

Network Working Group  
Request for Comments: 4588  
Category: Standards Track

J. Rey  
Panasonic  
D. Leon  
Consultant  
A. Miyazaki  
Panasonic  
V. Varsa  
Nokia  
R. Hakenberg  
Panasonic  
July 2006

## RTP Retransmission Payload Format

### Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

### Copyright Notice

Copyright (C) The Internet Society (2006).

### Abstract

RTP retransmission is an effective packet loss recovery technique for real-time applications with relaxed delay bounds. This document describes an RTP payload format for performing retransmissions. Retransmitted RTP packets are sent in a separate stream from the original RTP stream. It is assumed that feedback from receivers to senders is available. In particular, it is assumed that Real-time Transport Control Protocol (RTCP) feedback as defined in the extended RTP profile for RTCP-based feedback (denoted RTP/AVPF) is available in this memo.

## Table of Contents

1. Introduction .....	3
2. Terminology .....	3
3. Requirements and Design Rationale for a Retransmission Scheme ...	4
3.1. Multiplexing Scheme Choice .....	6
4. Retransmission Payload Format .....	7
5. Association of Retransmission and Original Streams .....	9
5.1. Retransmission Session Sharing .....	9
5.2. CNAME Use .....	9
5.3. Association at the Receiver .....	9
6. Use with the Extended RTP Profile for RTCP-based Feedback .....	11
6.1. RTCP at the Sender .....	11
6.2. RTCP Receiver Reports .....	11
6.3. Retransmission Requests .....	12
6.4. Timing Rules .....	13
7. Congestion Control .....	13
8. Retransmission Payload Format MIME Type Registration .....	15
8.1. Introduction .....	15
8.2. Registration of audio/rtx .....	16
8.3. Registration of video/rtx .....	17
8.4. Registration of text/rtx .....	18
8.5. Registration of application/rtx .....	19
8.6. Mapping to SDP .....	20
8.7. SDP Description with Session-Multiplexing .....	20
8.8. SDP Description with SSRC-Multiplexing .....	21
9. RTSP Considerations .....	22
9.1. RTSP Control with SSRC-Multiplexing .....	22
9.2. RTSP Control with Session-Multiplexing .....	22
9.3. RTSP Control of the Retransmission Stream .....	23
9.4. Cache Control .....	23
10. Implementation Examples .....	23
10.1. A Minimal Receiver Implementation Example .....	24
10.2. Retransmission of Layered Encoded Media in Multicast .....	25
11. IANA Considerations .....	26
12. Security Considerations .....	26
13. Acknowledgements .....	27
14. References .....	27
14.1. Normative References .....	27
14.2. Informative References .....	28
Appendix A. How to Control the Number of Rtxs. per Packet .....	29

## 1. Introduction

Packet losses between an RTP sender and receiver may significantly degrade the quality of the received media. Several techniques, such as forward error correction (FEC), retransmissions, or interleaving, may be considered to increase packet loss resiliency. RFC 2354 [8] discusses the different options.

When choosing a repair technique for a particular application, the tolerable latency of the application has to be taken into account. In the case of multimedia conferencing, the end-to-end delay has to be at most a few hundred milliseconds in order to guarantee interactivity, which usually excludes the use of retransmission.

With sufficient latency, the efficiency of the repair scheme can be increased. The sender may use the receiver feedback in order to react to losses before their playout time at the receiver.

In the case of multimedia streaming, the user can tolerate an initial latency as part of the session set-up and thus an end-to-end delay of several seconds may be acceptable. RTP retransmission as defined in this document is targeted at such applications.

Furthermore, the RTP retransmission method defined herein is applicable to unicast and (small) multicast groups. The present document defines a payload format for retransmitted RTP packets and provides protocol rules for the sender and the receiver involved in retransmissions.

This retransmission payload format was designed for use with the extended RTP profile for RTCP-based feedback, AVPF [1]. It may also be used with other RTP profiles defined in the future.

The AVPF profile allows for more frequent feedback and for early feedback. It defines a general-purpose feedback message, i.e., NACK, as well as codec and application-specific feedback messages. See [1] for details.

## 2. Terminology

The following terms are used in this document:

CSRC: contributing source. See [3].

Original packet: an RTP packet that carries user data sent for the first time by an RTP sender.

Original stream: the RTP stream of original packets.

Retransmission packet: an RTP packet that is to be used by the receiver instead of a lost original packet. Such a retransmission packet is said to be associated with the original RTP packet.

Retransmission request: a means by which an RTP receiver is able to request that the RTP sender should send a retransmission packet for a given original packet. Usually, an RTCP NACK packet as specified in [1] is used as retransmission request for lost packets.

Retransmission stream: the stream of retransmission packets associated with an original stream.

Session-multiplexing: scheme by which the original stream and the associated retransmission stream are sent into two different RTP sessions.

SSRC: synchronization source. See [3].

SSRC-multiplexing: scheme by which the original stream and the retransmission stream are sent in the same RTP session with different SSRC values.

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

### 3. Requirements and Design Rationale for a Retransmission Scheme

The use of retransmissions in RTP as a repair method for streaming media is appropriate in those scenarios with relaxed delay bounds and where full reliability is not a requirement. More specifically, RTP retransmission allows one to trade off reliability vs. delay; i.e., the endpoints may give up retransmitting a lost packet after a given buffering time has elapsed. Unlike TCP, there is thus no head-of-line blocking caused by RTP retransmissions. The implementer should be aware that in cases where full reliability is required or higher delay and jitter can be tolerated, TCP or other transport options should be considered.

The RTP retransmission scheme defined in this document is designed to fulfill the following set of requirements:

1. It must not break general RTP and RTCP mechanisms.
2. It must be suitable for unicast and small multicast groups.
3. It must work with mixers and translators.
4. It must work with all known payload types.
5. It must not prevent the use of multiple payload types in a session.

6. In order to support the largest variety of payload formats, the RTP receiver must be able to derive how many and which RTP packets were lost as a result of a gap in received RTP sequence numbers. This requirement is referred to as sequence number preservation. Without such a requirement, it would be impossible to use retransmission with payload formats, such as conversational text [9] or most audio/video streaming applications, that use the RTP sequence number to detect lost packets.

When designing a solution for RTP retransmission, several approaches may be considered for the multiplexing of the original RTP packets and the retransmitted RTP packets.

One approach may be to retransmit the RTP packet with its original sequence number and send original and retransmission packets in the same RTP stream. The retransmission packet would then be identical to the original RTP packet, i.e., the same header (and thus same sequence number) and the same payload. However, such an approach is not acceptable because it would corrupt the RTCP statistics. As a consequence, requirement 1 would not be met. Correct RTCP statistics require that for every RTP packet within the RTP stream, the sequence number be increased by one.

Another approach may be to multiplex original RTP packets and retransmission packets in the same RTP stream using different payload type values. With such an approach, the original packets and the retransmission packets would share the same sequence number space. As a result, the RTP receiver would not be able to infer how many and which original packets (which sequence numbers) were lost.

