

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

H. Lundin
S. Holmer
Google
L. De Cicco
S. Mascolo
Politecnico di Bari
H. Alvestrand, Ed.
Google
February 14, 2014

A Google Congestion Control Algorithm for Real-Time Communication
draft-alvestrand-rmcat-congestion-02

Abstract

This document describes two methods of congestion control when using real-time communications on the World Wide Web (RTCWEB); one sender-based and one receiver-based.

It is published as an input document to the RMCAT working group on congestion control for media streams. The mailing list of that WG is rmcat@ietf.org.

Requirements Language

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 [RFC2119].

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 August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Mathematical notation conventions	3
2. System model	4
3. Receiver side control	5
3.1. Procseing multiple streams using RTP timestamp to NTP time conversion	5
3.2. Arrival-time model	5
3.3. Arrival-time filter	7
3.4. Over-use detector	8
3.5. Rate control	10
4. Sender side control	12
5. Interoperability Considerations	14
6. Implementation Experience	14
7. Further Work	14
8. IANA Considerations	16
9. Security Considerations	16
10. Acknowledgements	16
11. References	16
11.1. Normative References	16
11.2. Informative References	17
Appendix A. Change log	17
A.1. Version -00 to -01	17
A.2. Version -01 to -02	18
A.3. Version -02 to -03	18
A.4. rtcweb-03 to rmcat-00	18
A.5. rmcat -00 to -01	18
A.6. rmcat -01 to -02	18
Authors' Addresses	18

1. Introduction

Congestion control is a requirement for all applications that wish to share the Internet [RFC2914].

The problem of doing congestion control for real-time media is made difficult for a number of reasons:

- o The media is usually encoded in forms that cannot be quickly changed to accommodate varying bandwidth, and bandwidth requirements can often be changed only in discrete, rather large steps
- o The participants may have certain specific wishes on how to respond - which may not be reducing the bandwidth required by the flow on which congestion is discovered
- o The encodings are usually sensitive to packet loss, while the real time requirement precludes the repair of packet loss by retransmission

This memo describes two congestion control algorithms that together are seen to give reasonable performance and reasonable (not perfect) bandwidth sharing with other conferences and with TCP-using applications that share the same links.

The signaling used consists of standard RTP timestamps [RFC3550] possibly augmented with RTP transmission time offsets [RFC5450], standard RTCP feedback reports and Temporary Maximum Media Stream Bit Rate Requests (TMMBR) as defined in [RFC5104] section 3.5.4, or by using the REMB feedback report defined in [I-D.alvestrand-remb]

1.1. Mathematical notation conventions

The mathematics of this document have been transcribed from a more formula-friendly format.

The following notational conventions are used:

\bar{X} The variable X , where X is a vector - conventionally marked by a bar on top of the variable name.

\hat{X} An estimate of the true value of variable X - conventionally marked by a circumflex accent on top of the variable name.

$X(i)$ The " i "th value of X - conventionally marked by a subscript i .

$[x\ y\ z]$ A row vector consisting of elements x , y and z .

X_{bar}^T The transpose of vector X_{bar} .

$E\{X\}$ The expected value of the stochastic variable X

2. System model

The following elements are in the system:

- o RTP packet - an RTP packet containing media data.
- o Frame - a set of RTP packets transmitted from the sender at the same time instant. This could be a video frame, an audio frame, or a mix of audio and video packets. A frame can be defined by the RTP packet send time (RTP timestamp + transmission time offset), or by the RTP timestamp if the transmission time offset field is not present.
- o Incoming media streams - a stream of frames consisting of RTP packets.
- o Media codec - has a bandwidth control, and encodes the incoming media stream into an RTP stream.
- o RTP sender - sends the RTP stream over the network to the RTP receiver. Generates the RTP timestamp.
- o RTP receiver - receives the RTP stream, notes the time of arrival. Regenerates the media stream for the recipient.
- o RTCP sender at RTP sender - sends sender reports with mappings between RTP timestamps and NTP time.
- o RTCP sender at RTP receiver - sends receiver reports and TMMBR/REMB messages.
- o RTCP receiver at RTP sender - receives receiver reports and TMMBR/REMB messages, reports these to sender side control.
- o RTCP receiver at RTP receiver.
- o Sender side control - takes loss rate info, round trip time info, and TMMBR/REMB messages and computes a sending bitrate.
- o Receiver side control - takes the packet arrival info at the RTP receiver and decides when to send TMMBR/REMB messages.

