a specific sender source within a given RTP session.

RTP Session: An association among a set of participants communicating with RTP. The distinguishing feature of an RTP session is that each session maintains a full, separate space of synchronization source (SSRC) identifiers. See also Section 3.3.1.

RTP Payload Format: The RTP payload format specifies how units of a specific encoded media are put into the RTP packet payloads and how the fields of the RTP packet header are used, thus enabling the format to be used in RTP applications.

A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources [RFC7656] defines many useful terms.

2.2. Abbreviations

ABNF: Augmented Backus-Naur Form [RFC5234]

ADU: Application Data Unit

ALF: Application Level Framing

ASM: Any-Source Multicast

BCP: Best Current Practice

I-D: Internet-Draft

IESG: Internet Engineering Steering Group

MTU: Maximum Transmission Unit

WG: Working Group

QoS: Quality of Service

RFC: Request For Comments

RTP: Real-time Transport Protocol

RTCP: RTP Control Protocol

RTT: Round-Trip Time

SSM: Source-Specific Multicast

2.3. Use of Normative Requirements Language

As this document is both Informational and instructional rather than a specification, this document does not use any RFC 2119 language and the use of "may", "should", "recommended", and "must" carries no special connotation.

3. Preparations

RTP is a complex real-time media delivery framework, and it has a lot of details that need to be considered when writing an RTP payload format. It is also important to have a good understanding of the media codec / format so that all of its important features and properties are considered. Only when one has sufficient understanding of both parts can one produce an RTP payload format of high quality. On top of this, one needs to understand the process within the IETF and especially the Working Group responsible for standardizing payload formats (currently the PAYLOAD WG) to go quickly from the initial idea stage to a finished RFC. This and the next sections help an author prepare himself in those regards.

3.1. Read and Understand the Media Coding Specification

It may be obvious, but it is necessary for an author of an RTP payload specification to have a solid understanding of the media to be transported. Important are not only the specifically spelled out transport aspects (if any) in the media coding specification, but also core concepts of the underlying technology. For example, an RTP payload format for video coded with inter-picture prediction will perform poorly if the payload designer does not take the use of inter-picture prediction into account. On the other hand, some (mostly older) media codecs offer error-resilience tools against bit errors, which, when misapplied over RTP, in almost all cases would only introduce overhead with no measurable return.

3.2. Recommended Reading

The following subsections list a number of documents. Not all need to be read in full detail. However, an author basically needs to be aware of everything listed below.

3.2.1. IETF Process and Publication

Newcomers to the IETF are strongly recommended to read the "Tao of the IETF" [TAO] that goes through most things that one needs to know about the IETF: the history, organizational structure, how the WGs and meetings work, etc.

It is very important to note and understand the IETF Intellectual Property Rights (IPR) policy that requires early disclosures based on personal knowledge from anyone contributing in IETF. The IETF policies associated with IPR are documented in BCP 78 [BCP78] (related to copyright, including software copyright, for example, code) and BCP 79 [BCP79] (related to patent rights). These rules may be different from other standardization organizations. For example, a person that has a patent or a patent application that he or she reasonably and personally believes to cover a mechanism that gets added to the Internet-Draft they are contributing to (e.g., by submitting the draft, posting comments or suggestions on a mailing list, or speaking at a meeting) will need to make a timely IPR disclosure. Read the above documents for the authoritative rules. Failure to follow the IPR rules can have dire implications for the specification and the author(s) as discussed in [RFC6701].

> Note: These IPR rules apply on what is specified in the RTP payload format Internet-Draft (and later RFC); an IPR that relates to a codec specification from an external body does not require IETF IPR disclosure. Informative text explaining the nature of the codec would not normally require an IETF IPR declaration. Appropriate IPR declarations for the codec itself would normally be found in files of the external body defining the codec, in accordance with that external body's own IPR rules.

The main part of the IETF process is formally defined in BCP 9 [BCP9]. BCP 25 [BCP25] describes the WG process, the relation between the IESG and the WG, and the responsibilities of WG Chairs and participants.

It is important to note that the RFC Series contains documents of several different publication streams as defined by The RFC Series and RFC Editor [RFC4844]. The most important stream for RTP payload formats authors is the IETF Stream. In this stream, the work of the IETF is published. The stream contains documents of several

different categories: Standards Track, Informational, Experimental, Best Current Practice, and Historic. "Standards Track" contains two maturity levels: Proposed Standard and Internet Standard [RFC6410]. A Standards Track document must start as a Proposed Standard; after successful deployment and operational experience with at least two implementations, it can be moved to an Internet Standard. The Independent Submission Stream could appear to be of interest as it provides a way of publishing documents of certain categories such as Experimental and Informational with a different review process. However, as long as IETF has a WG that is chartered to work on RTP payload formats, this stream should not be used.

As the content of a given RFC is not allowed to change once published, the only way to modify an RFC is to write and publish a new one that either updates or replaces the old one. Therefore, whether reading or referencing an RFC, it is important to consider both the Category field in the document header and to check if the RFC is the latest on the subject and still valid. One way of checking the current status of an RFC is to use the RFC Editor's RFC search page (https://www.rfc-editor.org/search), which displays the current status and which if any RFC has updated or obsoleted it. The RFC Editor search engine will also indicate if there exist any errata reports for the RFC. Any verified errata report contains issues of significant importance with the RFC; thus, they should be known prior to an update and replacement publication.

Before starting to write a draft, one should also read the Internet-Draft writing guidelines (http://www.ietf.org/ietf/1id-guidelines.txt), the I-D checklist (http://www.ietf.org/ID-Checklist.html), and the RFC Style Guide [RFC7322]. Another document that can be useful is "Guide for Internet Standards Writers" [RFC2360].

There are also a number of documents to consider in the process of writing drafts intended to become RFCs. These are important when writing certain types of text.

RFC 2606: When writing examples using DNS names in Internet-Drafts, those names shall be chosen from the example.com, example.net, and example.org domains.

RFC 3849: Defines the range of IPv6 unicast addresses (2001:DB8::/32) that should be used in any examples.

RFC 5737: Defines the ranges of IPv4 unicast addresses reserved for documentation and examples: 192.0.2.0/24, 198.51.100.0/24, and 203.0.113.0/24.

RFC 5234: Augmented Backus-Naur Form (ABNF) is often used when writing text field specifications. Not commonly used in RTP payload formats, but may be useful when defining media type parameters of some complexity.

3.2.2. RTP

The recommended reading for RTP consists of several different parts: design guidelines, the RTP protocol, profiles, robustness tools, and media-specific recommendations.

Any author of RTP payload formats should start by reading "Guidelines for Writers of RTP Payload Format Specifications" [RFC2736], which contains an introduction to the Application Level Framing (ALF) principle, the channel characteristics of IP channels, and design guidelines for RTP payload formats. The goal of ALF is to be able to transmit Application Data Units (ADUs) that are independently usable by the receiver in individual RTP packets, thus minimizing dependencies between RTP packets and the effects of packet loss.

Then, it is advisable to learn more about the RTP protocol, by studying the RTP specification "RTP: A Transport Protocol for Real-Time Applications" [RFC3550] and the existing profiles. As a complement to the Standards Track documents, there exists a book totally dedicated to RTP [CSP-RTP]. There exist several profiles for RTP today, but all are based on "RTP Profile for Audio and Video Conferences with Minimal Control" [RFC3551] (abbreviated as RTP/AVP). The other profiles that one should know about are "The Secure Real-time Transport Protocol (SRTP)" (RTP/SAVP) [RFC3711], "Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)" [RFC4585], and "Extended Secure Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)" [RFC5124]. It is important to understand RTP and the RTP/AVP profile in detail. For the other profiles, it is sufficient to have an understanding of what functionality they provide and the limitations they create.

A number of robustness tools have been developed for RTP. The tools are for different use cases and real-time requirements.

RFC 2198: "RTP Payload for Redundant Audio Data" [RFC2198] provides functionalities to transmit redundant copies of audio or text payloads. These redundant copies are sent together with a primary format in the same RTP payload. This format relies on the RTP timestamp to determine where data belongs in a sequence; therefore, it is usually most suitable to be used with audio. However, the RTP Payload format for T.140 [RFC4103] text format also uses this format. The format's major property is that it only preserves the timestamp of the redundant payloads, not the

original sequence number. This makes it unusable for most video formats. This format is also only suitable for media formats that produce relatively small RTP payloads.

RFC 6354: The "Forward-Shifted RTP Redundancy Payload Support" [RFC6354] is a variant of RFC 2198 that allows the redundant data to be transmitted prior to the original.

RFC 5109: The "RTP Payload Format for Generic Forward Error Correction" [RFC5109] provides an XOR-based Forward Error Correction (FEC) of the whole or parts of a number of RTP packets. This specification replaced the previous specification for XOR-based FEC [RFC2733]. These FEC packets are sent in a separate stream or as a redundant encoding using RFC 2198. This FEC scheme has certain restrictions in the number of packets it can protect. It is suitable for applications with low-to-medium delay tolerance with a limited amount of RTP packets.

RFC 6015: "RTP Payload Format for 1-D Interleaved Parity Forward Error Correction (FEC)" [RFC6015] provides a variant of the XOR-based Generic protection defined in [RFC2733]. The main difference is to use interleaving scheme on which packets gets included as source packets for a particular protection packet. The interleaving is defined by using every L packets as source data and then producing protection data over D number of packets. Thus, each block of D x L source packets will result in L number of Repair packets, each capable of repairing one loss. The goal is to provide better burst-error robustness when the packet rate is higher.

FEC Framework: "Forward Error Correction (FEC) Framework" [RFC6363] defines how to use FEC protection for arbitrary packet flows. This framework can be applied for RTP/RTCP packet flows, including using RTP for transmission of repair symbols, an example is in "RTP Payload Format for Raptor Forward Error Correction (FEC)" [RFC6682].

RTP Retransmission: The RTP retransmission scheme [RFC4588] is used for semi-reliability of the most important RTP packets in a RTP stream. The level of reliability between semi- and in-practice full reliability depends on the targeted properties and situation where parameters such as round-trip time (RTT) allowed additional overhead and allowable delay. It often requires the application to be quite delay tolerant as a minimum of one round-trip time plus processing delay is required to perform a retransmission. Thus, it is mostly suitable for streaming applications but may also be usable in certain other cases when operating in networks with short round-trip times.

RTP over TCP: RFC 4571 [RFC4571] defines how one sends RTP and RTCP packets over connection-oriented transports like TCP. If one uses TCP, one gets reliability for all packets but loses some of the real-time behavior that RTP was designed to provide. Issues with TCP transport of real-time media include head-of-line blocking and wasting resources on retransmission of data that is already late. TCP is also limited to point-to-point connections, which further restricts its applicability.

There have been both discussion and design of RTP payload formats, e.g., Adaptive Multi-Rate (AMR) and AMR Wideband (AMR-WB) [RFC4867], supporting the unequal error detection provided by UDP-Lite [RFC3828]. The idea is that by not having a checksum over part of the RTP payload one can allow bit errors from the lower layers. By allowing bit errors one can increase the efficiency of some link layers and also avoid unnecessary discarding of data when the payload and media codec can get at least some benefit from the data. The main issue is that one has no idea of the level of bit errors present in the unprotected part of the payload. This makes it hard or impossible to determine whether or not one can design something usable. Payload format designers are not recommended to consider features for unequal error detection using UDP-Lite unless very clear requirements exist.

There also exist some management and monitoring extensions.

RFC 2959: The RTP protocol Management Information Database (MIB) [RFC2959] that is used with SNMP [RFC3410] to configure and retrieve information about RTP sessions.

RFC 3611: The RTCP Extended Reports (RTCP XR) [RFC3611] consists of a framework for reports sent within RTCP. It can easily be extended by defining new report formats, which has and is occurring. The XRBLOCK WG in the IETF is chartered (at the time of writing) with defining new report formats. The list of specified formats is available in IANA's RTCP XR Block Type registry (http://www.iana.org/assignments/rtcp-xr-block-types/). The report formats that are defined in RFC 3611 provide report information on packet loss, packet duplication, packet reception times, RTCP statistics summary, and VoIP Quality. [RFC3611] also defines a mechanism that allows receivers to calculate the RTT to other session participants when used.

RMONMIB: The Remote Network Monitoring WG has defined a mechanism [RFC3577] based on usage of the MIB that can be an alternative to RTCP XR.

A number of transport optimizations have also been developed for use in certain environments. They are all intended to be transparent and do not require special consideration by the RTP payload format writer. Thus, they are primarily listed here for informational reasons.

RFC 2508: "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links" (CRTP) [RFC2508] is the first IETF-developed RTP header compression mechanism. It provides quite good compression; however, it has clear performance problems when subject to packet loss or reordering between compressor and decompressor.

