Payload Working Group
Internet-Draft
Intended status: Informational
Expires: April 24, 2014

M. Westerlund
Ericsson

October 21, 2013

How to Write an RTP Payload Format
draft-ietf-payload-rtp-howto-09

Abstract

This document contains information on how to best 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 Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 24, 2014.

Copyright Notice

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 non-trivial 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 have been defined, and robustness tools have also been defined, and these need to be considered.

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

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 do 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 and IANA considerations sections. This document ends with an appendix containing a template that can be used when writing RTP payload formats specifications.

2. Terminology

2.1. Definitions

Media 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 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.

2.2. Acronyms

ABNF: Augmented Backus-Naur Form [RFC5234]

ADU: Application Data Unit

ALF: Application Level Framing

ASM: Any-Source Multicast

BCP: Best Current Practice

ID: 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 an both in the informational category and being an instruction rather than a specification, this document does not use any RFC 2119 language and the interpretation of "may", "should", "recommended" and "must" are the ones of the English language.

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 one can 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 Spec

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 sub-sections 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" [RFC6722] that goes through most things that one needs to know about the IETF. This contains information about history, organizational structure, how the WG and meetings work and many more details.

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 [RFC5378] (related to copyright, including software copyright for example code) and BCP 79 [RFC3979] (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 the mailing list or speaking at a meeting) they 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 applies on what is specified in the RTP Payload format Internet Draft (and later RFC), IPRs 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 bodies own IPR rules.

The main part of the IETF process is formally defined in RFC 2026 [RFC2026]. RFC 2418 [RFC2418] 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 The RFC Series and RFC Editor [RFC4844]. The most important stream for RTP payload formats authors are the IETF Stream. In this streams the work of IETF is published. The stream contains documents of several different categories: standards track, informational, experimental, best current practice (BCP), and historic. The standard tracks previously allowed for documents of three different maturity classifications, proposed, draft and Internet Standard. Since October 2011 this has been reduced to only two levels: Proposed Standard and Internet Standard [RFC6410]. A standards track document must start as proposed; after successful deployment and operational experience with at least two implementations it can be moved to 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 which 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 engine, 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 RFC-errata. Any approved Errata is issues of significant importance with the RFC and thus should be known also 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 ID checklist (http://www.ietf.org/ID-Checklist.html) and the RFC editorial guidelines and procedures [RFC-ED]. Another document that can be useful is the "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 type 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 that 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 layer 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 RFC 3550 [RFC3550] and the existing profiles. As a complement to the standards documents there exists a book totally dedicated to RTP [CSP-RTP]. There exist several profiles for RTP today, but all are based on the "RTP Profile for Audio and Video Conferences with Minimal Control" (RFC 3551) [RFC3551] (abbreviated as RTP/AVP). The other profiles that one should know about are Secure RTP (RTP/SAVP) [RFC3711], "Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)" [RFC4585] and "Extended Secure RTP Profile for 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: The "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 and therefore 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 which allows the redundant data to be transmitted prior to the original.

RFC 5109: The "RTP Payload Format for Generic Forward Error Correction (FEC)" [RFC5109] provides an XOR-based 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 low-to-medium delay tolerant applications with limited amount of RTP packets.

RFC 6015: The "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 produce 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: The 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 the RTP Payload for Raptor FEC [RFC6682].

RTP Retransmission: The RTP retransmission scheme [RFC4588] is used for semi-reliability of the most important RTP packets in a media 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 already late data. TCP is also limited to point-to-point connections which further restricts its applicability.

There has also been both discussion and design of RTP payload formats, e.g., AMR and 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 if one can design something usable or not. Payload format designers are recommended against considering 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 IETF is chartered (at the time of writing) with defining new report formats. The list of specified formats are available in IANA's RTCP XR Block Type registry (http://www.iana.org/assignments/rtcp-xr-block-types/rtcp-xr-block-types.xhtml). The report formats that are defined in RFC3611 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 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 slow 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.