Together, sender side control and receiver side control implement the congestion control algorithm.

3. Receiver side control

The receive-side algorithm can be further decomposed into four parts: an RTP timestamp to NTP time conversion, arrival-time filter, an over-use detector, and a remote rate-control.

3.1. Processing multiple streams using RTP timestamp to NTP time conversion

It is common that multiple RTP streams are sent from the sender to the receiver. In such a situation the RTP timestamps of incoming can first be converted to a common time base using the RTP timestamp and NTP time pairs in RTCP SR reports[RFC3550]. The converted timestamps can then be used instead of RTP timestamps in the arrival-time filtering, and since all streams from the same sender have timestamps in the same time base they can all be processed by the same filter. This has the advantage of quicker reactions and reduces problems of noisy measurements due to self-inflicted cross-traffic.

In the time interval from the start of the call until a stream from the same sender has received an RTCP SR report, the receiver-side control operates in single-stream mode. In that mode only one RTP stream can be processed by the over-use detector. As soon as a stream has received one or more RTCP SR reports the receiver-side control can change to a multi-stream mode, where all RTP streams from the same sender which have received one or more RTCP SR reports can be processed by the over-use detector. When switching to the multi-stream mode the state of the over-use detector must be modified to avoid a time base mismatch. This can either be done by resetting the stored RTP timestamp values or by converting them using the newly received RTCP SR report.

3.2. Arrival-time model

This section describes an adaptive filter that continuously updates estimates of network parameters based on the timing of the received frames.

At the receiving side we are observing groups of incoming packets, where each group of packets corresponding to the same frame having timestamp $T(i)$.

Each frame is assigned a receive time $t(i)$, which corresponds to the time at which the whole frame has been received (ignoring any packet losses). A frame is delayed relative to its predecessor if $t(i) - t(i-1) > T(i) - T(i-1)$, i.e., if the arrival time difference is larger than the timestamp difference.

We define the (relative) inter-arrival time, $d(i)$ as

$$d(i) = t(i) - t(i-1) - (T(i) - T(i-1))$$

Since the time t_s to send a frame of size L over a path with a capacity of C is roughly

$$t_s = L/C$$

we can model the inter-arrival time as

$$d(i) = \frac{L(i) - L(i-1)}{C} + w(i) = dL(i)/C + w(i)$$

Here, $w(i)$ is a sample from a stochastic process W , which is a function of the capacity C , the current cross traffic $X(i)$, and the current send bit rate $R(i)$. We model W as a white Gaussian process. If we are over-using the channel we expect $w(i)$ to increase, and if a queue on the network path is being emptied, $w(i)$ will decrease; otherwise the mean of $w(i)$ will be zero.

Breaking out the mean $m(i)$ from $w(i)$ to make the process zero mean, we get

Equation 5

$$d(i) = dL(i)/C + m(i) + v(i)$$

This is our fundamental model, where we take into account that a large frame needs more time to traverse the link than a small frame, thus arriving with higher relative delay. The noise term represents network jitter and other delay effects not captured by the model.

When graphing the values for $d(i)$ versus $dL(i)$ on a scatterplot, we find that most samples cluster around the center, and the outliers are clustered along a line with average slope $1/C$ and zero offset.