RFCs 3095 and 5795: These are the base specifications of the robust header compression (ROHC) protocol version 1 [RFC3095] and version 2 [RFC5795]. This solution was created as a result of CRTP's lack of performance when compressed packets are subject to loss.

RFC 3545: Enhanced compressed RTP (E-CRTP) [RFC3545] was developed to provide extensions to CRTP that allow for better performance over links with long RTTs, packet loss, and/or reordering.

RFC 4170: "Tunneling Multiplexed Compressed RTP (TCRTP)" [RFC4170] is a solution that allows header compression within a tunnel carrying multiple multiplexed RTP flows. This is primarily used in voice trunking.

There exist a couple of different security mechanisms that may be used with RTP. By definition, generic mechanisms are transparent for the RTP payload format and do not need special consideration by the format designer. The main reason that different solutions exist is that different applications have different requirements; thus, different solutions have been developed. For more discussion on this, please see "Options for Securing RTP Sessions" [RFC7201] and "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202]. The main properties for an RTP security mechanism are to provide confidentiality for the RTP payload, integrity protection to detect manipulation of payload and headers, and source authentication. Not all mechanisms provide all of these features, a point that will need to be considered when a specific mechanisms is chosen.

The profile for Secure RTP - SRTP (RTP/SAVP) [RFC3711] and the derived profile (RTP/SAVPF [RFC5124]) are a solution that enables confidentiality, integrity protection, replay protection, and partial source authentication. It is the solution most commonly used with RTP at the time of writing this document. There exist several key-management solutions for SRTP, as well other choices, affecting the

security properties. For a more in-depth review of the options and solutions other than SRTP consult "Options for Securing RTP Sessions" [RFC7201].

3.3. Important RTP Details

This section reviews a number of RTP features and concepts that are available in RTP, independent of the payload format. The RTP payload format can make use of these when appropriate, and even affect the behavior (RTP timestamp and marker bit), but it is important to note that not all features and concepts are relevant to every payload format. This section does not remove the necessity to read up on RTP. However, it does point out a few important details to remember when designing a payload format.

3.3.1. The RTP Session

The definition of the RTP session from RFC 3550 is:

> An association among a set of participants communicating with RTP. A participant may be involved in multiple RTP sessions at the same time. In a multimedia session, each medium is typically carried in a separate RTP session with its own RTCP packets unless the encoding itself multiplexes multiple media into a single data stream. A participant distinguishes multiple RTP sessions by reception of different sessions using different pairs of destination transport addresses, where a pair of transport addresses comprises one network address plus a pair of ports for RTP and RTCP. All participants in an RTP session may share a common destination transport address pair, as in the case of IP multicast, or the pairs may be different for each participant, as in the case of individual unicast network addresses and port pairs. In the unicast case, a participant may receive from all other participants in the session using the same pair of ports, or may use a distinct pair of ports for each.
>
> The distinguishing feature of an RTP session is that each session maintains a full, separate space of SSRC identifiers (defined next). The set of participants included in one RTP session consists of those that can receive an SSRC identifier transmitted by any one of the participants either in RTP as the SSRC or a CSRC (also defined below) or in RTCP. For example, consider a three-party conference implemented using unicast UDP with each participant receiving from the other two on separate port pairs. If each participant sends RTCP feedback about data received from one other participant only back to that participant, then the conference is composed of three separate point-to-point RTP sessions. If each participant provides RTCP feedback about its

reception of one other participant to both of the other participants, then the conference is composed of one multi-party RTP session. The latter case simulates the behavior that would occur with IP multicast communication among the three participants.

The RTP framework allows the variations defined here, but a particular control protocol or application design will usually impose constraints on these variations.

3.3.2. RTP Header

The RTP header contains a number of fields. Two fields always require additional specification by the RTP payload format, namely the RTP timestamp and the marker bit. Certain RTP payload formats also use the RTP sequence number to realize certain functionalities, primarily related to the order of their application data units. The payload type is used to indicate the used payload format. The SSRC is used to distinguish RTP packets from multiple senders and media sources identifying the RTP stream. Finally, [RFC5285] specifies how to transport payload format independent metadata relating to the RTP packet or stream.

Marker Bit: A single bit normally used to provide important indications. In audio, it is normally used to indicate the start of a talk burst. This enables jitter buffer adaptation prior to the beginning of the burst with minimal audio quality impact. In video, the marker bit is normally used to indicate the last packet part of a frame. This enables a decoder to finish decoding the picture, where it otherwise may need to wait for the next packet to explicitly know that the frame is finished.

Timestamp: The RTP timestamp indicates the time instance the media sample belongs to. For discrete media like video, it normally indicates when the media (frame) was sampled. For continuous media, it normally indicates the first time instance the media present in the payload represents. For audio, this is the sampling time of the first sample. All RTP payload formats must specify the meaning of the timestamp value and the clock rates allowed. Selecting a timestamp rate is an active design choice and is further discussed in Section 5.2.

Discontinuous Transmission (DTX) that is common among speech codecs, typically results in gaps or jumps in the timestamp values due to that there is no media payload to transmit and the next used timestamp value represent the actual sampling time of the data transmitted.

Sequence Number: The sequence number is monotonically increasing and is set as the packet is sent. This property is used in many payload formats to recover the order of everything from the whole stream down to fragments of application data units (ADUs) and the order they need to be decoded. Discontinuous transmissions do not result in gaps in the sequence number, as it is monotonically increasing for each sent RTP packet.

Payload Type: The payload type is used to indicate, on a per-packet basis, which format is used. The binding between a payload type number and a payload format and its configuration are dynamically bound and RTP session specific. The configuration information can be bound to a payload type value by out-of-band signaling (Section 3.4). An example of this would be video decoder configuration information. Commonly, the same payload type is used for a media stream for the whole duration of a session. However, in some cases it may be necessary to change the payload format or its configuration during the session.

SSRC: The synchronization source (SSRC) identifier is normally not used by a payload format other than to identify the RTP timestamp and sequence number space a packet belongs to, allowing simultaneously reception of multiple media sources. However, some of the RTP mechanisms for improving resilience to packet loss uses multiple SSRCs to separate original data and repair or redundant data, as well as multi-stream transmission of scalable codecs.

Header Extensions: RTP payload formats often need to include metadata relating to the payload data being transported. Such metadata is sent as a payload header, at the start of the payload section of the RTP packet. The RTP packet also includes space for a header extension [RFC5285]; this can be used to transport payload format independent metadata, for example, an SMPTE time code for the packet [RFC5484]. The RTP header extensions are not intended to carry headers that relate to a particular payload format, and must not contain information needed in order to decode the payload.

The remaining fields do not commonly influence the RTP payload format. The padding bit is worth clarifying as it indicates that one or more bytes are appended after the RTP payload. This padding must be removed by a receiver before payload format processing can occur. Thus, it is completely separate from any padding that may occur within the payload format itself.

3.3.3. RTP Multiplexing

RTP has three multiplexing points that are used for different purposes. A proper understanding of this is important to correctly use them.

The first one is separation of RTP streams of different types or usages, which is accomplished using different RTP sessions. So, for example, in the common multimedia session with audio and video, RTP commonly multiplexes audio and video in different RTP sessions. To achieve this separation, transport-level functionalities are used, normally UDP port numbers. Different RTP sessions can also be used to realize layered scalability as it allows a receiver to select one or more layers for multicast RTP sessions simply by joining the multicast groups over which the desired layers are transported. This separation also allows different Quality of Service (QoS) to be applied to different media types. Use of multiple transport flows has potential issues due to NAT and firewall traversal. The choices how one applies RTP sessions as well as transport flows can affect the transport properties an RTP media stream experiences.

The next multiplexing point is separation of different RTP streams within an RTP session. Here, RTP uses the SSRC to identify individual sources of RTP streams. An example of individual media sources would be the capture of different microphones that are carried in an RTP session for audio, independently of whether they are connected to the same host or different hosts. There also exist cases where a single media source, is transmitted using multiple RTP streams. For each SSRC, a unique RTP sequence number and timestamp space is used.

The third multiplexing point is the RTP header payload type field. The payload type identifies what format the content in the RTP payload has. This includes different payload format configurations, different codecs, and also usage of robustness mechanisms like the one described in RFC 2198 [RFC2198].

3.3.4. RTP Synchronization

There are several types of synchronization, and we will here describe how RTP handles the different types:

Intra media: The synchronization within a media stream from a synchronization source (SSRC) is accomplished using the RTP timestamp field. Each RTP packet carries the RTP timestamp, which specifies the position in time of the media payload contained in this packet relative to the content of other RTP packets in the same RTP stream (i.e., a given SSRC). This is especially useful

in cases of discontinuous transmissions. Discontinuities can be caused by network conditions; when extensive losses occur the RTP timestamp tells the receiver how much later than previously received media the present media should be played out.

Inter-media: Applications commonly have a desire to use several media sources, possibly of different media types, at the same time. Thus, there exists a need to synchronize different media from the same endpoint. This puts two requirements on RTP: the possibility to determine which media are from the same endpoint and if they should be synchronized with each other and the functionality to facilitate the synchronization itself.

The first step in inter-media synchronization is to determine which SSRCs in each session should be synchronized with each other. This is accomplished by comparing the CNAME fields in the RTCP source description (SDES) packets. SSRCs with the same CNAME sent in any of multiple RTP sessions can be synchronized.

The actual RTCP mechanism for inter-media synchronization is based on the idea that each RTP stream provides a position on the media specific time line (measured in RTP timestamp ticks) and a common reference time line. The common reference time line is expressed in RTCP as a wall-clock time in the Network Time Protocol (NTP) format. It is important to notice that the wall-clock time is not required to be synchronized between hosts, for example, by using NTP [RFC5905]. It can even have nothing at all to do with the actual time; for example, the host system's up-time can be used for this purpose. The important factor is that all media streams from a particular source that are being synchronized use the same reference clock to derive their relative RTP timestamp time scales. The type of reference clock and its timebase can be signaled using RTP Clock Source Signaling [RFC7273].

Figure 1 illustrates how if one receives RTCP Sender Report (SR) packet P1 for one RTP stream and RTCP SR packet P2 for the other RTP stream, then one can calculate the corresponding RTP timestamp values for any arbitrary point in time T. However, to be able to do that, it is also required to know the RTP timestamp rates for each RTP stream currently used in the sessions.

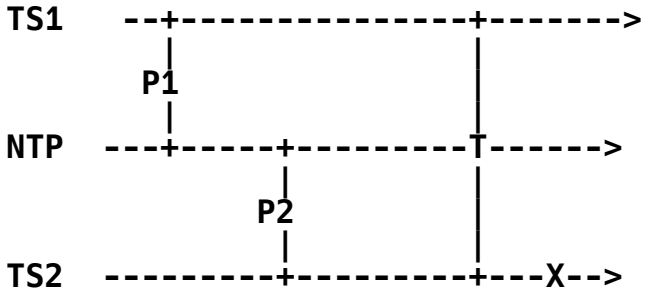

Figure 1: RTCP Synchronization

Assume that medium 1 uses an RTP timestamp clock rate of 16 kHz, and medium 2 uses a clock rate of 90 kHz. Then, TS1 and TS2 for point T can be calculated in the following way: TS1(T) = TS1(P1) + 16000 * (NTP(T)-NTP(P1)) and TS2(T) = TS2(P2) + 90000 * (NTP(T)-NTP(P2)). This calculation is useful as it allows the implementation to generate a common synchronization point for which all time values are provided (TS1(T), TS2(T) and T). So, when one wishes to calculate the NTP time that the timestamp value present in packet X corresponds to, one can do that in the following way: NTP(X) = NTP(T) + (TS2(X) - TS2(T))/90000.

Improved signaling for layered codecs and fast tune-in have been specified in "Rapid Synchronization for RTP Flows" [RFC6051].

Leap seconds are extra seconds added or seconds removed to keep our clocks in sync with the earth's rotation. Adding or removing seconds can impact the reference clock as discussed in "RTP and Leap Seconds" [RFC7164]; also, in cases where the RTP timestamp values are derived using the wall clock during the leap second event, errors can occur. Implementations need to consider leap seconds and should consider the recommendations in [RFC7164].

3.4. Signaling Aspects

RTP payload formats are used in the context of application signaling protocols such as SIP [RFC3261] using the Session Description Protocol (SDP) [RFC4566] with Offer/Answer [RFC3264], RTSP [RFC7826], or the Session Announcement Protocol [RFC2974]. These examples all use out-of-band signaling to indicate which type of RTP streams are desired to be used in the session and how they are configured. To be able to declare or negotiate the media format and RTP payload packetization, the payload format must be given an identifier. In addition to the identifier, many payload formats also have the need to signal further configuration information out-of-band for the RTP payloads prior to the media transport session.

The above examples of session-establishing protocols all use SDP, but other session description formats may be used. For example, there was discussion of a new XML-based session description format within the IETF (SDP-NG). In the end, the proposal did not get beyond draft protocol specification because of the enormous installed base of SDP implementations. However, to avoid locking the usage of RTP to SDP based out-of-band signaling, the payload formats are identified using a separate definition format for the identifier and associated parameters. That format is the media type.