In other words, this approach does not satisfy the sequence number preservation requirement (requirement 6). This in turn implies that requirement 4 would not be met. Interoperability with mixers and translators would also be more difficult if they did not understand this new retransmission payload type in a sender RTP stream. For these reasons, a solution based on payload type multiplexing of original packets and retransmission packets in the same RTP stream is excluded.

Finally, the original and retransmission packets may be sent in two separate streams. These two streams may be multiplexed either by sending them in two different sessions, i.e., session-multiplexing, or in the same session using different SSRC values, i.e., SSRC-multiplexing. Since original and retransmission packets carry media of the same type, the objections in Section 5.2 of RTP [3] to RTP multiplexing do not apply in this case.

Mixers and translators may process the original stream and simply discard the retransmission stream if they are unable to utilise it.

On the other hand, sending the original and retransmission packets in two separate streams does not alone satisfy requirements 1 and 6. For this purpose, this document includes the original sequence number in the retransmitted packets.

In this manner, using two separate streams satisfies all the requirements listed in this section.

### 3.1. Multiplexing Scheme Choice

Session-multiplexing and SSRC-multiplexing have different pros and cons:

Session-multiplexing is based on sending the retransmission stream in a different RTP session (as defined in RTP [3]) from that of the original stream; i.e., the original and retransmission streams are sent to different network addresses and/or port numbers. Having a separate session allows more flexibility. In multicast, using two separate sessions for the original and the retransmission streams allows a receiver to choose whether or not to subscribe to the RTP session carrying the retransmission stream. The original session may also be single-source multicast while separate unicast sessions are used to convey retransmissions to each of the receivers, which as a result will receive only the retransmission packets they request.

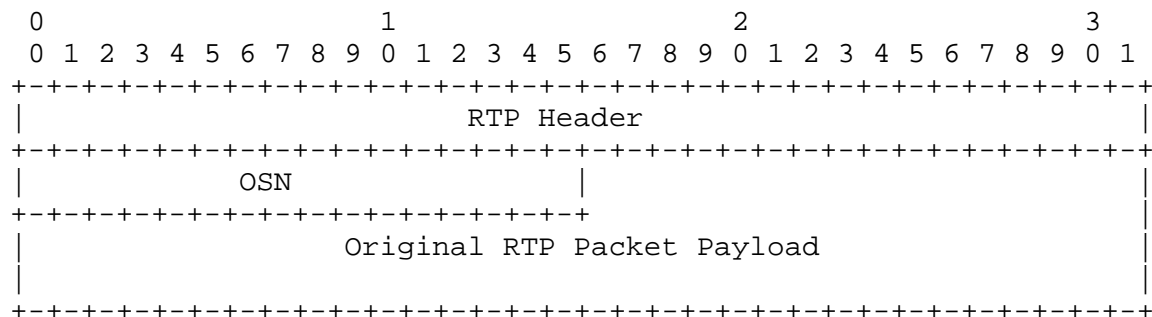
The use of separate sessions also facilitates differential treatment by the network and may simplify processing in mixers, translators, and packet caches.

With SSRC-multiplexing, a single session is needed for the original and the retransmission streams. This allows streaming servers and middleware that are involved in a high number of concurrent sessions to minimise their port usage.

This retransmission payload format allows both session-multiplexing and SSRC-multiplexing for unicast sessions. From an implementation point of view, there is little difference between the two approaches. Hence, in order to maximise interoperability, both multiplexing approaches SHOULD be supported by senders and receivers. For multicast sessions, session-multiplexing MUST be used because the association of the original stream and the retransmission stream is problematic if SSRC-multiplexing is used with multicast sessions (see Section 5.3 for motivation).

#### 4. Retransmission Payload Format

The format of a retransmission packet is shown below:



The RTP header usage is as follows:

In the case of session-multiplexing, the same SSRC value **MUST** be used for the original stream and the retransmission stream. In the case of an SSRC collision in either the original session or the retransmission session, the RTP specification requires that an RTCP BYE packet **MUST** be sent in the session where the collision happened. In addition, an RTCP BYE packet **MUST** also be sent for the associated stream in its own session. After a new SSRC identifier is obtained, the SSRC of both streams **MUST** be set to this value.

In the case of SSRC-multiplexing, two different SSRC values **MUST** be used for the original stream and the retransmission stream as required by RTP. If an SSRC collision is detected for either the original stream or the retransmission stream, the RTP specification requires that an RTCP BYE packet **MUST** be sent for this stream. An RTCP BYE packet **MUST NOT** be sent for the associated stream. Therefore, only the stream that experienced SSRC collision **MUST** choose a new SSRC value. Refer to Section 5.3 for the implications on the original stream and retransmission stream SSRC association at the receiver.

For either multiplexing scheme, the sequence number has the standard definition; i.e., it **MUST** be one higher than the sequence number of the preceding packet sent in the retransmission stream.

The retransmission packet timestamp **MUST** be set to the original timestamp, i.e., to the timestamp of the original packet. As a consequence, the initial RTP timestamp for the first packet of the retransmission stream is not random but equal to the original timestamp of the first packet that is retransmitted. See the Security Considerations section in this document for security implications.

Implementers have to be aware that the RTCP jitter value for the retransmission stream does not reflect the actual network jitter since there could be little correlation between the time a packet is retransmitted and its original timestamp.

The payload type is dynamic. If multiple payload types using retransmission are present in the original stream, then for each of these, a dynamic payload type MUST be mapped to the retransmission payload format. See Section 8.1 for the specification of how the mapping between original and retransmission payload types is done with Session Description Protocol (SDP).

As the retransmission packet timestamp carries the original media timestamp, the timestamp clockrate used by the retransmission payload type MUST be the same as the one used by the associated original payload type. Therefore, if an RTP stream carries payload types of different clockrates, this will also be the case for the associated retransmission stream. Note that an RTP stream does not usually carry payload types of different clockrates.

The payload of the RTP retransmission packet comprises the retransmission payload header followed by the payload of the original RTP packet. The length of the retransmission payload header is 2 octets. This payload header contains only one field, OSN (original sequence number), which MUST be set to the sequence number of the associated original RTP packet. The original RTP packet payload, including any possible payload headers specific to the original payload type, MUST be placed right after the retransmission payload header.

For payload formats that support encoding at multiple rates, instead of retransmitting the same payload as the original RTP packet the sender MAY retransmit the same data encoded at a lower rate. This aims at limiting the bandwidth usage of the retransmission stream. When doing so, the sender MUST ensure that the receiver will still be able to decode the payload of the already sent original packets that might have been encoded based on the payload of the lost original packet. In addition, if the sender chooses to retransmit at a lower rate, the values in the payload header of the original RTP packet may no longer apply to the retransmission packet and may need to be modified in the retransmission packet to reflect the change in rate. The sender SHOULD trade off the decrease in bandwidth usage with the decrease in quality caused by resending at a lower rate.

If the original RTP header carried any profile-specific extensions, the retransmission packet SHOULD include the same extensions immediately following the fixed RTP header as expected by applications running under this profile. In this case, the

retransmission payload header MUST be placed after the profile-specific extensions.