For instance, when using a regular video codec, most frames are roughly the same size after encoding (the central "cloud"); the exceptions are I-frames (or key frames) which are typically much larger than the average causing positive outliers (the I-frame itself) and negative outliers (the frame after an I-frame) on the dL axis. Audio frames on the other hand often consist of single packets

of equal size, and an audio-only media stream would have its frames scattered at $dL = 0$.

3.3. Arrival-time filter

The parameters $d(i)$ and $dL(i)$ are readily available for each frame $i > 1$, and we want to estimate $C(i)$ and $m(i)$ and use those estimates to detect whether or not we are over-using the bandwidth currently available. These parameters are easily estimated by any adaptive filter - we are using the Kalman filter.

Let

$$\theta_{\text{bar}}(i) = [1/C(i) \quad m(i)]^T$$

and call it the state of time i . We model the state evolution from time i to time $i+1$ as

$$\theta_{\text{bar}}(i+1) = \theta_{\text{bar}}(i) + u_{\text{bar}}(i)$$

where $u_{\text{bar}}(i)$ is the zero mean white Gaussian process noise with covariance

Equation 7

$$Q(i) = E\{u_{\text{bar}}(i) u_{\text{bar}}(i)^T\}$$

Given equation 5 we get

Equation 8

$$d(i) = h_{\text{bar}}(i)^T \theta_{\text{bar}}(i) + v(i)$$

$$h_{\text{bar}}(i) = [dL(i) \quad 1]^T$$

where $v(i)$ is zero mean white Gaussian measurement noise with variance $\text{var}_v = \sigma(v, i)^2$

The Kalman filter recursively updates our estimate

$$\theta_{\text{hat}}(i) = [1/C_{\text{hat}}(i) \quad m_{\text{hat}}(i)]^T$$

as

$$\begin{aligned}
z(i) &= d(i) - h_bar(i)^T * \theta_hat(i-1) \\
\theta_hat(i) &= \theta_hat(i-1) + z(i) * k_bar(i) \\
k_bar(i) &= \frac{E(i-1) * h_bar(i)}{\text{var_v_hat} + h_bar(i)^T * E(i-1) * h_bar(i)} \\
E(i) &= (I - K_bar(i) * h_bar(i)^T) * E(i-1) + Q(i)
\end{aligned}$$

I is the 2-by-2 identity matrix.

The variance $\text{var_v} = \sigma(v,i)^2$ is estimated using an exponential averaging filter, modified for variable sampling rate

$$\begin{aligned}
\text{var_v_hat} &= \beta * \sigma(v,i-1)^2 + (1-\beta) * z(i)^2 \\
\beta &= (1-\alpha)^{(30/(1000 * f_max))}
\end{aligned}$$

where $f_max = \max \{1/(T(j) - T(j-1))\}$ for j in $i-K+1 \dots i$ is the highest rate at which frames have been captured by the camera the last K frames and α is a filter coefficient typically chosen as a number in the interval $[0.1, 0.001]$. Since our assumption that $v(i)$ should be zero mean WGN is less accurate in some cases, we have introduced an additional outlier filter around the updates of var_v_hat . If $z(i) > 3 \text{var_v_hat}$ the filter is updated with $3 \sqrt{\text{var_v_hat}}$ rather than $z(i)$. For instance $v(i)$ will not be white in situations where packets are sent at a higher rate than the channel capacity, in which case they will be queued behind each other. In a similar way, $Q(i)$ is chosen as a diagonal matrix with main diagonal elements given by

$$\text{diag}(Q(i)) = 30/(1000 * f_max) [10^{-10} \ 10^{-2}]^T$$

It is necessary to scale these filter parameters with the frame rate to make the detector respond as quickly at low frame rates as at high frame rates.