3.4.1. Media Types

Media types [RFC6838] are identifiers originally created for identifying media formats included in email. In this usage, they were known as MIME types, where the expansion of the MIME acronym includes the word "mail". The term "media type" was introduced to reflect a broader usage, which includes HTTP [RFC7231], Message Session Relay Protocol (MSRP) [RFC4975], and many other protocols to identify arbitrary content carried within the protocols. Media types also provide a media hierarchy that fits RTP payload formats well. Media type names are of two parts and consist of content type and sub-type separated with a slash, e.g., 'audio/PCMA' or 'video/h263-2000'. It is important to choose the correct content-type when creating the media type identifying an RTP payload format. However, in most cases, there is little doubt what content type the format belongs to. Guidelines for choosing the correct media type and registration rules for media type names are provided in "Media Type Specifications and Registration Procedures" [RFC6838]. The additional rules for media types for RTP payload formats are provided in "Media Type Registration of RTP Payload Formats" [RFC4855].

Registration of the RTP payload name is something that is required to avoid name collision in the future. Note that "x-" names are not suitable for any documented format as they have the same problem with name collision and can't be registered. The list of already-registered media types can be found at <https://www.iana.org/assignments/media-types/media-types.xhtml>.

Media types are allowed any number of parameters, which may be required or optional for that media type. They are always specified on the form "name=value". There exist no restrictions on how the value is defined from the media type's perspective, except that parameters must have a value. However, the usage of media types in

SDP, etc., has resulted in the following restrictions that need to be followed to make media types usable for RTP-identifying payload formats:

1. Arbitrary binary content in the parameters is allowed, but it needs to be encoded so that it can be placed within text-based protocols. Base64 [RFC4648] is recommended, but for shorter content Base16 [RFC4648] may be more appropriate as it is simpler to interpret for humans. This needs to be explicitly stated when defining a media type parameter with binary values.

2. The end of the value needs to be easily found when parsing a message. Thus, parameter values that are continuous and not interrupted by common text separators, such as space and semicolon characters, are recommended. If that is not possible, some type of escaping should be used. Usage of quote (") is recommended; do not forget to provide a method of encoding any character used for quoting inside the quoted element.

3. A common representation form for the media type and its parameters is on a single line. In that case, the media type is followed by a semicolon-separated list of the parameter value pairs, e.g.:

   audio/amr octet-align=0; mode-set=0,2,5,7; mode-change-period=2

3.4.2. Mapping to SDP

Since SDP [RFC4566] is so commonly used as an out-of-band signaling protocol, a mapping of the media type into SDP exists. The details on how to map the media type and its parameters into SDP are described in [RFC4855]. However, this is not sufficient to explain how certain parameters must be interpreted, for example, in the context of Offer/Answer negotiation [RFC3264].

3.4.2.1. The Offer/Answer Model

The Offer/Answer (O/A) model allows SIP to negotiate which media formats and payload formats are to be used in a session and how they are to be configured. However, O/A does not define a default behavior; instead, it points out the need to define how parameters behave. To make things even more complex, the direction of media within a session has an impact on these rules, so that some cases may require separate descriptions for RTP streams that are send-only, receive-only, or both sent and received as identified by the SDP attributes a=sendonly, a=recvonly, and a=sendrecv. In addition, the usage of multicast adds further limitations as the same RTP stream is

delivered to all participants. If those multicast-imposed restrictions are too limiting for unicast, then separate rules for unicast and multicast will be required.

The simplest and most common O/A interpretation is that a parameter is defined to be declarative; i.e., the SDP Offer/Answer sending agent can declare a value and that has no direct impact on the other agent's values. This declared value applies to all media that are going to be sent to the declaring entity. For example, most video codecs have a level parameter that tells the other participants the highest complexity the video decoder supports. The level parameter can be declared independently by two participants in a unicast session as it will be the media sender's responsibility to transmit a video stream that fulfills the limitation the other side has declared. However, in multicast, it will be necessary to send a stream that follows the limitation of the weakest receiver, i.e., the one that supports the lowest level. To simplify the negotiation in these cases, it is common to require any answerer to a multicast session to take a yes or no approach to parameters.

A "negotiated" parameter is a different case, for which both sides need to agree on its value. Such a parameter requires the answerer to either accept it as it is offered or remove the payload type the parameter belonged to from its answer. The removal of the payload type from the answer indicates to the offerer the lack of support for the parameter values presented. An unfortunate implication of the need to use complete payload types to indicate each possible configuration so as to maximize the chances of achieving interoperability, is that the number of necessary payload types can quickly grow large. This is one reason to limit the total number of sets of capabilities that may be implemented.

The most problematic type of parameters are those that relate to the media the entity sends. They do not really fit the O/A model, but can be shoehorned in. Examples of such parameters can be found in the H.264 video codec's payload format [RFC6184], where the name of all parameters with this property starts with "sprop-". The issue with these parameters is that they declare properties for a RTP stream that the other party may not accept. The best one can make of the situation is to explain the assumption that the other party will accept the same parameter value for the media it will receive as the offerer of the session has proposed. If the answerer needs to change any declarative parameter relating to streams it will receive, then the offerer may be required to make a new offer to update the parameter values for its outgoing RTP stream.

Another issue to consider is the send-only RTP streams in offers. Parameters that relate to what the answering entity accepts to receive have no meaning other than to provide a template for the answer. It is worth pointing out in the specification that these really provide a set of parameter values that the sender recommends. Note that send-only streams in answers will need to indicate the offerer's parameters to ensure that the offerer can match the answer to the offer.

A further issue with Offer/Answer that complicates things is that the answerer is allowed to renumber the payload types between offer and answer. This is not recommended, but allowed for support of gateways to the ITU conferencing suite. This means that it must be possible to bind answers for payload types to the payload types in the offer even when the payload type number has been changed, and some of the proposed payload types have been removed. This binding must normally be done by matching the configurations originally offered against those in the answer. This may require specification in the payload format of which parameters that constitute a configuration, for example, as done in Section 8.2.2 of the H.264 RTP Payload format [RFC6184], which states: "The parameters identifying a media format configuration for H.264 are profile-level-id and packetization-mode".

3.4.2.2. Declarative Usage in RTSP and SAP

SAP (Session Announcement Protocol) [RFC2974] was experimentally used for announcing multicast sessions. Similar but better protocols are using SDP in a declarative style to configure multicast-based applications. Independently of the usage of Source-Specific Multicast (SSM) [RFC3569] or Any-Source Multicast (ASM), the SDP provided by these configuration delivery protocols applies to all participants. All media that is sent to the session must follow the RTP stream definition as specified by the SDP. This enables everyone to receive the session if they support the configuration. Here, SDP provides a one-way channel with no possibility to affect the configuration that the session creator has decided upon. Any RTP payload format that requires parameters for the send direction and that needs individual values per implementation or instance will fail in a SAP session for a multicast session allowing anyone to send.

Real-Time Streaming Protocol (RTSP) [RFC7826] allows the negotiation of transport parameters for RTP streams that are part of a streaming session between a server and client. RTSP has divided the transport parameters from the media configuration. SDP is commonly used for media configuration in RTSP and is sent to the client prior to session establishment, either through use of the DESCRIBE method or

by means of an out-of-band channel like HTTP, email, etc. The SDP is used to determine which RTP streams and what formats are being used prior to session establishment.

Thus, both SAP and RTSP use SDP to configure receivers and senders with a predetermined configuration for a RTP stream including the payload format and any of its parameters. All parameters are used in a declarative fashion. This can result in different treatment of parameters between Offer/Answer and declarative usage in RTSP and SAP. Any such difference will need to be spelled out by the payload format specification.

3.5. Transport Characteristics

The general channel characteristics that RTP flows experience are documented in Section 3 of "Guidelines for Writers of RTP Payload Format Specifications" [RFC2736]. The discussion below provides additional information.

3.5.1. Path MTU

At the time of writing, the most common IP Maximum Transmission Unit (MTU) in commonly deployed link layers is 1500 bytes (Ethernet data payload). However, there exist both links with smaller MTUs and links with much larger MTUs. An example for links with small MTU size is older generation cellular links. Certain parts of the Internet already support an IP MTU of 8000 bytes or more, but these are limited islands. The most likely places to find MTUs larger than 1500 bytes are within enterprise networks, university networks, data centers, storage networks, and over high capacity (10 Gbps or more) links. There is a slow, ongoing evolution towards larger MTU sizes. However, at the same time, it has become common to use tunneling protocols, often multiple ones, whose overhead when added together can shrink the MTU significantly. Thus, there exists a need both to consider limited MTUs as well as enable support of larger MTUs. This should be considered in the design, especially in regard to features such as aggregation of independently decodable data units.

3.5.2. Different Queuing Algorithms

Routers and switches on the network path between an IP sender and a particular receiver can exhibit different behaviors affecting the end-to-end characteristics. One of the more important aspects of this is queuing behavior. Routers and switches have some amount of queuing to handle temporary bursts of data that designated to leave the switch or router on the same egress link. A queue, when not empty, results in an increased path delay.

The implementation of the queuing affects the delay and also how congestion signals (Explicit Congestion Notification (ECN) [RFC6679] or packet drops) are provided to the flow. The other aspects are if the flow shares the queue with other flows and how the implementation affects the flow interaction. This becomes important, for example, when real-time flows interact with long-lived TCP flows. TCP has a built-in behavior in its congestion control that strives to fill the buffer; thus, all flows sharing the buffer experienced the delay build up.

A common, but quite poor, queue-handling mechanism is tail-drop, i.e., only drop packets when the incoming packet doesn't fit in the queue. If a bad queuing algorithm is combined with too much queue space, the queuing time can grow to be very significant and can even become multiple seconds. This is called "bufferbloat" [BLOAT]. Active Queue Management (AQM) is a term covering mechanisms that try to do something smarter by actively managing the queue, for example, sending congestion signals earlier by dropping packets earlier in the queue. The behavior also affects the flow interactions. For example, Random Early Detection (RED) [RED] selects which packet(s) to drop randomly. This gives flows that have more packets in the queue a higher probability to experience the packet loss (congestion signal). There is ongoing work in the IETF WG AQM to find suitable mechanisms to recommend for implementation and reduce the use of tail-drop.

3.5.3. Quality of Service

Using best-effort Internet has no guarantees for the path's properties. QoS mechanisms are intended to provide the possibility to bound the path properties. Where Diffserv [RFC2475] markings affect the queuing and forwarding behaviors of routers, the mechanism provides only statistical guarantees and care in how much marked packets of different types that are entering the network. Flow-based QoS, like IntServ [RFC1633], has the potential for stricter guarantees as the properties are agreed on by each hop on the path, at the cost of per-flow state in the network.

4. Standardization Process for an RTP Payload Format

This section discusses the recommended process to produce an RTP payload format in the described venues. This is to document the best current practice on how to get a well-designed and specified payload format as quickly as possible. For specifications that are defined by standards bodies other than the IETF, the primary milestone is the registration of the media type for the RTP payload format. For

proprietary media formats, the primary goal depends on whether interoperability is desired at the RTP level. However, there is also the issue of ensuring best possible quality of any specification.

4.1. IETF

For all standardized media formats, it is recommended that the payload format be specified in the IETF. The main reason is to provide an openly available RTP payload format specification that has been reviewed by people experienced with RTP payload formats. At the time of writing, this work is done in the PAYLOAD Working Group (WG), but that may change in the future.

4.1.1. Steps from Idea to Publication

There are a number of steps that an RTP payload format should go through from the initial idea until it is published. This also documents the process that the PAYLOAD WG applies when working with RTP payload formats.

Idea: Determine the need for an RTP payload format as an IETF specification.

Initial effort: Using this document as a guideline, one should be able to get started on the work. If one's media codec doesn't fit any of the common design patterns or one has problems understanding what the most suitable way forward is, then one should contact the PAYLOAD WG and/or the WG Chairs. The goal of this stage is to have an initial individual draft. This draft needs to focus on the introductory parts that describe the real-time media format and the basic idea on how to packetize it. Not all the details are required to be filled in. However, the security chapter is not something that one should skip, even initially. From the start, it is important to consider any serious security risks that need to be solved. The first step is completed when one has a draft that is sufficiently detailed for a first review by the WG. The less confident one is of the solution, the less work should be spent on details; instead, concentrate on the codec properties and what is required to make the packetization work.

Submission of the first version: When one has performed the above, one submits the draft as an individual draft (https://datatracker.ietf.org/submit/). This can be done at any time, except for a period prior to an IETF meeting (see important dates related to the next IETF meeting for draft submission cutoff date). When the Internet-Draft announcement has been sent out on

the draft announcement list (https://www.ietf.org/mailman/listinfo/I-D-Announce), forward it to the PAYLOAD WG (https://www.ietf.org/mailman/listinfo/payload) and request that it be reviewed. In the email, outline any issues the authors currently have with the design.