If the original RTP header carried an RTP header extension, the retransmission packet SHOULD carry the same header extension. This header extension MUST be placed right after the fixed RTP header, as specified in RTP [3]. In this case, the retransmission payload header MUST be placed after the header extension.

If the original RTP packet contained RTP padding, that padding MUST be removed before constructing the retransmission packet. If padding of the retransmission packet is needed, padding MUST be performed as with any RTP packets and the padding bit MUST be set.

The marker bit (M), the CSRC count (CC), and the CSRC list of the original RTP header MUST be copied "as is" into the RTP header of the retransmission packet.

## 5. Association of Retransmission and Original Streams

### 5.1. Retransmission Session Sharing

In the case of session-multiplexing, a retransmission session MUST map to exactly one original session; i.e., the same retransmission session cannot be used for different original sessions.

If retransmission session sharing were allowed, it would be a problem for receivers, since they would receive retransmissions for original sessions they might not have joined. For example, a receiver wishing to receive only audio would receive also retransmitted video packets if an audio and video session shared the same retransmission session.

### 5.2. CNAME Use

In both the session-multiplexing and the SSRC-multiplexing cases, a sender MUST use the same RTCP CNAME [3] for an original stream and its associated retransmission stream.

### 5.3. Association at the Receiver

A receiver receiving multiple original and retransmission streams needs to associate each retransmission stream with its original stream. The association is done differently depending on whether session-multiplexing or SSRC-multiplexing is used.

If session-multiplexing is used, the receiver associates the two streams having the same SSRC in the two sessions. Note that the payload type field cannot be used to perform the association as

several media streams may have the same payload type value. The two sessions are themselves associated out-of-band. See Section 8 for how the grouping of the two sessions is done with SDP.

If SSRC-multiplexing is used, the receiver should first of all look for two streams that have the same CNAME in the session. In some cases, the CNAME may not be enough to determine the association as multiple original streams in the same session may share the same CNAME. For example, there can be in the same video session multiple video streams mapping to different SSRCs and still using the same CNAME and possibly the same payload type (PT) values. Each (or some) of these streams may have an associated retransmission stream.

In this case, in order to find out the association between original and retransmission streams having the same CNAME, the receiver SHOULD behave as follows.

The association can generally be resolved when the receiver receives a retransmission packet matching a retransmission request that had been sent earlier. Upon reception of a retransmission packet whose original sequence number has been previously requested, the receiver can derive that the SSRC of the retransmission packet is associated to the sender SSRC from which the packet was requested.

However, this mechanism might fail if there are two outstanding requests for the same packet sequence number in two different original streams of the session. Note that since the initial packet sequence numbers are random, the probability of having two outstanding requests for the same packet sequence number would be very small. Nevertheless, in order to avoid ambiguity in the unicast case, the receiver MUST NOT have two outstanding requests for the same packet sequence number in two different original streams before the association is resolved. In multicast, this ambiguity cannot be completely avoided, because another receiver may have requested the same sequence number from another stream. Therefore, SSRC-multiplexing MUST NOT be used in multicast sessions.

If the receiver discovers that two senders are using the same SSRC or if it receives an RTCP BYE packet, it MUST stop requesting retransmissions for that SSRC. Upon reception of original RTP packets with a new SSRC, the receiver MUST perform the SSRC association again as described in this section.

## 6. Use with the Extended RTP Profile for RTCP-based Feedback

This section gives general hints for the usage of this payload format with the extended RTP profile for RTCP-based feedback, denoted AVPF [1]. Note that the general RTCP send and receive rules and the RTCP packet format as specified in RTP apply, except for the changes that the AVPF profile introduces. In short, the AVPF profile relaxes the RTCP timing rules and specifies additional general-purpose RTCP feedback messages. See [1] for details.

### 6.1. RTCP at the Sender

In the case of session-multiplexing, Sender Report (SR) packets for the original stream are sent in the original session and SR packets for the retransmission stream are sent in the retransmission session according to the rules of RTP.

In the case of SSRC-multiplexing, SR packets for both original and retransmission streams are sent in the same session according to the rules of RTP. The original and retransmission streams are seen, as far as the RTCP bandwidth calculation is concerned, as independent senders belonging to the same RTP session and are thus equally sharing the RTCP bandwidth assigned to senders.

Note that in both cases, session- and SSRC-multiplexing, BYE packets MUST still be sent for both streams as specified in RTP. In other words, it is not enough to send BYE packets for the original stream only.

### 6.2. RTCP Receiver Reports

In the case of session-multiplexing, the receiver will send report blocks for the original stream and the retransmission stream in separate Receiver Report (RR) packets belonging to separate RTP sessions. RR packets reporting on the original stream are sent in the original RTP session while RR packets reporting on the retransmission stream are sent in the retransmission session. The RTCP bandwidth for these two sessions may be chosen independently (e.g., through RTCP bandwidth modifiers [4]).

In the case of SSRC-multiplexing, the receiver sends report blocks for the original and the retransmission streams in the same RR packet since there is a single session.

### 6.3. Retransmission Requests

The NACK feedback message format defined in the AVPF profile SHOULD be used by receivers to send retransmission requests. Whether or not a receiver chooses to request a packet is an implementation issue. An actual receiver implementation should take into account such factors as the tolerable application delay, the network environment, and the media type.

The receiver should generally assess whether the retransmitted packet would still be useful at the time it is received. The timestamp of the missing packet can be estimated from the timestamps of packets preceding and/or following the sequence number gap caused by the missing packet in the original stream. In most cases, some form of linear estimate of the timestamp is good enough.

Furthermore, a receiver should compute an estimate of the round-trip time (RTT) to the sender. This can be done, for example, by measuring the retransmission delay to receive a retransmission packet after a NACK has been sent for that packet. This estimate may also be obtained from past observations, RTCP report round-trip time if available, or any other means. A standard mechanism for the receiver to estimate the RTT is specified in "RTP Control Protocol Extended Reports (RTCP XR)" [11].

The receiver should not send a retransmission request as soon as it detects a missing sequence number but should add some extra delay to compensate for packet reordering. This extra delay may, for example, be based on past observations of the experienced packet reordering. It should be noted that, in environments where packet reordering is rare or does not take place, e.g., if the underlying datalink layer affords ordered delivery, the delay may be extremely low or even take the value zero. In such cases, an appropriate "reorder delay" algorithm may not actually be timer based, but packet based. For example, if n number of packets are received after a gap is detected, then it may be assumed that the packet was truly lost rather than out of order. This may turn out to be far easier to code on some platforms as a very short fixed FIFO packet buffer as opposed to the timer-based mechanism.

To increase the robustness to the loss of a NACK or of a retransmission packet, a receiver may send a new NACK for the same packet. This is referred to as multiple retransmissions. Before sending a new NACK for a missing packet, the receiver should rely on a timer to be reasonably sure that the previous retransmission attempt has failed and so avoid unnecessary retransmissions. The timer value shall be based on the observed round-trip time. A static or an adaptive value MAY be used. For example, an adaptive timer

could be one that changes its value with every new request for the same packet. This document does not provide any guidelines as to how this adaptive value should be calculated because no experiments have been done to find this out.