3.4. Over-use detector

The offset estimate $m(i)$ is compared with a threshold $\gamma_1(i)$. An estimate above the threshold is considered as an indication of over-use. Such an indication is not enough for the detector to signal over-use to the rate control subsystem. Not until over-use has been detected for at least γ_2 milliseconds and at least γ_3 frames, a definitive over-use will be signaled. However, if the offset estimate $m(i)$ was decreased in the last update, over-use will

not be signaled even if all the above conditions are met. Similarly, the opposite state, under-use, is detected when $m(i) < -\text{gamma_1}(i)$. If neither over-use nor under-use is detected, the detector will be in the normal state.

The threshold gamma_1 has a remarkable impact on the overall dynamics and performance of the algorithm. In particular, it has been shown that when using a static threshold gamma_1 , a flow controlled by the proposed algorithm can be starved by a concurrent TCP flow [Pv13]. This starvation can be avoided by increasing the threshold gamma_1 to a sufficiently large value.

The reason is that, by using a larger value of gamma_1 , a larger queuing delay can be tolerated, whereas with a small gamma_1 , the over-use detector quickly reacts to a small increase in the offset estimate $m(i)$ by generating an over-use signal that reduces A_r . Thus, it is necessary to dynamically tune the threshold gamma_1 to get good performance in the most common scenarios, such as when competing with loss-based flows.

For this reason, we propose to vary the threshold $\text{gamma_1}(i)$ according to the following dynamic equation:

$$\text{gamma_1}(i) = \text{gamma_1}(i-1) + (t(i)-t(i-1)) * K(i) * (|m(i)| - \text{gamma_1}(i-1))$$

with $K(i)=K_d$ if $|m(i)| < \text{gamma_1}(i-1)$ or $K(i)=K_u$ otherwise. The rationale is to increase $\text{gamma_1}(i)$ when $m(i)$ is outside of the range $[-\text{gamma_1}(i-1), \text{gamma_1}(i-1)]$, whereas, when the offset estimate m falls back into the range, gamma_1 is decreased. In this way when $m(i)$ quickly increases, for instance due to a TCP flow entering the same bottleneck, $\text{gamma_1}(i)$ increases and avoids the uncontrolled generation of over-use signals which may lead to starvation of the flow controlled by the proposed algorithm [Dec13].

On the other hand, when $m(i)$ falls back into the range $[-\text{gamma_1}(i-1), \text{gamma_1}(i-1)]$ the threshold $\text{gamma_1}(i)$ is decreased so that a lower queuing delay can be achieved.

We suggest to choose $K_u > K_d$ so that the rate at which gamma_1 is increased is higher than the rate at which it is decreased. With this setting it is possible to quickly increase the threshold in the case of a concurrent TCP flow and prevent starvation.

3.5. Rate control

The rate control at the receiving side is designed to increase the receive-side estimate of the available bandwidth $A_{\hat{}}$ as long as the detected state is normal. Doing that assures that we, sooner or later, will reach the available bandwidth of the channel and detect an over-use.

As soon as over-use has been detected the receive-side estimate of the available bandwidth is decreased. In this way we get a recursive and adaptive estimate of the available bandwidth.

In this document we make the assumption that the rate control subsystem is executed periodically and that this period is constant.

The rate control subsystem has 3 states: Increase, Decrease and Hold. "Increase" is the state when no congestion is detected; "Decrease" is the state where congestion is detected, and "Hold" is a state that waits until built-up queues have drained before going to "increase" state.

The state transitions (with blank fields meaning "remain in state") are:

State ---->	Hold	Increase	Decrease
Signal-----			
v			
Over-use	Decrease	Decrease	
Normal	Increase		Hold
Under-use		Hold	Hold

The subsystem starts in the increase state, where it will stay until over-use or under-use has been detected by the detector subsystem. On every update the receive-side estimate of the available bandwidth is increased with a factor which is a function of the global system response time and the estimated measurement noise variance $\text{var}_v_{\hat{}}$. The global system response time is the time from an increase that causes over-use until that over-use can be detected by the over-use detector. The variance $\text{var}_v_{\hat{}}$ affects how responsive the Kalman filter is, and is thus used as an indicator of the delay inflicted by the Kalman filter.

$$A_hat(i) = \eta * A_hat(i-1) + \frac{1.001+B}{1+e^{(b(d*RTT - (c1 * var_v_hat + c2)))}}$$

Here, B, b, d, c1 and c2 are design parameters.