Iterative improvements: Taking the feedback received into account, one updates the draft and tries resolve issues. New revisions of the draft can be submitted at any time (again except for a short period before meetings). It is recommended to submit a new version whenever one has made major updates or has new issues that are easiest to discuss in the context of a new draft version.

Becoming a WG document: Given that the definition of RTP payload formats is part of the PAYLOAD WG's charter, RTP payload formats that are going to be published as Standards Track RFCs need to become WG documents. Becoming a WG document means that the WG Chairs or an appointed document shepherd are responsible for administrative handling, for example, issuing publication requests. However, be aware that making a document into a WG document changes the formal ownership and responsibility from the individual authors to the WG. The initial authors normally continue being the document editors, unless unusual circumstances occur. The PAYLOAD WG accepts new RTP payload formats based on their suitability and document maturity. The document maturity is a requirement to ensure that there are dedicated document editors and that there exists a good solution.

Iterative improvements: The updates and review cycles continue until the draft has reached the level of maturity suitable for publication. The authors are responsible for judging when the document is ready for the next step, most likely WG Last Call, but they can ask the WG chairs or Shepherd.

WG Last Call: A WG Last Call of at least two weeks is always performed for payload formats in the PAYLOAD WG (see Section 7.4 of [RFC2418]). The authors request WG Last Call for a draft when they think it is mature enough for publication. The WG Chairs or shepherd perform a review to check if they agree with the authors' assessment. If the WG Chairs or shepherd agree on the maturity, the WG Last Call is announced on the WG mailing list. If there are issues raised, these need to be addressed with an updated draft version. For any more substantial changes to the draft, a new WG Last Call is announced for the updated version. Minor changes, like editorial fixes, can be progressed without an additional WG Last Call.

Publication requested: For WG documents, the WG Chairs or shepherd request publication of the draft after it has passed WG Last Call. After this, the approval and publication process described in BCP 9 [BCP9] is performed. The status after the publication has been requested can be tracked using the IETF Datatracker [TRACKER]. Documents do not expire as they normally do after publication has been requested, so authors do not have to issue keep-alive updates. In addition, any submission of document updates requires the approval of WG Chair(s). The authors are commonly asked to address comments or issues raised by the IESG. The authors also do one last review of the document immediately prior to its publication as an RFC to ensure that no errors or formatting problems have been introduced during the publication process.

4.1.2. WG Meetings

WG meetings are for discussing issues, not presentations. This means that most RTP payload formats should never need to be discussed in a WG meeting. RTP payload formats that would be discussed are either those with controversial issues that failed to be resolved on the mailing list or those including new design concepts worth a general discussion.

There exists no requirement to present or discuss a draft at a WG meeting before it becomes published as an RFC. Thus, even authors who lack the possibility to go to WG meetings should be able to successfully specify an RTP payload format in the IETF. WG meetings may become necessary only if the draft gets stuck in a serious debate that cannot easily be resolved.

4.1.3. Draft Naming

To simplify the work of the PAYLOAD WG Chairs and WG members, a specific Internet-Draft file-naming convention shall be used for RTP payload formats. Individual submissions shall be named using the template: draft-<lead author family name>-payload-rtp-<descriptive name>-<version>. The WG documents shall be named according to this template: draft-ietf-payload-rtp-<descriptive name>-<version>. The inclusion of "payload" in the draft file name ensures that the search for "payload-" will find all PAYLOAD-related drafts. Inclusion of "rtp" tells us that it is an RTP payload format draft. The descriptive name should be as short as possible while still describing what the payload format is for. It is recommended to use the media format or codec abbreviation. Please note that the version must start at 00 and is increased by one for each submission to the IETF secretary of the draft. No version numbers may be skipped. For more details on draft naming, please see Section 7 of [ID-GUIDE].

4.1.4. Writing Style

When writing an Internet-Draft for an RTP payload format, one should observe some few considerations (that may be somewhat divergent from the style of other IETF documents and/or the media coding spec's author group may use):

Include Motivations: In the IETF, it is common to include the motivation for why a particular design or technical path was chosen. These are not long statements: a sentence here and there explaining why suffice.

Use the Defined Terminology: There exists defined terminology both in RTP and in the media codec specification for which the RTP payload format is designed. A payload format specification needs to use both to make clear the relation of features and their functions. It is unwise to introduce or, worse, use without introduction, terminology that appears to be more accessible to average readers but may miss certain nuances that the defined terms imply. An RTP payload format author can assume the reader to be reasonably familiar with the terminology in the media coding specification.

Keeping It Simple: The IETF has a history of specifications that are focused on their main usage. Historically, some RTP payload formats have a lot of modes and features, while the actual deployments have only included the most basic features that had very clear requirements. Time and effort can be saved by focusing on only the most important use cases and keeping the solution simple. An extension mechanism should be provided to enable backward-compatible extensions, if that is an organic fit.

Normative Requirements: When writing specifications, there is commonly a need to make it clear when something is normative and at what level. In the IETF, the most common method is to use "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119], which defines the meaning of "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL".

4.1.5. How to Speed Up the Process

There a number of ways to lose a lot of time in the above process. This section discusses what to do and what to avoid.

- Do not update the draft only for the meeting deadline. An update to each meeting automatically limits the draft to three updates per year. Instead, ignore the meeting schedule and publish new versions as soon as possible.

- Try to avoid requesting reviews when people are busy, like the few weeks before a meeting. It is actually more likely that people have time for them directly after a meeting.

- Perform draft updates quickly. A common mistake is that the authors let the draft slip. By performing updates to the draft text directly after getting resolution on an issue, things speed up. This minimizes the delay that the author has direct control over. The time taken for reviews, responses from Area Directors and WG Chairs, etc., can be much harder to speed up.

- Do not fail to take human nature into account. It happens that people forget or need to be reminded about tasks. Send a kind reminder to the people you are waiting for if things take longer than expected. Ask people to estimate when they expect to fulfill the requested task.

- Ensure there is enough review. It is common that documents take a long time and many iterations because not enough review is performed in each iteration. To improve the amount of review you get on your own document, trade review time with other document authors. Make a deal with some other document author that you will review their draft if they review yours. Even inexperienced reviewers can help with language, editorial, or clarity issues. Also, try approaching the more experienced people in the WG and getting them to commit to a review. The WG Chairs cannot, even if desirable, be expected to review all versions. Due to workload, the Chairs may need to concentrate on key points in a draft evolution like checking on initial submissions, a draft's readiness to become a WG document, or its readiness for WG Last Call.

4.2. Other Standards Bodies

Other standards bodies may define RTP payloads in their own specifications. When they do this, they are strongly recommended to contact the PAYLOAD WG Chairs and request review of the work. It is recommended that at least two review steps are performed. The first

should be early in the process when more fundamental issues can be easily resolved without abandoning a lot of effort. Then, when nearing completion, but while it is still possible to update the specification, a second review should be scheduled. In that pass, the quality can be assessed; hopefully, no updates will be needed. Using this procedure can avoid both conflicting definitions and serious mistakes, like breaking certain aspects of the RTP model.

RTP payload media types may be registered in the standards tree by other standards bodies. The requirements on the organization are outlined in the media types registration documents [RFC4855] and [RFC6838]). This registration requires a request to the IESG, which ensures that the filled-in registration template is acceptable. To avoid last-minute problems with these registrations the registration template must be sent for review both to the PAYLOAD WG and the media types list (ietf-types@iana.org) and is something that should be included in the IETF reviews of the payload format specification.

4.3. Proprietary and Vendor Specific

Proprietary RTP payload formats are commonly specified when the real-time media format is proprietary and not intended to be part of any standardized system. However, there are reasons why also proprietary formats should be correctly documented and registered:

- Usage in a standardized signaling environment, such as SIP/SDP. RTP needs to be configured with the RTP profiles, payload formats, and their payload types being used. To accomplish this, it is desirable to have registered media type names to ensure that the names do not collide with those of other formats.

- Sharing with business partners. As RTP payload formats are used for communication, situations often arise where business partners would like to support a proprietary format. Having a well-written specification of the format will save time and money for both parties, as interoperability will be much easier to accomplish.

- To ensure interoperability between different implementations on different platforms.

To avoid name collisions, there is a central registry keeping track of the registered media type names used by different RTP payload formats. When it comes to proprietary formats, they should be registered in the vendor's own tree. All vendor-specific registrations use sub-type names that start with "vnd.<vendor-name>". Names in the vendor's own tree are not required to be registered with IANA. However, registration [RFC6838] is recommended if the media type is used at all in public environments.

If interoperability at the RTP level is desired, a payload type specification should be standardized in the IETF following the process described above. The IETF does not require full disclosure of the codec when defining an RTP payload format to carry that codec, but a description must be provided that is sufficient to allow the IETF to judge whether the payload format is well designed. The media type identifier assigned to a standardized payload format of this sort will lie in the standards tree rather than the vendor tree.

4.4. Joint Development of Media Coding Specification and RTP Payload Format

In the last decade, there have been a few cases where the media codec and the associated RTP payload format have been developed concurrently and jointly. Developing the two specs not only concurrently but also jointly, in close cooperation with the group developing the media codec, allows one to leverage the benefits joint source/channel coding can provide. Doing so has historically resulted in well-performing payload formats and in success of both the media coding specification and associated RTP payload format. Insofar, whenever the opportunity presents it, it may be useful to closely keep the media coding group in the loop (through appropriate liaison means whatever those may be) and influence the media coding specification to be RTP friendly. One example for such a media coding specification is H.264, where the RTP payload header co-serves as the H.264 NAL unit header and vice versa, and is documented in both specifications.

5. Designing Payload Formats

The best summary of payload format design is KISS (Keep It Simple, Stupid). A simple payload format is easier to review for correctness, easier to implement, and has low complexity. Unfortunately, contradictory requirements sometimes make it hard to do things simply. Complexity issues and problems that occur for RTP payload formats are:

Too many configurations: Contradictory requirements lead to the result that one configuration is created for each conceivable case. Such contradictory requirements are often between functionality and bandwidth. This outcome has two big disadvantages; First all configurations need to be implemented. Second, the user application must select the most suitable configuration. Selecting the best configuration can be very difficult and, in negotiating applications, this can create interoperability problems. The recommendation is to try to select

a very limited set of configurations (preferably one) that perform well for the most common cases and are capable of handling the other cases, but maybe not that well.

Hard to implement: Certain payload formats may become difficult to implement both correctly and efficiently. This needs to be considered in the design.

Interaction with general mechanisms: Special solutions may create issues with deployed tools for RTP, such as tools for more robust transport of RTP. For example, a requirement for an unbroken sequence number space creates issues for mechanisms relying on payload type switching interleaving media-independent resilience within a stream.

5.1. Features of RTP Payload Formats

There are a number of common features in RTP payload formats. There is no general requirement to support these features; instead, their applicability must be considered for each payload format. In fact, it may be that certain features are not even applicable.

5.1.1. Aggregation

Aggregation allows for the inclusion of multiple Application Data Units (ADUs) within the same RTP payload. This is commonly supported for codecs that produce ADUs of sizes smaller than the IP MTU. One reason for the use of aggregation is the reduction of header overhead (IP/UDP/RTP headers). When setting into relation the ADU size and the MTU size, do remember that the MTU may be significantly larger than 1500 bytes. An MTU of 9000 bytes is available today and an MTU of 64k may be available in the future. Many speech codecs have the property of ADUs of a few fixed sizes. Video encoders may generally produce ADUs of quite flexible sizes. Thus, the need for aggregation may be less. But some codecs produce small ADUs mixed with large ones, for example, H.264 Supplemental Enhancement Information (SEI) messages. Sending individual SEI message in separate packets are not efficient compared to combing the with other ADUs. Also, some small ADUs are, within the media domain, semantically coupled to the larger ADUs (for example, in-band parameter sets in H.264 [RFC6184]). In such cases, aggregation is sensible, even if not required from a payload/header overhead viewpoint. There also exist cases when the ADUs are pre-produced and can't be adopted to a specific networks MTU. Instead, their packetization needs to be adopted to the network. All above factors should be taken into account when deciding on the inclusion of aggregation, and weighting its benefits

against the complexity of defining them (which can be significant especially when aggregation is performed over ADUs with different playback times).

The main disadvantage of aggregation, beyond implementation complexity, is the extra delay introduced (due to buffering until a sufficient number of ADUs have been collected at the sender) and reduced robustness against packet loss. Aggregation also introduces buffering requirements at the receiver.