RFC 3095 & RFC 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. Generic mechanisms by definition 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 [I-D.ietf-avtcore-rtp-security-options] and Why RTP Does Not Mandate a Single Security Mechanism [I-D.ietf-avt-srtp-not-mandatory]. 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 which 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 time of writing this documet. 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 also other solutions than SRTP consult "Options for Securing RTP Sessions" [I-D.ietf-avtcore-rtp-security-options].

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 behaviour (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 (RFC3550), 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. The payload type is used to indicate the used payload format. The Sender Source Identifier (SSRC) is used to distinguish RTP packets from multiple senders and media streams. Finally, [RFC5285] specifies how to transport payload format independent metadata relating to the RTP packet.

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 timestamp rate is an active design choice and is further discussed in Section 5.2.

Discontinuous transmissions (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 signalling (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 Synchronisation Source Identifier (SSRC) 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.

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 a 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 media 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 are also 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.

The next multiplexing point is separation of different sources within an RTP session. Here RTP uses the SSRC to identify individual sources. An example of individual sources in an audio RTP session would be different microphones, independently of whether they are connected to the same host or different hosts. 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].

For more discussion and consideration of how and when to use the different RTP multiplexing points see [I-D.ietf-avtcore-multiplex-guidelines].

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 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 media 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 also different media from the same end-point. This puts two requirements on RTP: the possibility to determine which media are from the same end-point 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 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 media 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 signalled using RTP Clock Source Signalling [I-D.ietf-avtcore-clksrc].

Figure 1 illustrates how if one receives RTCP Sender Report (SR) packet P1 in one media stream and RTCP SR packet P2 in the other session, 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 medium currently used in the sessions.

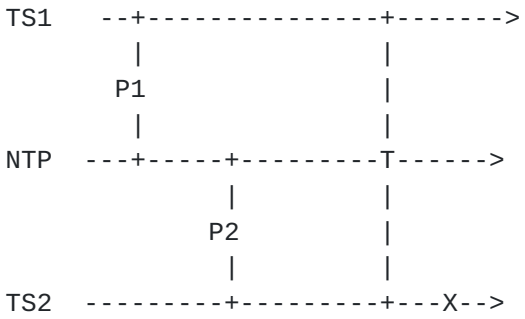

Figure 1: RTCP Synchronization

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 earth's rotation. Adding or removing seconds can impact first of all the reference clock as discussed in "RTP and Leap Seconds" [I-D.ietf-avtcore-leap-second]. But 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 [I-D.ietf-avtcore-leap-second].

3.4. Signalling Aspects

RTP payload formats are used in the context of application signalling protocols such as SIP [RFC3261] using the Session Description Protocol (SDP) [RFC4566] with Offer/Answer [RFC3264], RTSP [RFC2326] or SAP [RFC2326]. These examples all use out-of-band signalling to

indicate which type of RTP media streams that 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 have also 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 IETF (SDP-NG). In the event, the proposal did not get beyond the initial 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 signalling, 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 [RFC2616], 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 two-part 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 IANA Web site (http://www.iana.org).

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 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 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 semi-colon, are recommended. If that is not possible some type of escaping should be used. Usage of quote (") is recommended, and 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 signalling 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 RFC 4855 [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 and instead 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 media 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 media 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 which 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 shoe-horned 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 media 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 an new offer to update the parameter values for its outgoing media stream.

Another issue to consider is the send-only media 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 which 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]; "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] is used for announcing multicast sessions. Independently of the usage of Source-specific Multicast (SSM) [RFC3569] or Any-source Multicast (ASM), the SDP provided by SAP applies to all participants. All media that is sent to the session must follow the media 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 which 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) [RFC2326] allows the negotiation of transport parameters for media streams which 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 media 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 media 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 as well as 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 to consider both limited MTUs as well as enable support of larger MTUs. This should be considered in the design, especially in regards 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 Marking (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 strive 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. Active Queue Management (AQM) is a term covering mechanism that tries to do something smarter by actively managing the queue, for example by sending congestion signals earlier by dropping packets earlier in the queue. The behavior also affects the flow interactions. For example, Random Early Drop (RED) selects which packet(s) to drop randomly. This give flows that have more packets in the queue a higher probability to experience the packet loss (congestion signal). There is ongoing work 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 paths properties. Quality of Service (QoS) mechanism are intended to provide the possibility to bound the path properties. Where Diffserv [RFC2475] markings effects 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 [RFC1663] has the potential for stricter guarantees as the properties are agreed on by each hop on the path.