Since the system depends on over-using the channel to verify the current available bandwidth estimate, we must make sure that our estimate doesn't diverge from the rate at which the sender is actually sending. Thus, if the sender is unable to produce a bit stream with the bit rate the receiver is asking for, the available bandwidth estimate must stay within a given bound. Therefore we introduce a threshold

$$A_hat(i) < 1.5 * R_hat(i)$$

where $R_hat(i)$ is the incoming bit rate measured over a T seconds window:

$$R_hat(i) = 1/T * \sum(L(j)) \text{ for } j \text{ from } 1 \text{ to } N(i)$$

$N(i)$ is the number of frames received the past T seconds and $L(j)$ is the payload size of frame j. Ideally T should be chosen to match the rate controller at the sender. A window between 0.5 and 1 second is recommended.

When an over-use is detected the system transitions to the decrease state, where the receive-side available bandwidth estimate is decreased to a factor times the currently incoming bit rate.

$$A_hat(i) = \alpha * R_hat(i)$$

α is typically chosen to be in the interval [0.8, 0.95].

When the detector signals under-use to the rate control subsystem, we know that queues in the network path are being emptied, indicating that our available bandwidth estimate is lower than the actual available bandwidth. Upon that signal the rate control subsystem will enter the hold state, where the receive-side available bandwidth estimate will be held constant while waiting for the queues to stabilize at a lower level - a way of keeping the delay as low as possible. This decrease of delay is wanted, and expected, immediately after the estimate has been reduced due to over-use, but can also happen if the cross traffic over some links is reduced. In either case we want to measure the highest incoming rate during the under-use interval:

$$R_{\max} = \max\{R_{\text{hat}}(i)\} \text{ for } i \text{ in } 1..K$$

where K is the number of frames of under-use before returning to the normal state. R_{\max} is a measure of the actual bandwidth available and is a good guess of what bit rate the sender should be able to transmit at. Therefore the receive-side available bandwidth estimate will be set to R_{\max} when we transition from the hold state to the increase state.

One design decision is when to send rate control messages. The time from a change in congestion to the sending of the feedback message is a limitation on how fast the sender can react. Sending too many messages giving no new information is a waste of bandwidth - but in the case of severe congestion, feedback messages can be lost, resulting in a failure to react in a timely manner.

The conclusion is that feedback messages should be sent on a "heartbeat" schedule, allowing the sender side control to react to missing feedback messages by reducing its send rate, but they should also be sent whenever the estimated bandwidth value has changed significantly, without waiting for the heartbeat time, up to some limiting upper bound on the send rate.

The minimum interval is named `t_min_fb_interval`.

The maximum interval is named `t_max_fb_interval`.

The permissible values of these intervals will be bounded by the RTP session's RTCP bandwidth and its `rtcp_frr` setting.

[TODO: Get some example values for these timers]

4. Sender side control

An additional congestion controller resides at the sending side. It bases its decisions on the round-trip time, packet loss and available bandwidth estimates transmitted from the receiving side.

The available bandwidth estimates produced by the receiving side are only reliable when the size of the queues along the channel are large enough. If the queues are very short, over-use will only be visible through packet losses, which aren't used by the receiving side algorithm.

This algorithm is run every time a receive report arrives at the sender, which will happen no more often than `t_min_fb_interval`, and no less often than `t_max_fb_interval`. If no receive report is

received within $2 \times t_{\text{max_fb_interval}}$ (indicating at least 2 lost feedback reports), the algorithm will take action as if all packets in the interval have been lost, resulting in a halving of the send rate.