5.1.2. Fragmentation

If the real-time media format has the property that it may produce ADUs that are larger than common MTU sizes, then fragmentation support should be considered. An RTP payload format may always fall back on IP fragmentation; however, as discussed in RFC 2736, this has some drawbacks. Perhaps the most important reason to avoid IP fragmentation is that IP fragmented packets commonly are discarded in the network, especially by NATs or firewalls. The usage of fragmentation at the RTP payload format level allows for more efficient usage of RTP packet loss recovery mechanisms. It may also in some cases also allow better usage of partial ADUs by doing media specific fragmentation at media-specific boundaries. In use cases where the ADUs are pre-produced and can't be adopted to the network's MTU size, support for fragmentation can be crucial.

5.1.3. Interleaving and Transmission Rescheduling

Interleaving has been implemented in a number of payload formats to allow for less quality reduction when packet loss occurs. When losses are bursty and several consecutive packets are lost, the impact on quality can be quite severe. Interleaving is used to convert that burst loss to several spread-out individual packet losses. It can also be used when several ADUs are aggregated in the same packets. A loss of an RTP packet with several ADUs in the payload has the same effect as a burst loss if the ADUs would have been transmitted in individual packets. To reduce the burstiness of the loss, the data present in an aggregated payload may be interleaved, thus, spreading the loss over a longer time period.

A requirement for doing interleaving within an RTP payload format is the aggregation of multiple ADUs. For formats that do not use aggregation, there is still a possibility of implementing a transmission order rescheduling mechanism. That has the effect that the packets transmitted consecutively originate from different points in the RTP stream. This can be used to mitigate burst losses, which may be useful if one transmits packets at frequent intervals. However, it may also be used to transmit more significant data

earlier in combination with RTP retransmission to allow for more graceful degradation and increased possibility to receive the most important data, e.g., intra frames of video.

The drawback of interleaving is the significantly increased transmission buffering delay, making it less useful for low-delay applications. It may also create significant buffering requirements on the receiver. That buffering is also problematic, as it is usually difficult to indicate when a receiver may start consume data and still avoid buffer under run caused by the interleaving mechanism itself. Transmission rescheduling is only useful in a few specific cases, as in streaming with retransmissions. The potential gains must be weighed against the complexity of these schemes.

5.1.4. Media Back Channels

A few RTP payload formats have implemented back channels within the media format. Those have been for specific features, like the AMR [RFC4867] codec mode request (CMR) field. The CMR field is used in the operation of gateways to circuit-switched voice to allow an IP terminal to react to the circuit-switched network's need for a specific encoder mode. A common motivation for media back channels is the need to have signaling in direct relation to the media or the media path.

If back channels are considered for an RTP payload format they should be for a specific requirements which cannot be easily satisfied by more generic mechanisms within RTP or RTCP.

5.1.5. Media Scalability

Some codecs support various types of media scalability, i.e. some data of a RTP stream may be removed to adapt the media's properties, such as bitrate and quality. The adaptation may be applied in the following dimensions of the media:

Temporal: For most video codecs it is possible to adapt the frame rate without any specific definition of a temporal scalability mode, e.g., for H.264 [RFC6184]. In these cases, the sender changes which frames it delivers and the RTP timestamp makes it clear the frame interval and each frames relative capture time. H.264 Scalable Video Coding (SVC) [RFC6190] has more explicit support for temporal scalability.

Spatial: Video codecs supporting scalability may adapt the resolution, e.g., in SVC [RFC6190].

Quality: The quality of the encoded stream may be scaled by adapting the accuracy of the coding process, as, e.g. possible with Signal to Noise Ratio (SNR) fidelity scalability of SVC [RFC6190].

At the time of writing this document, codecs that support scalability have a bit of a revival. It has been realized that getting the required functionality for supporting the features of the media stream into the RTP framework is quite challenging. One of the recent examples for layered and scalable codecs is SVC [RFC6190].

SVC is a good example for a payload format supporting media scalability features, which have been in its basic form already included in RTP. A layered codec supports the dropping of data parts of a RTP stream, i.e., RTP packets may not be transmitted or forwarded to a client in order to adapt the RTP streams bitrate as well as the received encoded stream's quality, while still providing a decodable subset of the encoded stream to a client. One example for using the scalability feature may be an RTP Mixer (Multipoint Control Unit) [RFC7667], which controls the rate and quality sent out to participants in a communication based on dropping RTP packets or removing part of the payload. Another example may be a transport channel, which allows for differentiation in Quality of Service (QoS) parameters based on RTP sessions in a multicast session. In such a case, the more important packets of the scalable encoded stream (base layer) may get better QoS parameters than the less important packets (enhancement layer) in order to provide some kind of graceful degradation. The scalability features required for allowing an adaptive transport, as described in the two examples above, are based on RTP multiplexing in order to identify the packets to be dropped or transmitted/forwarded. The multiplexing features defined for Scalable Video Coding [RFC6190] are:

Single Session Transmission (SST), where all media layers of the media are transported as a single synchronization source (SSRC) in a single RTP session; as well as

Multi-Session Transmission (MST), which should more accurately be called multi-stream transmission, where different media layers or a set of media layers are transported in different RTP streams, i.e., using multiple sources (SSRCs).

In the first case (SST), additional in-band as well as out-of-band signaling is required in order to allow identification of packets belonging to a specific media layer. Furthermore, an adaptation of the encoded stream requires dropping of specific packets in order to provide the client with a compliant encoded stream. In case of using encryption, it is typically required for an adapting network device

to be in the security context to allow packet dropping and providing an intact RTP session to the client. This typically requires the network device to be an RTP mixer.

In general, having a media-unaware network device dropping excessive packets will be more problematic than having a Media-Aware Network Entity (MANE). First is the need to understand the media format and know which ADUs or payloads belong to the layers, that no other layer will be dependent on after the dropping. Second, if the MANE can work as an RTP mixer or translator, it can rewrite the RTP and RTCP in such a way that the receiver will not suspect unintentional RTP packet losses needing repair actions. This as the receiver can't determine if a lost packet was an important base layer packet or one of the less important extension layers.

In the second case (MST), the RTP packet streams can be sent using a single or multiple RTP session, and thus transport flows, e.g., on different multicast groups. Transmitting the streams in different RTP sessions, then the out-of-band signaling typically provides enough information to identify the media layers and its properties. The decision on dropping packets is based on the Network Address that identifies the RTP session to be dropped. In order to allow correct data provisioning to a decoder after reception from different sessions, data realignment mechanisms are required. In some cases, existing generic tools, as described below, can be employed to enable such realignment; when those generic mechanisms are sufficient, they should be used. For example, "Rapid Synchronisation for RTP Flows" [RFC6051], uses existing RTP mechanisms, i.e. the NTP timestamp, to ensure timely inter-session synchronization. Another is the signaling feature for indicating dependencies of RTP sessions in SDP, as defined in the Media Decoding Dependency Grouping in SDP [RFC5583].

Using MST within a single RTP session is also possible and allows stream level handling instead of looking deeper into the packets by a MANE. However, transport flow-level properties will be the same unless packet based mechanisms like Diffserv is used.

When QoS settings, e.g., Diffserv markings, are used to ensure that the extension layers are dropped prior the base layer the receiving endpoint has the benefit in MST to know which layer or set of layers the missing packets belong to as it will be bound to different RTP sessions or RTP packet streams (SSRCs), thus, explicitly indicating the importance of the loss.

5.1.6. High Packet Rates

Some media codecs require high packet rates; in these cases, the RTP sequence number wraps too quickly. As a rule of thumb, it must not be possible to wrap the sequence number space within at least three RTCP reporting intervals. As the reporting interval can vary widely due to configuration and session properties, and also must take into account the randomization of the interval, one can use the TCP maximum segment lifetime (MSL), i.e., 2 minutes, in ones consideration. If earlier wrapping may occur, then the payload format should specify an extended sequence number field to allow the receiver to determine where a specific payload belongs in the sequence, even in the face of extensive reordering. The RTP payload format for uncompressed video [RFC4175] can be used as an example for such a field.

RTCP is also affected by high packet rates. For RTCP mechanisms that do not use extended counters, there is significant risk that they wrap multiple times between RTCP reporting or feedback; thus, producing uncertainty about which packet(s) are referenced. The payload designer can't effect the RTCP packet formats used and their design, but can note this considerations when configuring RTCP bandwidth and reporting intervals to avoid to wrapping issues.

5.2. Selecting Timestamp Definition

The RTP timestamp is an important part and has two design choices associated with it. The first is the definition that determines what the timestamp value in a particular RTP packet will be, the second is which timestamp rate should be used.

The timestamp definition needs to explicitly define what the timestamp value in the RTP packet represent for a particular payload format. Two common definitions are used; for discretely sampled media, like video frames, the sampling time of the earliest included video frame which the data represent (fully or partially) is used; for continuous media like audio, the sampling time of the earliest sample which the payload data represent. There exist cases where more elaborate or other definitions are used.

RTP payload formats with a timestamp definition that results in no or little correlation between the media time instance and its transmission time cause the RTCP jitter calculation to become unusable due to the errors introduced on the sender side. A common example is a payload format for a video codec where the RTP timestamp represents the capture time of the video frame, but frames are large

enough that multiple RTP packets need to be sent for each frame spread across the framing interval. It should be noted whether or not the payload format has this property.

An RTP payload format also needs to define what timestamp rates, or clock rates (as it is also called), may be used. Depending on the RTP payload format, this may be a single rate or multiple ones or theoretically any rate. So what needs to be considered when selecting a rate?

The rate needs be selected so that one can determine where in the time line of the media a particular sample (e.g., individual audio sample, or video frame) or set of samples (e.g., audio frames) belong. To enable correct synchronization of this data with previous frames, including over periods of discontinuous transmission or irregularities.

For audio, it is common to require audio sample accuracy. Thus, one commonly selects the input sampling rate as the timestamp rate. This can, however, be challenging for audio codecs that support multiple different sampling frequencies, either as codec input or being used internally but effecting output, for example, frame duration. Depending on how one expects to use these different sampling rates one can allow multiple timestamp rates, each matching a particular codec input or sampling rate. However, due to the issues with using multiple different RTP timestamp rates for the same source (SSRC) [RFC7160], this should be avoided if one expects to need to switch between modes.

Then, an alternative is to find a common denominator frequency between the different modes, e.g., OPUS [RFC7587] that uses 48 kHz. If the different modes uses or can use a common input/output frequency, then selecting this also needs to be considered. However, it is important to consider all aspects as the case of AMR-WB+ [RFC4352] illustrates. AMR-WB+'s RTP timestamp rate has the very unusual value of 72 kHz, despite the fact that output normally is at a sample rate of 48kHz. The design is motivated by the media codec's production of a large range of different frame lengths in time perspective. The 72 kHz timestamp rate is the smallest found value that would make all of the frames the codec could produce result in an integer frame length in RTP timestamp ticks. This way, a receiver can always correctly place the frames in relation to any other frame, even when the frame length changes. The downside is that the decoder outputs for certain frame lengths are, in fact, partial samples. The result is that the output in samples from the codec will vary from frame to frame, potentially making implementation more difficult.

Video codecs have commonly been using 90 kHz; the reason is this is a common denominator between the usually used frame rates such as 24, 25, 30, 50 and 60, and NTSC's odd 29.97 Hz. There does, however, exist at least one exception in the payload format for SMPTE 292M video [RFC3497] that uses a clock rate of 148.5 MHz. The reason here is that the timestamp then identify the exact start sample within a video frame.

Timestamp rates below 1000 Hz are not appropriate, because this will cause a resolution too low in the RTCP measurements that are expressed in RTP timestamps. This is the main reason that the text RTP payload formats, like T.140 [RFC4103], use 1000 Hz.

6. Noteworthy Aspects in Payload Format Design

This section provides a few examples of payload formats that are worth noting for good or bad design in general or in specific details.

6.1. Audio Payloads

The AMR [RFC4867], AMR-WB [RFC4867], EVRC [RFC3558], SMV [RFC3558] payload formats are all quite similar. They are all for frame-based audio codecs and use a table of contents structure. Each frame has a table of contents entry that indicates the type of the frame and if additional frames are present. This is quite flexible, but produces unnecessary overhead if the ADU is of fixed size and if, when aggregating multiple ADUs, they are commonly of the same type. In that case, a solution like the one in AMR-WB+ [RFC4352] may be more suitable.

The RTP payload format for MIDI [RFC6295] contains some interesting features. MIDI is an audio format sensitive to packet losses, as the loss of a "note off" command will result in a note being stuck in an "on" state. To counter this, a recovery journal is defined that provides a summarized state that allows the receiver to recover from packet losses quickly. It also uses RTCP and the reported highest sequence number to be able to prune the state the recovery journal needs to contain. These features appear limited in applicability to media formats that are highly stateful and primarily use symbolic media representations.

There exists a security concern with variable bitrate audio and speech codecs that changes their payload length based on the input data. This can leak information, especially in structured communication like a speech recognition prompt service that asks people to enter information verbally. This issue also exists to some degree for discontinuous transmission as that allows the length of

phrases to be determined. The issue is further discussed in "Guidelines for the Use of Variable Bit Rate Audio with Secure RTP" [RFC6562], which needs to be read by anyone writing an RTP payload format for an audio or speech codec with these properties.