4. Standardisation 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 Working Group applies when working with RTP payload formats.

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

Initial effort: Using this document as 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 Working Group 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. It is important to consider from the start 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. This can be done at any time except the 3 weeks (current deadline at the time of writing, consult current announcements) prior to an IETF meeting. When the IETF draft announcement has been sent out on the draft announcement list, forward it to the PAYLOAD WG and request that it be reviewed. In the email outline any issues the authors currently have with the design.

Iterative improvements: Taking the feedback 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 buffer

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 chairs 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.

WG Last Call: A WG Last Call of at least 2 weeks is always performed for payload formats in the PAYLOAD WG. The authors request WG last call for a draft when they think it is mature enough for publication. The chairs perform a review to check if they agree with the authors' assessment. If the chairs 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 updates of 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 chairs request publication of the draft, after it has passed WG Last Call. After this, the approval and publication process described in RFC 2026 [RFC2026] is performed. The status after the publication has been requested can be tracked using the IETF data 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 its correctness.

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 its WG members a specific draft file naming convention shall be used for RTP payload formats. Individual submissions shall be named 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 acronym. 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.

4.1.4. Writing Style

When writing an IETF draft for an RTP payload format one should observe some few considerations (that may be somewhat diverging from the style 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 choice 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 spec.

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 actually 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 keep 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 IETF the most common method is to use "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119] that 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 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. Try also 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 initial submissions, checking if a draft is ready to become a WG document or ready 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 and 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 standard bodies. The requirements on the organization are outlined in the media types registration document [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 signalling 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 tracks 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 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 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 media coding spec 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 spec to be RTP friendly. One example for such a media coding spec 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 specs.

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 a non-broken 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 requirements to support these features; instead, their applicability must be considered for each payload format. It may in fact 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 produces small ADUs mixed with large, for example H.264 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 of 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. The perhaps most important reason to avoid IP fragmentation is that IP fragmented packets commonly are discarded in the network, especially by Network Address Translators or Firewalls. The usage of RTP payload format-level fragmentation 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 Re-Scheduling

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 spread 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 re-scheduling mechanism. That has the effect that the packets transmitted consecutively originate from different points in the media 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 re-scheduling 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 signalling 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 media 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 change 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 media 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 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 Scalable Video Coding [RFC6190] (SVC).

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 media stream, i.e., RTP packets may be not transmitted or forwarded to a client in order to adapt the media stream rate as well as the media stream quality, while still providing a decodable subset of the media stream to a client. One example for using the scalability feature may be an RTP Mixer (Multipoint Control Unit) 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 media stream (base layer) may get better QoS parameters, then 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 single 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 media stream requires dropping of specific packets in order to provide the client with a compliant media 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 that belongs to the layers, that no other layer will be dependent on after the dropping. Secondly, if the MANE can work as RTP mixer or translator it can rewrite the RTP and RTCP in such a way that the receiver will not suspect non-intentional 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 sessions, 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 for dropping packets is based on the Network Address which identifies the RTP session to be dropped. In order to allow correct data provisioning to a decoder after reception from different sessions, data re-alignment mechanisms are required. In some cases, existing generic tools as described below can be employed to enable such re-alignment, and when those generic mechanisms are sufficient,

they should be used. For example, Rapid Sync 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 end-point has the benefit in MST to know which layer or set of layers the missing packets belong 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, and in these cases the RTP sequence number wraps too quickly. As 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, which is 2 minutes in ones calculations. 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 mechanism 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 which 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 if the payload format has this property or not.

A RTP payload format also needs to define what timestamp rates, or clock rates (as it is also called), that 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 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) [I-D.ietf-avtext-multiple-clock-rates] this should be avoided if one expects to need to switch between modes.