NACKs MUST be sent only for the original RTP stream. Otherwise, if a receiver wanted to perform multiple retransmissions by sending a NACK in the retransmission stream, it would not be able to know the original sequence number and a timestamp estimation of the packet it requests.

Appendix A gives some guidelines as to how to control the number of retransmissions.

#### 6.4. Timing Rules

The NACK feedback message may be sent in a regular full compound RTCP packet or in an early RTCP packet, as per AVPF [1]. Sending a NACK in an early packet allows reacting more quickly to a given packet loss. However, in that case if a new packet loss occurs right after the early RTCP packet was sent, the receiver will then have to wait for the next regular RTCP compound packet after the early packet. Sending NACKs only in regular RTCP compound decreases the maximum delay between detecting an original packet loss and being able to send a NACK for that packet. Implementers should consider the possible implications of this fact for the application being used.

Furthermore, receivers may make use of the minimum interval between regular RTCP compound packets. This interval can be used to keep regular receiver reporting down to a minimum, while still allowing receivers to send early RTCP packets during periods requiring more frequent feedback, e.g., times of higher packet loss rate. Note that although RTCP packets may be suppressed because they do not contain NACKs, the same RTCP bandwidth as if they were sent needs to be available. See AVPF [1] for details on the use of the minimum interval.

#### 7. Congestion Control

RTP retransmission poses a risk of increasing network congestion. In a best-effort environment, packet loss is caused by congestion. Reacting to loss by retransmission of older data without decreasing the rate of the original stream would thus further increase congestion. Implementations SHOULD follow the recommendations below in order to use retransmission.

The RTP profile under which the retransmission scheme is used defines an appropriate congestion control mechanism in different environments. Following the rules under the profile, an RTP application can determine its acceptable bitrate and packet rate in order to be fair to other TCP or RTP flows.

If an RTP application uses retransmission, the acceptable packet rate and bitrate include both the original and retransmitted data. This guarantees that an application using retransmission achieves the same fairness as one that does not. Such a rule would translate in practice into the following actions:

If enhanced service is used, it should be made sure that the total bitrate and packet rate do not exceed that of the requested service. It should be further monitored that the requested services are actually delivered. In a best-effort environment, the sender **SHOULD NOT** send retransmission packets without reducing the packet rate and bitrate of the original stream (for example, by encoding the data at a lower rate).

In addition, the sender **MAY** selectively retransmit only the packets that it deems important and ignore NACK messages for other packets in order to limit the bitrate.

These congestion control mechanisms should keep the packet loss rate within acceptable parameters. In the context of congestion control, packet loss is considered acceptable if a TCP flow across the same network path and experiencing the same network conditions would achieve, on a reasonable timescale, an average throughput that is not less than the one the RTP flow achieves. If congestion is not kept under control, then retransmission **SHOULD NOT** be used.

Retransmissions **MAY** still be sent in some cases, e.g., in wireless links where packet losses are not caused by congestion, if the server (or the client that makes the retransmission request) estimates that a particular packet or frame is important to continue play out, or if an RTSP PAUSE has been issued to allow the buffer to fill up (RTSP PAUSE does not affect the sending of retransmissions).

Finally, it may further be necessary to adapt the transmission rate (or the number of layers subscribed for a layered multicast session), or to arrange for the receiver to leave the session.

## 8. Retransmission Payload Format MIME Type Registration

### 8.1. Introduction

The following MIME subtype name and parameters are introduced in this document: "rtx", "rtx-time", and "apt".

The binding used for the retransmission stream to the payload type number is indicated by an rtpmap attribute. The MIME subtype name used in the binding is "rtx".

The "apt" (associated payload type) parameter MUST be used to map the retransmission payload type to the associated original stream payload type. If multiple original payload types are used, then multiple "apt" parameters MUST be included to map each original payload type to a different retransmission payload type.

An OPTIONAL payload-format-specific parameter, "rtx-time", indicates the maximum time a sender will keep an original RTP packet in its buffers available for retransmission. This time starts with the first transmission of the packet.

The syntax is as follows:

```
a=fmtp:<number> apt=<apt-value>;rtx-time=<rtx-time-val>
```

where

<number>: indicates the dynamic payload type number assigned to the retransmission payload format in an rtpmap attribute.

<apt-value>: is the value of the original stream payload type to which this retransmission stream payload type is associated.

<rtx-time-val>: specifies the time in milliseconds (measured from the time a packet was first sent) that a sender keeps an RTP packet in its buffers available for retransmission. The absence of the rtx-time parameter for a retransmission stream means that the maximum retransmission time is not defined, but MAY be negotiated by other means.

## 8.2. Registration of audio/rtx

MIME type: audio

MIME subtype: rtx

Required parameters:

rate: the RTP timestamp clockrate is equal to the RTP timestamp clockrate of the media that is retransmitted.

apt: associated payload type. The value of this parameter is the payload type of the associated original stream.

Optional parameters:

rtx-time: indicates the time in milliseconds (measured from the time a packet was first sent) that the sender keeps an RTP packet in its buffers available for retransmission.

Encoding considerations: this type is only defined for transfer via RTP.

Security considerations: see Section 12 of RFC 4588

Interoperability considerations: none

Published specification: RFC 4588

Applications which use this media type: multimedia streaming applications

Additional information: none

Person & email address to contact for further information:  
jose.rey@eu.panasonic.com  
davidleon123@yahoo.com  
avt@ietf.org

Intended usage: COMMON

Authors:  
Jose Rey  
David Leon

Change controller:  
IETF AVT WG delegated from the IESG

### 8.3. Registration of video/rtx

MIME type: video

MIME subtype: rtx

Required parameters:

rate: the RTP timestamp clockrate is equal to the RTP timestamp clockrate of the media that is retransmitted.

apt: associated payload type. The value of this parameter is the payload type of the associated original stream.

Optional parameters:

rtx-time: indicates the time in milliseconds (measured from the time a packet was first sent) that the sender keeps an RTP packet in its buffers available for retransmission.

Encoding considerations: this type is only defined for transfer via RTP.

Security considerations: see Section 12 of RFC 4588

Interoperability considerations: none

Published specification: RFC 4588

Applications which use this media type: multimedia streaming applications

Additional information: none

Person & email address to contact for further information:  
jose.rey@eu.panasonic.com  
davidleon123@yahoo.com  
avt@ietf.org

Intended usage: COMMON

Authors:  
Jose Rey  
David Leon

Change controller:  
IETF AVT WG delegated from the IESG

#### 8.4. Registration of text/rtx

MIME type: text

MIME subtype: rtx

Required parameters:

rate: the RTP timestamp clockrate is equal to the RTP timestamp clockrate of the media that is retransmitted.

apt: associated payload type. The value of this parameter is the payload type of the associated original stream.

Optional parameters:

rtx-time: indicates the time in milliseconds (measured from the time a packet was first sent) that the sender keeps an RTP packet in its buffers available for retransmission.

Encoding considerations: this type is only defined for transfer via RTP.