6.2. Video

The definition of RTP payload formats for video has seen an evolution from the early ones such as H.261 [RFC4587] towards the latest for VP8 [RFC7741] and H.265/HEVC [RFC7798].

The H.264 RTP payload format [RFC3984] can be seen as a smorgasbord of functionality: some of it, such as the interleaving, being pretty advanced. The reason for this was to ensure that the majority of applications considered by the ITU-T and MPEG that can be supported by RTP are indeed supported. This has created a payload format that rarely is fully implemented. Despite that, no major issues with interoperability has been reported with one exception namely the Offer/Answer and parameter signaling, which resulted in a revised specification [RFC6184]. However, complaints about its complexity are common.

The RTP payload format for uncompressed video [RFC4175] must be mentioned in this context as it contains a special feature not commonly seen in RTP payload formats. Due to the high bitrate and thus packet rate of uncompressed video (gigabits rather than megabits per second) the payload format includes a field to extend the RTP sequence number since the normal 16-bit one can wrap in less than a second. [RFC4175] also specifies a registry of different color sub-samplings that can be reused in other video RTP payload formats.

Both the H.264 and the uncompressed video format enable the implementer to fulfill the goals of application-level framing, i.e., each individual RTP Packet's payload can be independently decoded and its content used to create a video frame (or part of) and that irrespective of whether preceding packets has been lost (see Section 4) [RFC2736]. For uncompressed, this is straightforward as each pixel is independently represented from others and its location in the video frame known. H.264 is more dependent on the actual implementation, configuration of the video encoder and usage of the RTP payload format.

The common challenge with video is that, in most cases, a single compressed video frame doesn't fit into a single IP packet. Thus, the compressed representation of a video frame needs to be split over multiple packets. This can be done unintelligently with a basic payload level fragmentation method or more integrated by interfacing with the encoder's possibilities to create ADUs that are independent

and fit the MTU for the RTP packet. The latter is more robust and commonly recommended unless strong packet loss mechanisms are used and sufficient delay budget for the repair exist. Commonly, both payload-level fragmentation as well as explaining how tailored ADUs can be created are needed in a video payload format. Also, the handling of crucial metadata, like H.264 Parameter Sets, needs to be considered as decoding is not possible without receiving the used parameter sets.

6.3. Text

Only a single format text format has been standardized in the IETF, namely T.140 [RFC4103]. The 3GPP Timed Text format [RFC4396] should be considered to be text, even though in the end was registered as a video format. It was registered in that part of the tree because it deals with decorated text, usable for subtitles and other embellishments of video. However, it has many of the properties that text formats generally have.

The RTP payload format for T.140 was designed with high reliability in mind as real-time text commonly is an extremely low bitrate application. Thus, it recommends the use of RFC 2198 with many generations of redundancy. However, the format failed to provide a text-block-specific sequence number and instead relies on the RTP one to detect loss. This makes detection of missing text blocks unnecessarily difficult and hinders deployment with other robustness mechanisms that would involve switching the payload type, as that may result in erroneous error marking in the T.140 text stream.

6.4. Application

At the time of writing, the application content type contains two media types that aren't RTP transport robustness tools such as FEC [RFC3009] [RFC5109] [RFC6015] [RFC6682] and RTP retransmission [RFC4588].

The first one is H.224 [RFC4573], which enables far-end camera control over RTP. This is not an IETF-defined RTP format, only an IETF-performed registration.

The second one is "RTP Payload Format for Society of Motion Picture and Television Engineers (SMPTE) ST 336 Encoded Data" [RFC6597], which carries generic key length value (KLV) triplets. These pairs may contain arbitrary binary metadata associated with video transmissions. It has a very basic fragmentation mechanism requiring reception without packet loss, not only of the triplet itself but also one packet before and after the sequence of fragmented KLV triplet, to ensure correct reception. Specific KLV triplets

themselves may have recommendations on how to handle incomplete ones allowing the use and repair of them. In general, the application using such a mechanism must be robust to errors and also use some combination of application-level repetition, RTP-level transport robustness tools, and network-level requirements to achieve low levels of packet loss rates and repair of KLV triplets.

An author should consider applying for a media subtype under the application media type (application/<foo>) when the payload format is of a generic nature or does not clearly match any of the media types described above (audio, video, or text). However, existing limitations in, for example, SDP, have resulted in generic mechanisms normally registered in all media types possibly having been associated with any existing media types in an RTP session.

7. Important Specification Sections

A number of sections in the payload format draft need special consideration. These include the Security Considerations and IANA Considerations sections that are required in all drafts. Payload formats are also strongly recommended to have the media format description and congestion control considerations. The included RTP payload format template (Appendix A) contains sample text for some of these sections.

7.1. Media Format Description

The intention of this section is to enable reviewers and other readers to get an overview of the capabilities and major properties of the media format. It should be kept short and concise and is not a complete replacement for reading the media format specification.

The actual specification of the RTP payload format generally uses normative references to the codec format specification to define how codec data elements are included in the payload format. This normative reference can be to anything that have sufficient stability for a normative reference. There exist no formal requirement on the codec format specification being publicly available or free to access. However, it significantly helps in the review process if that specification is made available to any reviewer. There exist RTP payload format RFCs for open-source project specifications as well as an individual company's proprietary format, and a large variety of standards development organizations or industrial forums.

7.2. Security Considerations

All Internet-Drafts require a Security Considerations section. The Security Considerations section in an RTP payload format needs to concentrate on the security properties this particular format has. Some payload formats have very few specific issues or properties and can fully fall back on the security considerations for RTP in general and those of the profile being used. Because those documents are always applicable, a reference to these is normally placed first in the Security Considerations section. There is suggested text in the template below.

The security issues of confidentiality, integrity protection, replay protection and source authentication are common issue for all payload formats. These should be solved by mechanisms external to the payload and do not need any special consideration in the payload format except for a reminder on these issues. There exist exceptions, such as payload formats that includes security functionality, like ISMAcrypt [ISMACrypt2]. Reasons for this division is further documented in "Securing the RTP Protocol Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202]. For a survey of available mechanisms to meet these goals, review "Options for Securing RTP Sessions" [RFC7201]. This also includes key-exchange mechanisms for the security mechanisms, which can be both integrated or separate. The choice of key-management can have significant impact on the security properties of the RTP-based application. Suitable stock text to inform people about this is included in the template.

Potential security issues with an RTP payload format and the media encoding that need to be considered if they are applicable:

1. The decoding of the payload format or its media results in substantial non-uniformity, either in output or in complexity to perform the decoding operation. For example, a generic non-destructive compression algorithm may provide an output of almost an infinite size for a very limited input, thus consuming memory or storage space out of proportion with what the receiving application expected. Such inputs can cause some sort of disruption, i.e., a denial-of-service attack on the receiver side by preventing that host from performing usable work. Certain decoding operations may also vary in the amount of processing needed to perform those operations depending on the input. This may also be a security risk if it is possible to raise processing load significantly above nominal simply by designing a malicious input sequence. If such potential attacks exist, this must be

made clear in the Security Considerations section to make implementers aware of the need to take precautions against such behavior.

2. The inclusion of active content in the media format or its transport. "Active content" means scripts, etc., that allow an attacker to perform potentially arbitrary operations on the receiver. Most active contents has limited possibility to access the system or perform operations outside a protected sandbox. RFC 4855 [RFC4855] has a requirement that it be noted in the media types registration whether or not the payload format contains active content. If the payload format has active content, it is strongly recommended that references to any security model applicable for such content are provided. A boilerplate text for "no active content" is included in the template. This must be changed if the format actually carries active content.

3. Some media formats allow for the carrying of "user data", or types of data which are not known at the time of the specification of the payload format. Such data may be a security risk and should be mentioned.

4. Audio or Speech codecs supporting variable bitrate based on 'audio/speech' input or having discontinuous transmission support must consider the issues discussed in "Guidelines for the Use of Variable Bit Rate Audio with Secure RTP" [RFC6562].

Suitable stock text for the Security Considerations section is provided in the template in Appendix A. However, authors do need to actively consider any security issues from the start. Failure to address these issues may block approval and publication.

7.3. Congestion Control

RTP and its profiles do discuss congestion control. There is ongoing work in the IETF with both a basic circuit-breaker mechanism [RFC8083] using basic RTCP messages intended to prevent persistent congestion and also work on more capable congestion avoidance / bitrate adaptation mechanism in the RMCAT WG.

Congestion control is an important issue in any usage in networks that are not dedicated. For that reason, it is recommended that all RTP payload format documents discuss the possibilities that exist to regulate the bitrate of the transmissions using the described RTP payload format. Some formats may have limited or step-wise regulation of bitrate. Such limiting factors should be discussed.

7.4. IANA Considerations

Since all RTP payload formats contain a media type specification, they also need an IANA Considerations section. The media type name must be registered, and this is done by requesting that IANA register that media name. When that registration request is written, it shall also be requested that the media type is included under the "RTP Payload Format media types" subregistry of the RTP registry (http://www.iana.org/assignments/rtp-parameters).

Parameters for the payload format need to be included in this registration and can be specified as required or optional ones. The format of these parameters should be such that they can be included in the SDP attribute "a=fmtp" string (see Section 6 [RFC4566]), which is the common mapping. Some parameters, such as "Channel" are normally mapped to the rtpmap attribute instead; see Section 3 of [RFC4855].

In addition to the above request for media type registration, some payload formats may have parameters where, in the future, new parameter values need to be added. In these cases, a registry for that parameter must be created. This is done by defining the registry in the IANA Considerations section. BCP 26 [BCP26] provides guidelines to specifying such registries. Care should be taken when defining the policy for new registrations.

Before specifying a new registry, it is worth checking the existing ones in the IANA "MIME Media Type Sub-Parameter Registries". For example, video formats that need a media parameter expressing color sub-sampling may be able to reuse those defined for 'video/raw' [RFC4175].

8. Authoring Tools

This section provides information about some tools that may be used. Don't feel pressured to follow these recommendations. There exist a number of alternatives, including the ones listed at <http://tools.ietf.org>. But these suggestions are worth checking out before deciding that the grass is greener somewhere else.

Note that these options are related to the old text only RFC format, and do not cover tools for at the time of publication recently approved new RFC format, see [RFC7990].

8.1. Editing Tools

There are many choices when it comes to tools to choose for authoring Internet-Drafts. However, in the end, they need to be able to produce a draft that conforms to the Internet-Draft requirements. If you don't have any previous experience with authoring Internet-Drafts, xml2rfc does have some advantages. It helps by creating a lot of the necessary boilerplate in accordance with the latest rules, thus reducing the effort. It also speeds up publication after approval as the RFC Editor can use the source XML document to produce the RFC more quickly.

Another common choice is to use Microsoft Word and a suitable template (see [RFC5385]) to produce the draft and print that to file using the generic text printer. It has some advantages when it comes to spell checking and change bars. However, Word may also produce some problems, like changing formatting, and inconsistent results between what one sees in the editor and in the generated text document, at least according to the author's personal experience.

8.2. Verification Tools

There are a few tools that are very good to know about when writing a draft. These help check and verify parts of one's work. These tools can be found at <http://tools.ietf.org>.

- I-D Nits checker (https://tools.ietf.org/tools/idnits/). It checks that the boilerplate and some other things that are easily verifiable by machine are okay in your draft. Always use it before submitting a draft to avoid direct refusal in the submission step.

- ABNF Parser and verification (https://tools.ietf.org/tools/bap/abnf.cgi). Checks that your ABNF parses correctly and warns about loose ends, like undefined symbols. However, the actual content can only be verified by humans knowing what it intends to describe.

- RFC diff (https://tools.ietf.org/rfcdiff). A diff tool that is optimized for drafts and RFCs. For example, it does not point out that the footer and header have moved in relation to the text on every page.

9. Security Considerations

As this is an Informational RFC about writing drafts that are intended to become RFCs, there are no direct security considerations. However, the document does discuss the writing of Security Considerations sections and what should be particularly considered when specifying RTP payload formats.

10. Informative References

[BCP9] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

Kolkman, O., Bradner, S., and S. Turner, "Characterization of Proposed Standards", BCP 9, RFC 7127, January 2014.

Dusseault, L. and R. Sparks, "Guidance on Interoperation and Implementation Reports for Advancement to Draft Standard", BCP 9, RFC 5657, September 2009.

Housley, R., Crocker, D., and E. Burger, "Reducing the Standards Track to Two Maturity Levels", BCP 9, RFC 6410, October 2011.

Resnick, P., "Retirement of the "Internet Official Protocol Standards" Summary Document", BCP 9, RFC 7100, December 2013.

Dawkins, S., "Increasing the Number of Area Directors in an IETF Area", BCP 9, RFC 7475, March 2015.