An alternatives then is to find a common denominator frequency between the different modes, e.g. OPUS [I-D.ietf-payload-rtp-opus] 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 is 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 it will cause a too low resolution 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] uses 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 specific details of their design.

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 content 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 that 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 exist a security concern with variable bit-rate audio and speech codecs that changes their payload length based on the input data. This can leak information, especially in structured communication like 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 towards the latest for VP-8 and H.265/HEVC.

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 signalling, 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 bit-rate and thus packet rate of uncompressed video (gigabits rather than megabits) 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 re-used 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 don'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 meta data, 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 bit-rate 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 relies instead of 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

The application content type contains at the time of writing 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 H224 [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 the 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 meta data associated with video transmissions. It has a very basic fragmentation mechanism requiring packet loss free reception 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 recommendation on how to handle non-complete 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.

The application top media type (application/<foo>) should be considered to be used when the payload format defined is not clearly matching any of the existing media types (audio, video or text) or are of a generic nature. However, existing limitations in for example SDP, has resulted in that generic mechanisms normally are registered in all media types to be possible to have 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 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 draft 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.

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 an 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" [I-D.ietf-avt-srtp-not-mandatory]. For a survey of available mechanisms to meet these goals, review "Options for Securing RTP Sessions" [I-D.ietf-avtcore-rtp-security-options]. 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 if the payload format contains active content or not. 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 bit-rate 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 the appendix. 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 [I-D.ietf-avtcore-rtp-circuit-breakers] using basic RTCP messages intended to prevent persistent congestion, but also work on more capable congestion avoidance / bit-rate adaptation mechanism in the RMCAT WG.

Congestion control is an important issue in any usage in non-dedicated networks. For that reason it is recommended that all RTP payload format documents discuss the possibilities that exist to regulate the bit-rate of the transmissions using the described RTP payload format. Some formats may have limited or step wise regulation of bit-rate. 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" list part of the RTP registry (http://www.iana.org/assignments/rtp-parameters).

Parameters for the payload format needs to be included in this registration and can be specified as required or optional ones. The format of these parameter 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 [RFC5226] 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" list. For example video formats needing 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. But these suggestions are worth checking out before deciding that the field is greener somewhere else.

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 create a lot of the necessary boiler plate 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 authors' 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.

- ID Nits checker. It checks that the boiler plate 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. 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. 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. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

10. Security Considerations

As this is an informational document 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.

11. 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).

12. Acknowledgements

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

13. Informative References

[CSP-RTP] Perkins, C., "RTP: Audio and Video for the Internet", June 2003.

[I-D.ietf-avt-srtp-not-mandatory]
Perkins, C. and M. Westerlund, "Securing the RTP Protocol Framework: Why RTP Does Not Mandate a Single Media Security Solution", draft-ietf-avt-srtp-not-mandatory-14 (work in progress), October 2013.

[I-D.ietf-avtcore-clksrc]
Williams, A., Gross, K., Brandenburg, R., and H. Stokking, "RTP Clock Source Signalling", draft-ietf-avtcore-clksrc-07 (work in progress), October 2013.

[I-D.ietf-avtcore-leap-second]
Gross, K. and R. Brandenburg, "RTP and Leap Seconds", draft-ietf-avtcore-leap-second-05 (work in progress), October 2013.

[I-D.ietf-avtcore-multiplex-guidelines]
Westerlund, M., Perkins, C., and H. Alvestrand, "Guidelines for using the Multiplexing Features of RTP to Support Multiple Media Streams", draft-ietf-avtcore-multiplex-guidelines-01 (work in progress), July 2013.

[I-D.ietf-avtcore-rtp-circuit-breakers]
Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", draft-ietf-avtcore-rtp-circuit-breakers-03 (work in progress), July 2013.

[I-D.ietf-avtcore-rtp-security-options]
Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", draft-ietf-avtcore-rtp-security-options-07 (work in progress), October 2013.

[I-D.ietf-avtext-multiple-clock-rates]
Petit-Huguenin, M. and G. Zorn, "Support for Multiple Clock Rates in an RTP Session", draft-ietf-avtext-multiple-clock-rates-10 (work in progress), September 2013.