Security considerations: see Section 12 of RFC 4588

Interoperability considerations: none

Published specification: RFC 4588

Applications which use this media type: multimedia streaming applications

Additional information: none

Person & email address to contact for further information:  
jose.rey@eu.panasonic.com  
davidleon123@yahoo.com  
avt@ietf.org

Intended usage: COMMON

Authors:  
Jose Rey  
David Leon

Change controller:  
IETF AVT WG delegated from the IESG

## 8.5. Registration of application/rtx

MIME type: application

MIME subtype: rtx

Required parameters:

rate: the RTP timestamp clockrate is equal to the RTP timestamp clockrate of the media that is retransmitted.

apt: associated payload type. The value of this parameter is the payload type of the associated original stream.

Optional parameters:

rtx-time: indicates the time in milliseconds (measured from the time a packet was first sent) that the sender keeps an RTP packet in its buffers available for retransmission.

Encoding considerations: this type is only defined for transfer via RTP.

Security considerations: see Section 12 of RFC 4588

Interoperability considerations: none

Published specification: RFC 4588

Applications which use this media type: multimedia streaming applications

Additional information: none

Person & email address to contact for further information:  
jose.rey@eu.panasonic.com  
davidleon123@yahoo.com  
avt@ietf.org

Intended usage: COMMON

Authors:  
Jose Rey  
David Leon

Change controller:  
IETF AVT WG delegated from the IESG

## 8.6. Mapping to SDP

The information carried in the MIME media type specification has a specific mapping to fields in SDP [5], which is commonly used to describe RTP sessions. When SDP is used to specify retransmissions for an RTP stream, the mapping is done as follows:

- The MIME types ("video"), ("audio"), ("text"), and ("application") go in the SDP "m=" as the media name.
- The MIME subtype ("rtx") goes in SDP "a=rtpmap" as the encoding name. The RTP clockrate in "a=rtpmap" MUST be that of the retransmission payload type. See Section 4 for details on this.
- The AVPF profile-specific parameters "ack" and "nack" go in SDP "a=rtcp-fb". Several SDP "a=rtcp-fb" are used for several types of feedback. See the AVPF profile [1] for details.
- The retransmission payload-format-specific parameters "apt" and "rtx-time" go in the SDP "a=fmtp" as a semicolon-separated list of parameter=value pairs.
- Any remaining parameters go in the SDP "a=fmtp" attribute by copying them directly from the MIME media type string as a semicolon-separated list of parameter=value pairs.

In the following sections, some example SDP descriptions are presented. In some of these examples, long lines are folded to meet the column width constraints of this document; the backslash ("\") at the end of a line and the carriage return that follows it should be ignored.

## 8.7. SDP Description with Session-Multiplexing

In the case of session-multiplexing, the SDP description contains one media specification "m" line per RTP session. The SDP MUST provide the grouping of the original and associated retransmission sessions' "m" lines, using the Flow Identification (FID) semantics defined in RFC 3388 [6].

The following example specifies two original, AMR and MPEG-4, streams on ports 49170 and 49174 and their corresponding retransmission streams on ports 49172 and 49176, respectively:

```
v=0
o=mascha 2980675221 2980675778 IN IP4 host.example.net
c=IN IP4 192.0.2.0
a=group:FID 1 2
```

```
a=group:FID 3 4
m=audio 49170 RTP/AVPF 96
a=rtpmap:96 AMR/8000
a=fmtp:96 octet-align=1
a=rtcp-fb:96 nack
a=mid:1
m=audio 49172 RTP/AVPF 97
a=rtpmap:97 rtx/8000
a=fmtp:97 apt=96;rtx-time=3000
a=mid:2
m=video 49174 RTP/AVPF 98
a=rtpmap:98 MP4V-ES/90000
a=rtcp-fb:98 nack
a=fmtp:98 profile-level-id=8;config=01010000012000884006682C209\
0A21F
a=mid:3
m=video 49176 RTP/AVPF 99
a=rtpmap:99 rtx/90000
a=fmtp:99 apt=98;rtx-time=3000
a=mid:4
```

A special case of the SDP description is a description that contains only one original session "m" line and one retransmission session "m" line, the grouping is then obvious and FID semantics MAY be omitted in this special case only.

This is illustrated in the following example, which is an SDP description for a single original MPEG-4 stream and its corresponding retransmission session:

```
v=0
o=mascha 2980675221 2980675778 IN IP4 host.example.net
c=IN IP4 192.0.2.0
m=video 49170 RTP/AVPF 96
a=rtpmap:96 MP4V-ES/90000
a=rtcp-fb:96 nack
a=fmtp:96 profile-level-id=8;config=01010000012000884006682C209\
0A21F
m=video 49172 RTP/AVPF 97
a=rtpmap:97 rtx/90000
a=fmtp:97 apt=96;rtx-time=3000
```

#### 8.8. SDP Description with SSRC-Multiplexing

The following is an example of an SDP description for an RTP video session using SSRC-multiplexing with similar parameters as in the single-session example above:

```
v=0
o=mascha 2980675221 2980675778 IN IP4 host.example.net
c=IN IP4 192.0.2.0
m=video 49170 RTP/AVPF 96 97
a=rtpmap:96 MP4V-ES/90000
a=rtcp-fb:96 nack
a=fmtp:96 profile-level-id=8;config=01010000012000884006682C209\
0A21F
a=rtpmap:97 rtx/90000
a=fmtp:97 apt=96;rtx-time=3000
```

## 9. RTSP Considerations

The Real Time Streaming Protocol (RTSP), RFC 2326 [7], is an application-level protocol for control over the delivery of data with real-time properties. This section looks at the issues involved in controlling RTP sessions that use retransmissions.

### 9.1. RTSP Control with SSRC-Multiplexing

In the case of SSRC-multiplexing, the "m" line includes both original and retransmission payload types and has a single RTSP "control" attribute. The receiver uses the "m" line to request SETUP and TEARDOWN of the whole media session. The RTP profile contained in the Transport header MUST be the AVPF profile or another suitable profile allowing extended feedback. If the SSRC value is included in the SETUP response's Transport header, it MUST be that of the original stream.

In order to control the sending of the session original media stream, the receiver sends as usual PLAY and PAUSE requests to the sender for the session. The RTP-info header that is used to set RTP-specific parameters in the PLAY response MUST be set according to the RTP information of the original stream.

When the receiver starts receiving the original stream, it can then request retransmission through RTCP NACKs without additional RTSP signalling.

### 9.2. RTSP Control with Session-Multiplexing

In the case of session-multiplexing, each SDP "m" line has an RTSP "control" attribute. Hence, when retransmission is used, both the original session and the retransmission have their own "control" attributes. The receiver can associate the original session and the retransmission session through the FID semantics as specified in Section 8.

The original and the retransmission streams are set up and torn down separately through their respective media "control" attribute. The RTP profile contained in the Transport header MUST be the AVPF profile or another suitable profile allowing extended feedback for both the original and the retransmission sessions.