- o If 2-10% of the packets have been lost since the previous report from the receiver, the sender available bandwidth estimate $As(i)$ (As denotes 'sender available bandwidth') will be kept unchanged.
- o If more than 10% of the packets have been lost a new estimate is calculated as $As(i) = As(i-1)(1-0.5p)$, where p is the loss ratio.
- o As long as less than 2% of the packets have been lost $As(i)$ will be increased as $As(i) = 1.05(As(i-1) + 1000)$

The new send-side estimate is limited by the TCP Friendly Rate Control formula [RFC3448] and the receive-side estimate of the available bandwidth $A(i)$:

$$As(i) \geq \frac{8s}{R\sqrt{2bp/3} + (t_{\text{RTO}}(3\sqrt{3bp/8}) * p * (1+32p^2))}$$

$$As(i) \leq A(i)$$

where b is the number of packets acknowledged by a single TCP acknowledgment (set to 1 per TFRC recommendations), t_{RTO} is the TCP retransmission timeout value in seconds (set to $4 \times R$) and s is the average packet size in bytes. R is the round-trip time in seconds.

(The multiplication by 8 comes because TFRC is computing bandwidth in bytes, while this document computes bandwidth in bits.)

In words: The sender-side estimate will never be larger than the receiver-side estimate, and will never be lower than the estimate from the TFRC formula.

We motivate the packet loss thresholds by noting that if the transmission channel has a small amount of packet loss due to over-use, that amount will soon increase if the sender does not adjust his bit rate. Therefore we will soon enough reach above the 10 % threshold and adjust $As(i)$. However if the packet loss rate does not increase, the losses are probably not related to self-induced channel over-use and therefore we should not react on them.

5. Interoperability Considerations

There are three scenarios of interest, and one included for reference

- o Both parties implement the algorithms described here
- o Sender implements the algorithm described in section Section 4, recipient does not implement Section 3
- o Recipient implements the algorithm in section Section 3, sender does not implement Section 4.

In the case where both parties implement the algorithms, we expect to see most of the congestion control response to slowly varying conditions happen by TMMBR/REMB messages from recipient to sender. At most times, the sender will send less than the congestion-inducing bandwidth limit C , and when he sends more, congestion will be detected before packets are lost.

If sudden changes happen, packets will be lost, and the sender side control will trigger, limiting traffic until the congestion becomes low enough that the system switches back to the receiver-controlled state.

In the case where sender only implements, we expect to see somewhat higher loss rates and delays, but the system will still be overall TCP friendly and self-adjusting; the governing term in the calculation will be the TFRC formula.

In the case where recipient implements this algorithm and sender does not, congestion will be avoided for slow changes as long as the sender understands and obeys TMMBR/REMB; there will be no backoff for packet-loss-inducing changes in capacity. Given that some kind of congestion control is mandatory for the sender according to the TMMBR spec, this case has to be reevaluated against the specific congestion control implemented by the sender.

6. Implementation Experience

This algorithm has been implemented in the open-source WebRTC project.

7. Further Work

This draft is offered as input to the congestion control discussion.

Work that can be done on this basis includes:

- o Consideration of timing info: It may be sensible to use the proposed TFRC RTP header extensions [I-D.gharai-avtcore-rtp-tfrc] to carry per-packet timing information, which would both give more data points and a timestamp applied closer to the network interface. This draft includes consideration of using the transmission time offset defined in [RFC5450]
- o Considerations of cross-channel calculation: If all packets in multiple streams follow the same path over the network, congestion or queuing information should be considered across all packets between two parties, not just per media stream. A feedback message (REMB) that may be suitable for such a purpose is given in [I-D.alvestrand-rmcat-remb].
- o Considerations of cross-channel balancing: The decision to slow down sending in a situation with multiple media streams should be taken across all media streams, not per stream.
- o Considerations of additional input: How and where packet loss detected at the recipient can be added to the algorithm.
- o Considerations of locus of control: Whether the sender or the recipient is in the best position to figure out which media streams it makes sense to slow down, and therefore whether one should use TMMBR to slow down one channel, signal an overall bandwidth change and let the sender make the decision, or signal the (possibly processed) delay info and let the sender run the algorithm.
- o Considerations of over-bandwidth estimation: Whether we can use the estimate of how much we're over bandwidth in section 3 to influence how much we reduce the bandwidth, rather than using a fixed factor.
- o Startup considerations. It's unreasonable to assume that just starting at full rate is always the best strategy.
- o Dealing with sender traffic shaping, which delays sending of packets. Using send-time timestamps rather than RTP timestamps may be useful here, but as long as the sender's traffic shaping does not spread out packets more than the bottleneck link, it should not matter.
- o Stability considerations. It is not clear how to show that the algorithm cannot provide an oscillating state, either alone or when competing with other algorithms / flows.

These are matters for further work; since some of them involve extensions that have not yet been standardized, this could take some time.

8. IANA Considerations

This document makes no request of IANA.

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

9. Security Considerations

An attacker with the ability to insert or remove messages on the connection will, of course, have the ability to mess up rate control, causing people to send either too fast or too slow, and causing congestion.

In this case, the control information is carried inside RTP, and can be protected against modification or message insertion using SRTP, just as for the media. Given that timestamps are carried in the RTP header, which is not encrypted, this is not protected against disclosure, but it seems hard to mount an attack based on timing information only.

10. Acknowledgements

Thanks to Randell Jesup, Magnus Westerlund, Varun Singh, Tim Panton, Soo-Hyun Choo, Jim Gettys, Ingemar Johansson, Michael Welzl and others for providing valuable feedback on earlier versions of this draft.

11. References

11.1. Normative References

- [I-D.alvestrand-rmcat-remb]
Alvestrand, H., "RTCP message for Receiver Estimated Maximum Bitrate", draft-alvestrand-rmcat-remb-03 (work in progress), October 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3448] Handley, M., Floyd, S., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 3448, January 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.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.
- [RFC5450] Singer, D. and H. Desineni, "Transmission Time Offsets in RTP Streams", RFC 5450, March 2009.

11.2. Informative References

- [I-D.gharai-avtcore-rtp-tfrc]
Gharai, L. and C. Perkins, "RTP with TCP Friendly Rate Control", draft-gharai-avtcore-rtp-tfrc-01 (work in progress), September 2011.
- [Pv13] De Cicco, L., Carlucci, G., and S. Mascolo, "Understanding the Dynamic Behaviour of the Google Congestion Control", Packet Video Workshop , December 2013.
- [RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, September 2000.

Appendix A. Change log

A.1. Version -00 to -01

- o Added change log
- o Added appendix outlining new extensions
- o Added a section on when to send feedback to the end of section 3.3 "Rate control", and defined min/max FB intervals.
- o Added size of over-bandwidth estimate usage to "further work" section.
- o Added startup considerations to "further work" section.
- o Added sender-delay considerations to "further work" section.
- o Filled in acknowledgments section from mailing list discussion.

A.2. Version -01 to -02

- o Defined the term "frame", incorporating the transmission time offset into its definition, and removed references to "video frame".
- o Referred to "m(i)" from the text to make the derivation clearer.
- o Made it clearer that we modify our estimates of available bandwidth, and not the true available bandwidth.
- o Removed the appendixes outlining new extensions, added pointers to REMB draft and RFC 5450.

A.3. Version -02 to -03

- o Added a section on how to process multiple streams in a single estimator using RTP timestamps to NTP time conversion.
- o Stated in introduction that the draft is aimed at the RMCAT working group.

A.4. rtcweb-03 to rmcat-00

Renamed draft to link the draft name to the RMCAT WG.