[I-D.ietf-payload-rtp-opus]
Spittka, J., Vos, K., and J. Valin, "RTP Payload Format for Opus Speech and Audio Codec", draft-ietf-payload-rtp-opus-01 (work in progress), August 2013.

[ISMACrypt2]
, "ISMA Encryption and Authentication, Version 2.0 release version", November 2007.

[MACOSFILETYPES]
Apple Knowledge Base Article 55381<http://www.info.apple.com/kbnum/n55381>, "Mac OS: File Type and Creator Codes, and File Formats", 1993.

[RFC-ED] http://www.rfc-editor.org/policy.html, "RFC Editorial Guidelines and Procedures", July 2008.

[RFC1663] Rand, D., "PPP Reliable Transmission", RFC 1663, July 1994.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[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, September 1997.

[RFC2326] Schulzrinne, H., Rao, A., and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[RFC2360] Scott, G., "Guide for Internet Standards Writers", BCP 22, RFC 2360, June 1998.

[RFC2418] Bradner, S., "IETF Working Group Guidelines and Procedures", BCP 25, RFC 2418, September 1998.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.

[RFC2508] Casner, S. and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", RFC 2508, February 1999.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2733] Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", RFC 2733, December 1999.

[RFC2736] Handley, M. and C. Perkins, "Guidelines for Writers of RTP Payload Format Specifications", BCP 36, RFC 2736, December 1999.

[RFC2959] Baugher, M., Strahm, B., and I. Suconick, "Real-Time Transport Protocol Management Information Base", RFC 2959, October 2000.

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

[RFC3009] Rosenberg, J. and H. Schulzrinne, "Registration of parityfec MIME types", RFC 3009, November 2000.

[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, July 2001.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[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, March 2003.

[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, July 2003.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, July 2003.

[RFC3558] Li, A., "RTP Payload Format for Enhanced Variable Rate Codecs (EVRC) and Selectable Mode Vocoders (SMV)", RFC 3558, July 2003.

[RFC3569] Bhattacharyya, S., "An Overview of Source-Specific Multicast (SSM)", RFC 3569, July 2003.

[RFC3577] Waldbusser, S., Cole, R., Kalbfleisch, C., and D. Romascanu, "Introduction to the Remote Monitoring (RMON) Family of MIB Modules", RFC 3577, August 2003.

[RFC3611] Friedman, T., Caceres, R., and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., and G. Fairhurst, "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, July 2004.

[RFC3979] Bradner, S., "Intellectual Property Rights in IETF Technology", BCP 79, RFC 3979, March 2005.

[RFC3984] Wenger, S., Hannuksela, M., Stockhammer, T., Westerlund, M., and D. Singer, "RTP Payload Format for H.264 Video", RFC 3984, February 2005.

[RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, June 2005.

[RFC4170] Thompson, B., Koren, T., and D. Wing, "Tunneling Multiplexed Compressed RTP (TCRTP)", BCP 110, RFC 4170, November 2005.

[RFC4175] Gharai, L. and C. Perkins, "RTP Payload Format for Uncompressed Video", RFC 4175, September 2005.

[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, January 2006.

[RFC4396] Rey, J. and Y. Matsui, "RTP Payload Format for 3rd Generation Partnership Project (3GPP) Timed Text", RFC 4396, February 2006.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC4571] Lazzaro, J., "Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport", RFC 4571, July 2006.

[RFC4573] Even, R. and A. Lochbaum, "MIME Type Registration for RTP Payload Format for H.224", RFC 4573, July 2006.

[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, July 2006.

[RFC4588] Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R. Hakenberg, "RTP Retransmission Payload Format", RFC 4588, July 2006.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

[RFC4844] Daigle, L. Internet Architecture Board, "The RFC Series and RFC Editor", RFC 4844, July 2007.

[RFC4855] Casner, S., "Media Type Registration of RTP Payload Formats", RFC 4855, February 2007.

[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, April 2007.

[RFC4975] Campbell, B., Mahy, R., and C. Jennings, "The Message Session Relay Protocol (MSRP)", RFC 4975, September 2007.

[RFC5109] Li, A., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, December 2007.