The RTSP presentation SHOULD support aggregate control and SHOULD contain a session-level RTSP URL. The receiver SHOULD use aggregate control for an original session and its associated retransmission session. Otherwise, there would need to be two different 'session-id' values, i.e., different values for the original and retransmission sessions, and the sender would not know how to associate them.

The session-level "control" attribute is then used as usual to control the playing of the original stream. When the receiver starts receiving the original stream, it can then request retransmissions through RTCP without additional RTSP signalling.

### 9.3. RTSP Control of the Retransmission Stream

Because of the nature of retransmissions, the sending of retransmission packets SHOULD NOT be controlled through RTSP PLAY and PAUSE requests. The PLAY and PAUSE requests SHOULD NOT affect the retransmission stream. Retransmission packets are sent upon receiver requests in the original RTCP stream, regardless of the state.

### 9.4. Cache Control

Retransmission streams SHOULD NOT be cached.

In the case of session-multiplexing, the "Cache-Control" header SHOULD be set to "no-cache" for the retransmission stream.

In the case of SSRC-multiplexing, RTSP cannot specify independent caching for the retransmission stream, because there is a single "m" line in SDP. Therefore, the implementer should take this fact into account when deciding whether or not to cache an SSRC-multiplexed session.

## 10. Implementation Examples

This document mandates only the sender and receiver behaviours that are necessary for interoperability. In addition, certain algorithms, such as rate control or buffer management when targeted at specific environments, may enhance the retransmission efficiency.

This section gives an overview of different implementation options allowed within this specification.

The first example describes a minimal receiver implementation. With this implementation, it is possible to retransmit lost RTP packets, detect efficiently the loss of retransmissions, and perform multiple retransmissions, if needed. Most of the necessary processing is done at the server.

The second example shows how retransmissions may be used in (small) multicast groups in conjunction with layered encoding. It illustrates that retransmissions and layered encoding may be complementary techniques.

#### 10.1. A Minimal Receiver Implementation Example

This section gives an example of an implementation supporting multiple retransmissions. The sender transmits the original data in RTP packets using the MPEG-4 video RTP payload format. It is assumed that NACK feedback messages are used, as per [1]. An SDP description example with SSRC-multiplexing is given below:

```
v=0
o=mascha 2980675221 2980675778 IN IP4 host.example.net
c=IN IP4 192.0.2.0
m=video 49170 RTP/AVPF 96 97
a=rtpmap:96 MP4V-ES/90000
a=rtcp-fb:96 nack
a=rtpmap:97 rtx/90000
a=fmtp:97 apt=96;rtx-time=3000
```

The format-specific parameter "rtx-time" indicates that the server will buffer the sent packets in a retransmission buffer for 3.0 seconds, after which the packets are deleted from the retransmission buffer and will never be sent again.

In this implementation example, the required RTP receiver processing to handle retransmission is kept to a minimum. The receiver detects packet loss from the gaps observed in the received sequence numbers. It signals lost packets to the sender through NACKs as defined in the AVPF profile [1]. The receiver should take into account the signalled sender retransmission buffer length in order to dimension its own reception buffer. It should also derive from the buffer length the maximum number of times the retransmission of a packet can be requested.

The sender should retransmit the packets selectively; i.e., it should choose whether to retransmit a requested packet depending on the packet importance, the observed Quality of Service (QoS), and congestion state of the network connection to the receiver. Obviously, the sender processing increases with the number of receivers as state information and processing load must be allocated to each receiver.

## 10.2. Retransmission of Layered Encoded Media in Multicast

This section shows how to combine retransmissions with layered encoding in multicast sessions. Note that the retransmission framework is offered only for small multicast applications. Refer to RFC 2887 [10] for a discussion of the problems of NACK implosion, severe congestion caused by feedback traffic, in large-group reliable multicast applications.

Packets of different importance are sent in different RTP sessions. The retransmission streams corresponding to the different layers can themselves be seen as different retransmission layers. The relative importance of the different retransmission streams should reflect the relative importance of the different original streams.

In multicast, SSRC-multiplexing of the original and retransmission streams is not allowed as per Section 5.3 of this document. For this reason, the retransmission stream(s) MUST be sent in different RTP session(s) using session-multiplexing.

An SDP description example of multicast retransmissions for layered encoded media is given below:

```
m=video 8000 RTP/AVPF 98
c=IN IP4 224.2.1.0/127/3
a=rtpmap:98 MP4V-ES/90000
a=rtcp-fb:98 nack
m=video 8000 RTP/AVPF 99
c=IN IP4 224.2.1.3/127/3
a=rtpmap:99 rtx/90000
a=fmtp:99 apt=98;rtx-time=3000
```

The server and the receiver may implement the retransmission methods illustrated in the previous examples. In addition, they may choose to request and retransmit a lost packet depending on the layer it belongs to.

## 11. IANA Considerations

A new MIME subtype name, "rtx", has been registered for four different media types, as follows: "video", "audio", "text" and "application". An additional REQUIRED parameter, "apt", and an OPTIONAL parameter, "rtx-time", are defined. See Section 8 for details.

## 12. Security Considerations

RTP packets using the payload format defined in this specification are subject to the general security considerations discussed in RTP [3], Section 9.

In common streaming scenarios message authentication, data integrity, replay protection, and confidentiality are desired.

The absence of authentication may enable man-in-the-middle and replay attacks, which can be very harmful for RTP retransmission. For example: tampered RTCP packets may trigger inappropriate retransmissions that effectively reduce the actual bitrate share allocated to the original data stream, tampered RTP retransmission packets could cause the client's decoder to crash, and tampered retransmission requests may invalidate the SSRC association mechanism described in Section 5 of this document. On the other hand, replayed packets could lead to false reordering and RTT measurements (required for the retransmission request strategy) and may cause the receiver buffer to overflow.

Furthermore, in order to ensure confidentiality of the data, the original payload data needs to be encrypted. There is actually no need to encrypt the 2-byte retransmission payload header since it does not provide any hints about the data content.

Furthermore, it is RECOMMENDED that the cryptography mechanisms used for this payload format provide protection against known plaintext attacks. RTP recommends that the initial RTP timestamp SHOULD be random to secure the stream against known plaintext attacks. This payload format does not follow this recommendation as the initial timestamp will be the media timestamp of the first retransmitted packet. However, since the initial timestamp of the original stream is itself random, if the original stream is encrypted, the first retransmitted packet timestamp would also be random to an attacker. Therefore, confidentiality would not be compromised.

If cryptography is used to provide security services on the original stream, then the same services, with equivalent cryptographic strength, MUST be provided on the retransmission stream. The use of

the same key for the retransmitted stream and the original stream may lead to security problems, e.g., two-time pads. Refer to Section 9.1 of the Secure Real-Time Transport Protocol (SRTP) [12] for a discussion the implications of two-time pads and how to avoid them.

At the time of writing this document, SRTP does not provide all the security services mentioned. There are, at least, two reasons for this: 1) the occurrence of two-time pads and 2) the fact that this payload format typically works under the RTP/AVPF profile whereas SRTP only supports RTP/AVP. An adapted variant of SRTP shall solve these shortcomings in the future.