<http://www.rfc-editor.org/info/bcp9>

[BCP25] Wasserman, M., "Updates to RFC 2418 Regarding the Management of IETF Mailing Lists", BCP 25, RFC 3934, October 2004.

Bradner, S., "IETF Working Group Guidelines and Procedures", BCP 25, RFC 2418, September 1998.

Resnick, P. and A. Farrel, "IETF Anti-Harassment Procedures", BCP 25, RFC 7776, March 2016.

<http://www.rfc-editor.org/info/bcp25>

[BCP26] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008, <http://www.rfc-editor.org/info/bcp26>.

[BCP78] Bradner, S., Ed. and J. Contreras, Ed., "Rights Contributors Provide to the IETF Trust", BCP 78, RFC 5378, November 2008, <http://www.rfc-editor.org/info/bcp78>.

[BCP79] Bradner, S., Ed., "Intellectual Property Rights in IETF Technology", BCP 79, RFC 3979, March 2005.

Narten, T., "Clarification of the Third Party Disclosure Procedure in RFC 3979", BCP 79, RFC 4879, April 2007.

<http://www.rfc-editor.org/info/bcp79>

[BLOAT] Nichols, K. and V. Jacobson, "Controlling Queue Delay", ACM Networks, Vol. 10, No. 5, DOI 10.1145/2208917.2209336, May 2012, <http://queue.acm.org/detail.cfm?id=2209336>.

[CSP-RTP] Perkins, C., "RTP: Audio and Video for the Internet", Addison-Wesley Professional, ISBN 0-672-32249-8, June 2003.

[ID-GUIDE] Housley, R., "Guidelines to Authors of Internet-Drafts", December 2010, <http://www.ietf.org/id-info/guidelines.html>.

[ISMACrypt2] Internet Streaming Media Alliance (ISMA), "ISMA Encryption and Authentication, Version 2.0 release version", November 2007, <http://www.oipf.tv/docs/mpegif/isma_easpec2.0.pdf>.

[RED] Floyd, S. and V. Jacobson, "Random Early Detection (RED) gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking 1(4) 397--413, August 1993, <http://www.aciri.org/floyd/papers/early.pdf>.

[RFC1633] Braden, R., Clark, D., and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC 1633, DOI 10.17487/RFC1633, June 1994, <http://www.rfc-editor.org/info/rfc1633>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2198] Perkins, C., Kouvelas, I., Hodson, O., Hardman, V., Handley, M., Bolot, J., Vega-Garcia, A., and S. Fosse-Parisis, "RTP Payload for Redundant Audio Data", RFC 2198, DOI 10.17487/RFC2198, September 1997, <http://www.rfc-editor.org/info/rfc2198>.

[RFC2360] Scott, G., "Guide for Internet Standards Writers", BCP 22, RFC 2360, DOI 10.17487/RFC2360, June 1998, <http://www.rfc-editor.org/info/rfc2360>.

[RFC2418] Bradner, S., "IETF Working Group Guidelines and Procedures", BCP 25, RFC 2418, DOI 10.17487/RFC2418, September 1998, <http://www.rfc-editor.org/info/rfc2418>.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, <http://www.rfc-editor.org/info/rfc2475>.

[RFC2508] Casner, S. and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", RFC 2508, DOI 10.17487/RFC2508, February 1999, <http://www.rfc-editor.org/info/rfc2508>.

[RFC2733] Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", RFC 2733, DOI 10.17487/RFC2733, December 1999, <http://www.rfc-editor.org/info/rfc2733>.

[RFC2736] Handley, M. and C. Perkins, "Guidelines for Writers of RTP Payload Format Specifications", BCP 36, RFC 2736, DOI 10.17487/RFC2736, December 1999, <http://www.rfc-editor.org/info/rfc2736>.

[RFC2959] Baugher, M., Strahm, B., and I. Suconick, "Real-Time Transport Protocol Management Information Base", RFC 2959, DOI 10.17487/RFC2959, October 2000, <http://www.rfc-editor.org/info/rfc2959>.

[RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, DOI 10.17487/RFC2974, October 2000, <http://www.rfc-editor.org/info/rfc2974>.

[RFC3009] Rosenberg, J. and H. Schulzrinne, "Registration of parityfec MIME types", RFC 3009, DOI 10.17487/RFC3009, November 2000, <http://www.rfc-editor.org/info/rfc3009>.

[RFC3095] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, DOI 10.17487/RFC3095, July 2001, <http://www.rfc-editor.org/info/rfc3095>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <http://www.rfc-editor.org/info/rfc3264>.

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, DOI 10.17487/RFC3410, December 2002, <http://www.rfc-editor.org/info/rfc3410>.

[RFC3497] Gharai, L., Perkins, C., Goncher, G., and A. Mankin, "RTP Payload Format for Society of Motion Picture and Television Engineers (SMPTE) 292M Video", RFC 3497, DOI 10.17487/RFC3497, March 2003, <http://www.rfc-editor.org/info/rfc3497>.

[RFC3545] Koren, T., Casner, S., Geevarghese, J., Thompson, B., and P. Ruddy, "Enhanced Compressed RTP (CRTP) for Links with High Delay, Packet Loss and Reordering", RFC 3545, DOI 10.17487/RFC3545, July 2003, <http://www.rfc-editor.org/info/rfc3545>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <http://www.rfc-editor.org/info/rfc3551>.

[RFC3558] Li, A., "RTP Payload Format for Enhanced Variable Rate Codecs (EVRC) and Selectable Mode Vocoders (SMV)", RFC 3558, DOI 10.17487/RFC3558, July 2003, <http://www.rfc-editor.org/info/rfc3558>.

[RFC3569] Bhattacharyya, S., Ed., "An Overview of Source-Specific Multicast (SSM)", RFC 3569, DOI 10.17487/RFC3569, July 2003, <http://www.rfc-editor.org/info/rfc3569>.

[RFC3577] Waldbusser, S., Cole, R., Kalbfleisch, C., and D. Romascanu, "Introduction to the Remote Monitoring (RMON) Family of MIB Modules", RFC 3577, DOI 10.17487/RFC3577, August 2003, <http://www.rfc-editor.org/info/rfc3577>.

[RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, DOI 10.17487/RFC3611, November 2003, <http://www.rfc-editor.org/info/rfc3611>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <http://www.rfc-editor.org/info/rfc3711>.

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <http://www.rfc-editor.org/info/rfc3828>.

[RFC3984] Wenger, S., Hannuksela, M., Stockhammer, T., Westerlund, M., and D. Singer, "RTP Payload Format for H.264 Video", RFC 3984, DOI 10.17487/RFC3984, February 2005, <http://www.rfc-editor.org/info/rfc3984>.

[RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <http://www.rfc-editor.org/info/rfc4103>.

[RFC4170] Thompson, B., Koren, T., and D. Wing, "Tunneling Multiplexed Compressed RTP (TCRTP)", BCP 110, RFC 4170, DOI 10.17487/RFC4170, November 2005, <http://www.rfc-editor.org/info/rfc4170>.

[RFC4175] Gharai, L. and C. Perkins, "RTP Payload Format for Uncompressed Video", RFC 4175, DOI 10.17487/RFC4175, September 2005, <http://www.rfc-editor.org/info/rfc4175>.

[RFC4352] Sjoberg, J., Westerlund, M., Lakaniemi, A., and S. Wenger, "RTP Payload Format for the Extended Adaptive Multi-Rate Wideband (AMR-WB+) Audio Codec", RFC 4352, DOI 10.17487/RFC4352, January 2006, <http://www.rfc-editor.org/info/rfc4352>.

[RFC4396] Rey, J. and Y. Matsui, "RTP Payload Format for 3rd Generation Partnership Project (3GPP) Timed Text", RFC 4396, DOI 10.17487/RFC4396, February 2006, <http://www.rfc-editor.org/info/rfc4396>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <http://www.rfc-editor.org/info/rfc4566>.

[RFC4571] Lazzaro, J., "Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport", RFC 4571, DOI 10.17487/RFC4571, July 2006, <http://www.rfc-editor.org/info/rfc4571>.

[RFC4573] Even, R. and A. Lochbaum, "MIME Type Registration for RTP Payload Format for H.224", RFC 4573, DOI 10.17487/RFC4573, July 2006, <http://www.rfc-editor.org/info/rfc4573>.

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, DOI 10.17487/RFC4585, July 2006, <http://www.rfc-editor.org/info/rfc4585>.

[RFC4587] Even, R., "RTP Payload Format for H.261 Video Streams", RFC 4587, DOI 10.17487/RFC4587, August 2006, <http://www.rfc-editor.org/info/rfc4587>.

[RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, "RTP Retransmission Payload Format", RFC 4588, DOI 10.17487/RFC4588, July 2006, <http://www.rfc-editor.org/info/rfc4588>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <http://www.rfc-editor.org/info/rfc4648>.

[RFC4844] Daigle, L., Ed. and Internet Architecture Board, "The RFC Series and RFC Editor", RFC 4844, DOI 10.17487/RFC4844, July 2007, <http://www.rfc-editor.org/info/rfc4844>.

[RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, DOI 10.17487/RFC4855, February 2007, <http://www.rfc-editor.org/info/rfc4855>.

[RFC4867] Sjoberg, J., Westerlund, M., Lakaniemi, A., and Q. Xie, "RTP Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs", RFC 4867, DOI 10.17487/RFC4867, April 2007, <http://www.rfc-editor.org/info/rfc4867>.

[RFC4975] Campbell, B., Ed., Mahy, R., Ed., and C. Jennings, Ed., "The Message Session Relay Protocol (MSRP)", RFC 4975, DOI 10.17487/RFC4975, September 2007, <http://www.rfc-editor.org/info/rfc4975>.

[RFC5109] Li, A., Ed., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, DOI 10.17487/RFC5109, December 2007, <http://www.rfc-editor.org/info/rfc5109>.

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, DOI 10.17487/RFC5124, February 2008, <http://www.rfc-editor.org/info/rfc5124>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <http://www.rfc-editor.org/info/rfc5285>.

[RFC5385] Touch, J., "Version 2.0 Microsoft Word Template for Creating Internet Drafts and RFCs", RFC 5385, DOI 10.17487/RFC5385, February 2010, <http://www.rfc-editor.org/info/rfc5385>.

[RFC5484] Singer, D., "Associating Time-Codes with RTP Streams", RFC 5484, DOI 10.17487/RFC5484, March 2009, <http://www.rfc-editor.org/info/rfc5484>.

[RFC5583] Schierl, T. and S. Wenger, "Signaling Media Decoding Dependency in the Session Description Protocol (SDP)", RFC 5583, DOI 10.17487/RFC5583, July 2009, <http://www.rfc-editor.org/info/rfc5583>.

[RFC5795] Sandlund, K., Pelletier, G., and L-E. Jonsson, "The RObust Header Compression (ROHC) Framework", RFC 5795, DOI 10.17487/RFC5795, March 2010, <http://www.rfc-editor.org/info/rfc5795>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <http://www.rfc-editor.org/info/rfc5905>.

[RFC6015] Begen, A., "RTP Payload Format for 1-D Interleaved Parity Forward Error Correction (FEC)", RFC 6015, DOI 10.17487/RFC6015, October 2010, <http://www.rfc-editor.org/info/rfc6015>.

[RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, DOI 10.17487/RFC6051, November 2010, <http://www.rfc-editor.org/info/rfc6051>.

[RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, DOI 10.17487/RFC6184, May 2011, <http://www.rfc-editor.org/info/rfc6184>.

[RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, DOI 10.17487/RFC6190, May 2011, <http://www.rfc-editor.org/info/rfc6190>.

[RFC6295] Lazzaro, J. and J. Wawrzynek, "RTP Payload Format for MIDI", RFC 6295, DOI 10.17487/RFC6295, June 2011, <http://www.rfc-editor.org/info/rfc6295>.

[RFC6354] Xie, Q., "Forward-Shifted RTP Redundancy Payload Support", RFC 6354, DOI 10.17487/RFC6354, August 2011, <http://www.rfc-editor.org/info/rfc6354>.

[RFC6363] Watson, M., Begen, A., and V. Roca, "Forward Error Correction (FEC) Framework", RFC 6363, DOI 10.17487/RFC6363, October 2011, <http://www.rfc-editor.org/info/rfc6363>.

[RFC6410] Housley, R., Crocker, D., and E. Burger, "Reducing the Standards Track to Two Maturity Levels", BCP 9, RFC 6410, DOI 10.17487/RFC6410, October 2011, <http://www.rfc-editor.org/info/rfc6410>.

[RFC6562] Perkins, C. and JM. Valin, "Guidelines for the Use of Variable Bit Rate Audio with Secure RTP", RFC 6562, DOI 10.17487/RFC6562, March 2012, <http://www.rfc-editor.org/info/rfc6562>.