[RFC5124] Ott, J. and E. Carrara, "Extended Secure RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/SAVPF)", RFC 5124, February 2008.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

[RFC5378] Bradner, S. and J. Contreras, "Rights Contributors Provide to the IETF Trust", BCP 78, RFC 5378, November 2008.

[RFC5385] Touch, J., "Version 2.0 Microsoft Word Template for Creating Internet Drafts and RFCs", RFC 5385, February 2010.

[RFC5484] Singer, D., "Associating Time-Codes with RTP Streams", RFC 5484, March 2009.

[RFC5583] Schierl, T. and S. Wenger, "Signaling Media Decoding Dependency in the Session Description Protocol (SDP)", RFC 5583, July 2009.

[RFC5795] Sandlund, K., Pelletier, G., and L-E. Jonsson, "The RObust Header Compression (ROHC) Framework", RFC 5795, March 2010.

[RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC6015] Begen, A., "RTP Payload Format for 1-D Interleaved Parity Forward Error Correction (FEC)", RFC 6015, October 2010.

[RFC6051] Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, November 2010.

[RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, May 2011.

[RFC6190] Wenger, S., Wang, Y., Schierl, T., and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, May 2011.

[RFC6295] Lazzaro, J. and J. Wawrzynek, "RTP Payload Format for MIDI", RFC 6295, June 2011.

[RFC6354] Xie, Q., "Forward-Shifted RTP Redundancy Payload Support", RFC 6354, August 2011.

[RFC6363] Watson, M., Begen, A., and V. Roca, "Forward Error Correction (FEC) Framework", RFC 6363, October 2011.

[RFC6410] Housley, R., Crocker, D., and E. Burger, "Reducing the Standards Track to Two Maturity Levels", BCP 9, RFC 6410, October 2011.

[RFC6562] Perkins, C. and JM. Valin, "Guidelines for the Use of Variable Bit Rate Audio with Secure RTP", RFC 6562, March 2012.

[RFC6597] Downs, J. and J. Arbeiter, "RTP Payload Format for Society of Motion Picture and Television Engineers (SMPTE) ST 336 Encoded Data", RFC 6597, April 2012.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, August 2012.

[RFC6682] Watson, M., Stockhammer, T., and M. Luby, "RTP Payload Format for Raptor Forward Error Correction (FEC)", RFC 6682, August 2012.

[RFC6701] Farrel, A. and P. Resnick, "Sanctions Available for Application to Violators of IETF IPR Policy", RFC 6701, August 2012.

[RFC6722] Hoffman, P., "Publishing the "Tao of the IETF" as a Web Page", RFC 6722, August 2012.

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

Appendix A. RTP Payload Format Template

This section contains a template for writing an RTP payload format in form as a Internet draft. Text within [...] are instructions and must be removed. 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 boiler plate 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 references are allowed, no use of RFC 2119 language either.]

A.4. Table of Content

[All drafts over 15 pages in length must have an Table of Content.]

A.5. Introduction

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

A.6. Conventions, Definitions and Acronyms

[Define conventions, definitions and acronyms used in the document in this section. The most common definition used in RTP Payload formats are the RFC 2119 definitions of the upper case 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 field 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 [RFC 3550].

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 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 bit-rate 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 [I-D.ietf-avtcore-rtp-circuit-breakers] 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 RFC6838 [RFC6838].

Security considerations:

Please see security consideration in RFCXXXX

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. 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:

Some discussion of Macintosh file type codes and their purpose can be found in [MACOSFILETYPES].

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 consideration 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" [I-D.ietf-avt-srtp-not-mandatory] discusses it is not an RTP payload formats 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 [I-D.ietf-avtcore-rtp-security-options]. The rest of the this security consideration 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.

RFCXXXX 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.]

A.15.2. Informative References

[All other references.]

A.16. Author Addresses

[All Authors need to include their Name and email addresses as a minimal. Commonly also surface mail and possibly phone numbers are included.]

[The Template Ends Here!]

Author's Address

Magnus Westerlund
Ericsson
Farogatan 6
SE-164 80 Kista
Sweden

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