Congestion control considerations with the use of retransmission are dealt with in Section 7 of this document.

### 13. Acknowledgements

We would like to express our gratitude to Carsten Burmeister for his participation in the development of this document. Our thanks also go to Koichi Hata, Colin Perkins, Stephen Casner, Magnus Westerlund, Go Hori, and Rahul Agarwal for their helpful comments.

### 14. References

#### 14.1. Normative References

- [1] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP profile for Real-time Transport Control Protocol (RTCP)-Based feedback", RFC 4585, July 2006.
- [2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [3] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.
- [4] Casner, S., "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", RFC 3556, July 2003.
- [5] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.
- [6] Camarillo, G., Eriksson, G., Holler, J., and H. Schulzrinne, "Grouping of Media Lines in the Session Description Protocol (SDP)", RFC 3388, December 2002.

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

#### 14.2. Informative References

- [8] Perkins, C. and O. Hodson, "Options for Repair of Streaming Media", RFC 2354, June 1998.
- [9] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, June 2005.
- [10] Handley, M., Floyd, S., Whetten, B., Kermode, R., Vicisano, L., and M. Luby, "The Reliable Multicast Design Space for Bulk Data Transfer", RFC 2887, August 2000.
- [11] Friedman, T., Caceres, R., and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.
- [12] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

## Appendix A. How to Control the Number of Rtxs. per Packet

Finding out the number of retransmissions (rtxs.) per packet for achieving the best possible transmission is a difficult task. Of course, the absolute minimum should be one (1); otherwise, do not use this payload format. Moreover, as of date of publication, the authors were not aware of any studies on the number of retransmissions per packet that should be used for best performance. To help implementers and researchers on this item, this section describes an estimate of the buffering time required for achieving a given number of retransmissions. Once this time has been calculated, it can be communicated to the client via SDP parameter "rtx-time", as defined in this document.

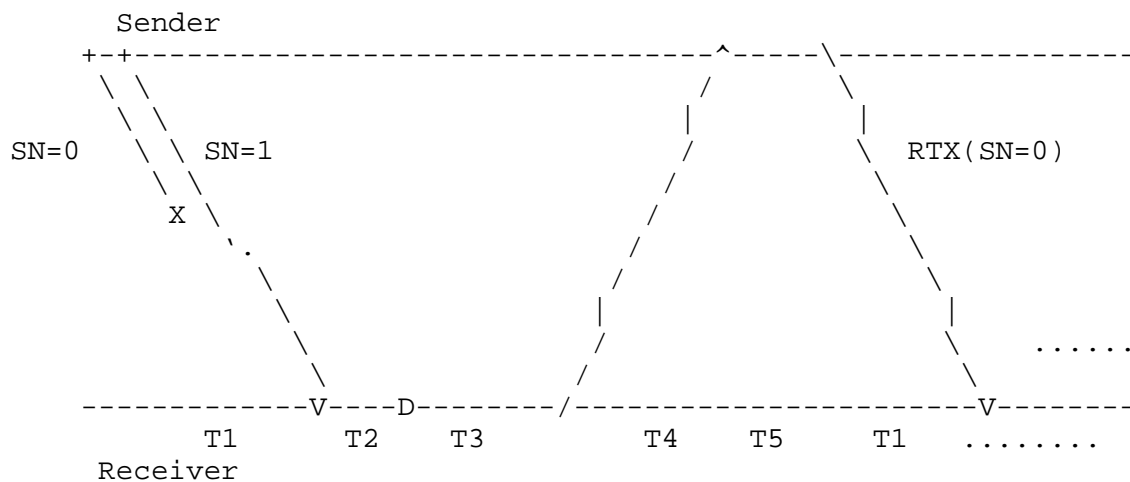
### A.1. Scenario and Assumptions

- \* Streaming scenario with relaxed delay bounds. Client and server are provided with buffering space as indicated by the parameter "rtx-time" in SDP.
- \* RTP AVPF profile used with SSRC-multiplexing retransmission scheme: 1 SSRC for original packets, 1 for retransmission packets.
- \* Default RTCP bandwidth share for SRs and RRs, i.e.,  $SR+RR = 0.05$ . We have senders (2) and receivers (1). Receivers and senders get equally  $1/3$  of the RTCP bandwidth share because the proportion of senders is greater than  $1/4$  of session members.
- \* avg-rtcp-size is approximated by 120 bytes. This is a rounded-up average of 2 SRs, one for each SSRC, containing 40/8/28/32 bytes for IPv6/UDP/SR/SDES with CNAME, thus making 105 bytes each; and a RR with 40/8/64/32 bytes for IPv6/UDP/2\*RR/SDES, making 157 bytes. Since senders and receivers share the RTCP bandwidth equally, then  $avg-rtcp-size = (157+105+105)/3 = 117.3 \approx 120$  bytes. The important characteristic of this value is that it is something over 100 bytes, which seems to be a representative figure for typical configurations.
- \* The profile used is AVPF [1] and Generic NACKs are used for requesting retransmissions. This adds 16 bytes of overhead for 1 NACK and 4 bytes more for every additional NACK Feedback Control Information (FCI) field.
- \* We assume a worst-case scenario in which each packet exhausts its corresponding number of available retransmissions, N, before it is received. This means that if a packet is requested for retransmission a maximum of 2 times, the corresponding generic NACK report block requesting that particular packet is sent in two

consecutive RTCP compounds; likewise, if it is requested for retransmission 10 times, then the generic NACK is sent 10 times. This assumption makes the RTCP packet size approximately constant after  $N \cdot \text{RTCP intervals (seconds)}$ , namely, to  $\text{avg-rtcp-size} = 120 + (\text{receiver-RTCP-bw-share}) \cdot (12 + 4 \cdot N)$ . In our case, the receiver RTCP bandwidth share is  $1/3$ ; thus,  $\text{avg-rtcp-size} = 124 + 4 \cdot N/3$ .

- \* Two delay parameters are difficult to approximate and may be implementation dependent. Therefore, we list them here explicitly without assigning them a particular value: one is the packet loss detection time ( $T2$ ), and the other is feedback processing and queuing time for retransmissions ( $T5$ ). Implementers shall assign appropriate values to these two parameters.

Graphically, we have the following:



Legend:

=====

DL: downlink (client->server)

UL: uplink (server->client)

Time unit is seconds, s.

Bitrate unit is bits per second, bps.

DL transmission time:  $T1 = \text{physical-delay-DL} + \text{tx-delay-DL} (= \text{avg-pkt-size} / \text{DL-bitrate}) + \text{interarrival-delay-jitter}$

Time to detect packet loss:  $T2 = \text{pkt-loss-detect-time}$

Time to report packet loss:  $T3 = \text{time-to-next-rtcp-report}$

UL transmission time:  $T4 = \text{physical-delay-UL} + \text{transmission-delay-UL} + \text{interarrival-delay-jitter}$

Retransmissions processing time:  $T5 = \text{feedback-processing-time} + \text{rtx-queuing-time}$

## A.2. Goal

To find an estimate of the buffering time,  $T()$ , that a streaming server shall use in order to enable a given number of retransmissions for each packet,  $N$ . This time is approximately equal at the server and at the client, if one considers that the client starts buffering  $T1$  seconds later.