[RFC6597] Downs, J., Ed. and J. Arbeiter, Ed., "RTP Payload Format for Society of Motion Picture and Television Engineers (SMPTE) ST 336 Encoded Data", RFC 6597, DOI 10.17487/RFC6597, April 2012, <http://www.rfc-editor.org/info/rfc6597>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <http://www.rfc-editor.org/info/rfc6679>.

[RFC6682] Watson, M., Stockhammer, T., and M. Luby, "RTP Payload Format for Raptor Forward Error Correction (FEC)", RFC 6682, DOI 10.17487/RFC6682, August 2012, <http://www.rfc-editor.org/info/rfc6682>.

[RFC6701] Farrel, A. and P. Resnick, "Sanctions Available for Application to Violators of IETF IPR Policy", RFC 6701, DOI 10.17487/RFC6701, August 2012, <http://www.rfc-editor.org/info/rfc6701>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <http://www.rfc-editor.org/info/rfc6838>.

[RFC7160] Petit-Huguenin, M. and G. Zorn, Ed., "Support for Multiple Clock Rates in an RTP Session", RFC 7160, DOI 10.17487/RFC7160, April 2014, <http://www.rfc-editor.org/info/rfc7160>.

[RFC7164] Gross, K. and R. Brandenburg, "RTP and Leap Seconds", RFC 7164, DOI 10.17487/RFC7164, March 2014, <http://www.rfc-editor.org/info/rfc7164>.

[RFC7201] Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", RFC 7201, DOI 10.17487/RFC7201, April 2014, <http://www.rfc-editor.org/info/rfc7201>.

[RFC7202] Perkins, C. and M. Westerlund, "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution", RFC 7202, DOI 10.17487/RFC7202, April 2014, <http://www.rfc-editor.org/info/rfc7202>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <http://www.rfc-editor.org/info/rfc7231>.

[RFC7273] Williams, A., Gross, K., van Brandenburg, R., and H. Stokking, "RTP Clock Source Signalling", RFC 7273, DOI 10.17487/RFC7273, June 2014, <http://www.rfc-editor.org/info/rfc7273>.

[RFC7322] Flanagan, H. and S. Ginoza, "RFC Style Guide", RFC 7322, DOI 10.17487/RFC7322, September 2014, <http://www.rfc-editor.org/info/rfc7322>.

[RFC7587] Spittka, J., Vos, K., and JM. Valin, "RTP Payload Format for the Opus Speech and Audio Codec", RFC 7587, DOI 10.17487/RFC7587, June 2015, <http://www.rfc-editor.org/info/rfc7587>.

[RFC7656] Lennox, J., Gross, K., Nandakumar, S., Salgueiro, G., and B. Burman, Ed., "A Taxonomy of Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", RFC 7656, DOI 10.17487/RFC7656, November 2015, <http://www.rfc-editor.org/info/rfc7656>.

[RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <http://www.rfc-editor.org/info/rfc7667>.

[RFC7741] Westin, P., Lundin, H., Glover, M., Uberti, J., and F. Galligan, "RTP Payload Format for VP8 Video", RFC 7741, DOI 10.17487/RFC7741, March 2016, <http://www.rfc-editor.org/info/rfc7741>.

[RFC7798] Wang, Y., Sanchez, Y., Schierl, T., Wenger, S., and M. Hannuksela, "RTP Payload Format for High Efficiency Video Coding (HEVC)", RFC 7798, DOI 10.17487/RFC7798, March 2016, <http://www.rfc-editor.org/info/rfc7798>.

[RFC7826] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, Ed., "Real-Time Streaming Protocol Version 2.0", RFC 7826, DOI 10.17487/RFC7826, December 2016, <http://www.rfc-editor.org/info/rfc7826>.

[RFC7990] Flanagan, H., "RFC Format Framework", RFC 7990, DOI 10.17487/RFC7990, December 2016, <http://www.rfc-editor.org/info/rfc7990>.

[RFC8083] Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", RFC 8083, DOI 10.17487/RFC8083, March 2017, <http://www.rfc-editor.org/info/rfc8083>.

[TAO] Hoffman, P., Ed., "The Tao of IETF: A Novice's Guide to the Internet Engineering Task Force", November 2012, <http://www.ietf.org/tao.html>.

[TRACKER] "IETF Datatracker", <https://datatracker.ietf.org/>.

Appendix A. RTP Payload Format Template

This section contains a template for writing an RTP payload format in the form of an Internet-Draft. Text within [...] are instructions and must be removed from the draft itself. Some text proposals that are included are conditional. "..." is used to indicate where further text should be written.

A.1. Title

[The title shall be descriptive but as compact as possible. RTP is allowed and recommended abbreviation in the title]

RTP payload format for ...

A.2. Front-Page Boilerplate

Status of this Memo

[Insert the IPR notice and copyright boilerplate from BCP 78 and 79 that applies to this draft.]

[Insert the current Internet-Draft document explanation. At the time of publishing it was:]

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."

A.3. Abstract

[A payload format abstract should mention the capabilities of the format, for which media format is used, and a little about that codec formats capabilities. Any abbreviation used in the payload format must be spelled out here except the very well known like RTP. No citations are allowed, and no use of language from RFC 2119 either.]

A.4. Table of Contents

[If your draft is approved for publication as an RFC, a Table of Contents is required, per [RFC7322].]

A.5. Introduction

[The Introduction should provide a background and overview of the payload format's capabilities. No normative language in this section, i.e., no MUST, SHOULDs etc.]

A.6. Conventions, Definitions, and Abbreviations

[Define conventions, definitions, and abbreviations used in the document in this section. The most common definition used in RTP payload formats are the RFC 2119 definitions of the uppercase normative words, e.g., MUST and SHOULD.]

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.

A.7. Media Format Description

[The intention of this section is to enable reviewers and persons to get an overview of the capabilities and major properties of the media format. It should be kept short and concise and is not a complete replacement for reading the media format specification.]

A.8. Payload Format

[Overview of payload structure]

A.8.1. RTP Header Usage

[RTP header usage needs to be defined. The fields that absolutely need to be defined are timestamp and marker bit. Further fields may be specified if used. All the rest should be left to their RTP specification definition.]

The remaining RTP header fields are used as specified in RTP [RFC3550].

A.8.2. Payload Header

[Define how the payload header, if it exists, is structured and used.]

A.8.3. Payload Data

[The payload data, i.e., what the media codec has produced. Commonly done through reference to the media codec specification, which defines how the data is structured. Rules for padding may need to be defined to bring data to octet alignment.]

A.9. Payload Examples

[One or more examples are good to help ease the understanding of the RTP payload format.]

A.10. Congestion Control Considerations

[This section is to describe the possibility to vary the bitrate as a response to congestion. Below is also a proposal for an initial text that reference RTP and profiles definition of congestion control.]

Congestion control for RTP SHALL be used in accordance with RFC 3550 [RFC3550], and with any applicable RTP profile: e.g., RFC 3551 [RFC3551]. An additional requirement if best-effort service is being used is users of this payload format MUST monitor packet loss to ensure that the packet loss rate is within acceptable parameters. Circuit Breakers [RFC8083] is an update to RTP [RFC3550] that defines criteria for when one is required to stop sending RTP Packet Streams. The circuit breakers is to be implemented and followed.

A.11. Payload Format Parameters

This RTP payload format is identified using the ... media type, which is registered in accordance with RFC 4855 [RFC4855] and using the template of RFC 6838 [RFC6838].

A.11.1. Media Type Definition

[Here the media type registration template from RFC 6838 is placed and filled out. This template is provided with some common RTP boilerplate.]

Type name:

Subtype name:

Required parameters:

Optional parameters:

Encoding considerations:

This media type is framed and binary; see Section 4.8 in RFC 6838 [RFC6838].

Security considerations:

Please see the Security Considerations section in RFC XXXX

Interoperability considerations:

Published specification:

Applications that use this media type:

Additional information:

Deprecated alias names for this type:

[Only applicable if there exists widely deployed alias for this media type; see Section 4.2.9 of [RFC6838]. Remove or use N/A otherwise.]

Magic number(s):

[Only applicable for media types that has file format specification. Remove or use N/A otherwise.]

File extension(s):

[Only applicable for media types that has file format specification. Remove or use N/A otherwise.]

Macintosh file type code(s):

[Only applicable for media types that has file format specification. Even for file formats they can be skipped as they are not relied on after Mac OS 9.X. Remove or use N/A otherwise.]

Person & email address to contact for further information:

Intended usage:

[One of COMMON, LIMITED USE, or OBSOLETE.]

Restrictions on usage:

[The below text is for media types that is only defined for RTP payload formats. There exist certain media types that are defined both as RTP payload formats and file transfer. The rules for such types are documented in RFC 4855 [RFC4855].]

This media type depends on RTP framing and, hence, is only defined for transfer via RTP [RFC3550]. Transport within other framing protocols is not defined at this time.

Author:

Change controller:

IETF Payload working group delegated from the IESG.

Provisional registration? (standards tree only):

No

(Any other information that the author deems interesting may be added below this line.)

[From RFC 6838:

"N/A", written exactly that way, can be used in any field if desired to emphasize the fact that it does not apply or that the question was not omitted by accident. Do not use 'none' or other words that could be mistaken for a response.

Limited-use media types should also note in the applications list whether or not that list is exhaustive.]

A.11.2. Mapping to SDP

The mapping of the above defined payload format media type and its parameters SHALL be done according to Section 3 of RFC 4855 [RFC4855].

[More specific rules only need to be included if some parameter does not match these rules.]

A.11.2.1. Offer/Answer Considerations

[Here write your Offer/Answer considerations section; please see Section 3.4.2.1 for help.]

A.11.2.2. Declarative SDP Considerations

[Here write your considerations for declarative SDP, please see Section 3.4.2.2 for help.]

A.12. IANA Considerations

This memo requests that IANA registers [insert media type name here] as specified in Appendix A.11.1. The media type is also requested to be added to the IANA registry for "RTP Payload Format MIME types" <http://www.iana.org/assignments/rtp-parameters>.

[See Section 7.4 and consider if any of the parameter needs a registered name space.]

A.13. Security Considerations

[See Section 7.2.]

RTP packets using the payload format defined in this specification are subject to the security considerations discussed in the RTP specification [RFC3550] , and in any applicable RTP profile such as RTP/AVP [RFC3551], RTP/AVPF [RFC4585], RTP/SAVP [RFC3711], or RTP/SAVPF [RFC5124]. However, as "Securing the RTP Protocol Framework: Why RTP Does Not Mandate a Single Media Security Solution" [RFC7202] discusses, it is not an RTP payload format's responsibility to discuss or mandate what solutions are used to meet the basic security goals like confidentiality, integrity, and source authenticity for RTP in general. This responsibility lays on anyone using RTP in an application. They can find guidance on available security mechanisms and important considerations in "Options for Securing RTP Sessions" [RFC7201]. Applications SHOULD use one or more appropriate strong security mechanisms. The rest of this Security Considerations section discusses the security impacting properties of the payload format itself.

This RTP payload format and its media decoder do not exhibit any significant non-uniformity in the receiver-side computational complexity for packet processing, and thus are unlikely to pose a denial-of-service threat due to the receipt of pathological data. Nor does the RTP payload format contain any active content.

[The previous paragraph may need editing due to the format breaking either of the statements. Fill in here any further potential security threats created by the payload format itself.]

A.14. RFC Editor Considerations

Note to RFC Editor: This section may be removed after carrying out all the instructions of this section.

RFC XXXX is to be replaced by the RFC number this specification receives when published.

A.15. References

[References must be classified as either normative or informative and added to the relevant section. References should use descriptive reference tags.]

A.15.1. Normative References

[Normative references are those that are required to be used to correctly implement the payload format. Also, when requirements language is used, as in the sample text for "Congestion Control Considerations" above, there should be a normative reference to [RFC2119].]

A.15.2. Informative References

[All other references.]

A.16. Authors' Addresses

[All authors need to include their name and email address as a minimum: postal mail and possibly phone numbers are included commonly.]

[The Template Ends Here!]

Acknowledgements

The author would like to thank the individuals who have provided input to this document. These individuals include Richard Barnes, Ali C. Begen, Bo Burman, Ross Finlayson, Russ Housley, John Lazzaro, Jonathan Lennox, Colin Perkins, Tom Taylor, Stephan Wenger, and Qin Wu.

Contributors

The author would like to thank Tom Taylor for the editing pass of the whole document and contributing text regarding proprietary RTP payload formats. Thanks also goes to Thomas Schierl who contributed text regarding Media Scalability features in payload formats (Section 5.1.5). Stephan Wenger has contributed text on the need to understand the media coding (Section 3.1) as well as joint development of payload format with the media coding (Section 4.4).

Author's Address

Magnus Westerlund
Ericsson
Farogatan 2
SE-164 80 Kista
Sweden

Phone: +46 10 714 82 87
Email: magnus.westerlund@ericsson.com