## A.3. Solution

First, we find the value of the estimate for 1 retransmission,  $T(1)=T$ :

$$T = T1 + T2 + T3 + T4 + T5$$

Since  $T1 + T4 \approx \text{RTT}$ ,

$$T = \text{RTT} + T2 + T3 + T5$$

The worst case for  $T3$  would be that we assume that reporting has to wait a whole RTCP interval and that the maximum randomization factor of 1.5 is applied. Therefore, after applying the subsequent compensation to avoid traffic bursts (see Appendix A.7 of RTP [3]), we have that  $T3 = 1.5/1.21828*\text{RTCP-Interval}$ . Thus,

$$T = \text{RTT} + 1.2312*\text{RTCP-Interval} + T2 + T5$$

On the other hand,  $\text{RTCP-Interval} = \text{avg-rtcp-size}*8*(\text{senders} + \text{receivers})/(\text{RR}+\text{RS})$ . In this scenario:  $\text{sender} + \text{receivers} = 3$ ;  $\text{RR}+\text{RS}$  is the receiver report plus sender report bandwidth share, in this case, equal to the default 5% of session bandwidth,  $\text{bw}$ . We assume an average RTCP packet size,  $\text{avg-rtcp-size} = 120$  bytes. Thus:

$$T = \text{RTT} + 1.2312*\text{avg-rtcp-size}*8*3/(0.05*\text{bw}) + T2 + T5$$

for 1 retransmission.

For enabling  $N$  retransmissions, the available buffering time in a streaming server or client is approximately:

$$T(N) = N*(\text{RTT}+1.2312*\text{avg-rtcp-size}*8*3/(0.05*\text{bw}) + T2 + T5)$$

where, as per above,

$$\begin{aligned}\text{avg-rtcp-size} &= 120 + (\text{receiver-RTCP-bw-share}) * (12 + 4 * N) \\ &= 120 + (1/3) * (12 + 4 * N) \\ &= 124 + 4 * N / 3.\end{aligned}$$

#### A.4. Numbers

If we ignore the effect of T2 and T5, i.e., assume that all losses are detected immediately and that there is no additional delay due to feedback processing or retransmission queuing, we have the following buffering times for different values of N:

RTCP w/ several Generic NACKs; variable packet size =  $124 + 4 * N / 3$  bytes

=====	=====	=====				
RTP BW	RTT	N value				
=====	=====	1	2	5	7	10
=====						
64000	0,05	1,21	2,44	6,28	8,97	13,18
128000	0,05	0,63	1,27	3,27	4,66	6,84
256000	0,05	0,34	0,68	1,76	2,50	3,67
512000	0,05	0,19	0,39	1,00	1,43	2,09
1024000	0,05	0,12	0,25	0,63	0,89	1,29
5000000	0,05	0,06	0,13	0,33	0,46	0,66
10000000	0,05	0,06	0,11	0,29	0,41	0,58
64000	0,2	1,36	2,74	7,03	10,02	14,68
128000	0,2	0,78	1,57	4,02	5,71	8,34
256000	0,2	0,49	0,98	2,51	3,55	5,17
512000	0,2	0,34	0,69	1,75	2,48	3,59
1024000	0,2	0,27	0,55	1,38	1,94	2,79
5000000	0,2	0,21	0,43	1,08	1,51	2,16
10000000	0,2	0,21	0,41	1,04	1,46	2,08
64000	1	2,16	4,34	11,03	15,62	22,68
128000	1	1,58	3,17	8,02	11,31	16,34
256000	1	1,29	2,58	6,51	9,15	13,17
512000	1	1,14	2,29	5,75	8,08	11,59
1024000	1	1,07	2,15	5,38	7,54	10,79
5000000	1	1,01	2,03	5,08	7,11	10,16
10000000	1	1,01	2,01	5,04	7,06	10,08

To quantify the error of not taking the Generic NACKS into account, we can do the same numbers, but ignoring the Generic NACK contribution, avg-rtcp-size  $\approx$  120 bytes. As we see from below, this may result in a buffer estimation error of 1-1.5 seconds (5-10%) for lower bandwidth values and higher number of retransmissions. This effect is low in this case. Nevertheless, it should be carefully evaluated for the particular scenario; that is why the formula includes it.

RTCP w/o Generic NACK, fixed packet size  $\approx$  120 bytes

RTP BW	RTT	N value				
		1	2	5	7	10
64000	0,05	1,16	2,32	5,79	8,11	11,58
128000	0,05	0,60	1,21	3,02	4,23	6,04
256000	0,05	0,33	0,65	1,64	2,29	3,27
512000	0,05	0,19	0,38	0,94	1,32	1,89
1024000	0,05	0,12	0,24	0,60	0,83	1,19
5000000	0,05	0,06	0,13	0,32	0,45	0,64
10000000	0,05	0,06	0,11	0,29	0,40	0,57
64000	0,2	1,31	2,62	6,54	9,16	13,08
128000	0,2	0,75	1,51	3,77	5,28	7,54
256000	0,2	0,48	0,95	2,39	3,34	4,77
512000	0,2	0,34	0,68	1,69	2,37	3,39
1024000	0,2	0,27	0,54	1,35	1,88	2,69
5000000	0,2	0,21	0,43	1,07	1,50	2,14
10000000	0,2	0,21	0,41	1,04	1,45	2,07
64000	1	2,11	4,22	10,54	14,76	21,08
128000	1	1,55	3,11	7,77	10,88	15,54
256000	1	1,28	2,55	6,39	8,94	12,77
512000	1	1,14	2,28	5,69	7,97	11,39
1024000	1	1,07	2,14	5,35	7,48	10,69
5000000	1	1,01	2,03	5,07	7,10	10,14
10000000	1	1,01	2,01	5,04	7,05	10,07

## Authors' Addresses

Jose Rey  
Panasonic R&D Center Germany GmbH  
Monzastr. 4c  
D-63225 Langen, Germany

Phone: +49-6103-766-134  
Fax: +49-6103-766-166  
EMail: jose.rey@eu.panasonic.com

David Leon  
Consultant

EMail: davidleon123@yahoo.com

Akihiro Miyazaki  
Matsushita Electric Industrial Co., Ltd  
1006, Kadoma, Kadoma City, Osaka, Japan

Phone: +81-6-6900-9172  
Fax: +81-6-6900-9173  
EMail: miyazaki.akihiro@jp.panasonic.com

Viktor Varsa  
Nokia Research Center  
6000 Connection Drive  
Irving, TX. USA

Phone: 1-972-374-1861  
EMail: viktor.varsa@nokia.com

Rolf Hakenberg  
Panasonic R&D Center Germany GmbH  
Monzastr. 4c  
D-63225 Langen, Germany

Phone: +49-6103-766-162  
Fax: +49-6103-766-166  
EMail: rolf.hakenberg@eu.panasonic.com

## Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at [ietf-ipr@ietf.org](mailto:ietf-ipr@ietf.org).

## Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