A.5. rmcat -00 to -01

Spellcheck. Otherwise no changes, this is a "keepalive" release.

A.6. rmcat -01 to -02

- o Added Luca De Cicco and Saverio Mascolo as authors.
- o Extended the "Over-use detector" section with new technical details on how to dynamically tune the offset `gamma_1` for improved fairness properties.
- o Added a reference to a paper analyzing the behavior of the proposed algorithm.

Authors' Addresses

Henrik Lundin
Google
Kungsbron 2
Stockholm 11122
Sweden

Stefan Holmer
Google
Kungsbron 2
Stockholm 11122
Sweden

Email: holmer@google.com

Luca De Cicco
Politecnico di Bari
Via Orabona, 4
Bari 70125
Italy

Email: l.decicco@poliba.it

Saverio Mascolo
Politecnico di Bari
Via Orabona, 4
Bari 70125
Italy

Email: mascolo@poliba.it

Harald Alvestrand (editor)
Google
Kungsbron 2
Stockholm 11122
Sweden

Email: harald@alvestrand.no

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

R. Jesup
Mozilla
February 14, 2014

Congestion Control Requirements For RMCAT
draft-ietf-rmcat-cc-requirements-02

Abstract

Congestion control is needed for all data transported across the Internet, in order to promote fair usage and prevent congestion collapse. The requirements for interactive, point-to-point real time multimedia, which needs low-delay, semi-reliable data delivery, are different from the requirements for bulk transfer like FTP or bursty transfers like Web pages.

This document attempts to describe a set of requirements that can be used to evaluate other congestion control mechanisms in order to figure out their fitness for this purpose, and in particular to provide a set of possible requirements for proposals coming out of the RMCAT Working Group.

This document is derived from draft-jesup-rtp-congestion-reqs [I-D.jesup-rtp-congestion-reqs].

Requirements Language

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 [RFC2119].

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 August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements	3
3. IANA Considerations	7
4. Security Considerations	7
5. Acknowledgements	8
6. References	8
6.1. Normative References	8
6.2. Informative References	8
Author's Address	9

1. Introduction

The traditional TCP congestion control requirements were developed in order to promote efficient use of the Internet for reliable bulk transfer of non-time-critical data, such as transfer of large files. They have also been used successfully to govern the reliable transfer of smaller chunks of data in as short a time as possible, such as when fetching Web pages.

These algorithms have also been used for transfer of media streams that are viewed in a non-interactive manner, such as "streaming" video, where having the data ready when the viewer wants it is important, but the exact timing of the delivery is not.

When doing real time interactive media, the requirements are different; one needs to provide the data continuously, within a very limited time window (no more than 100s of milliseconds end-to-end delay), the sources of data may be able to adapt the amount of data that needs sending within fairly wide margins, and may tolerate some

amount of packet loss, but since the data is generated in real time, sending "future" data is impossible, and since it's consumed in real time, data delivered late is useless.

While the requirements for RMCAT differ from the requirements for the other flow types, these other flow types will be present in the network. The RMCAT congestion control algorithm must work properly when these other flow types are present as cross traffic on the network.

One particular protocol portfolio being developed for this use case is WebRTC [I-D.ietf-rtcweb-overview], where one envisions sending multiple RTP-based flows between two peers, in conjunction with data flows, all at the same time, without having special arrangements with the intervening service providers.

Given that this use case is the focus of this document, use cases involving noninteractive media such as YouTube-like video streaming, and use cases using multicast/broadcast-type technologies, are out of scope.

The terminology defined in [I-D.ietf-rtcweb-overview] is used in this memo.

2. Requirements

1. The congestion control algorithm must attempt to provide as-low-as-possible-delay transit for real-time traffic while still providing a useful amount of bandwidth, even when faced with intermediate bottlenecks and competing flows. There may be lower limits on the amount of bandwidth that is useful, but this is largely application-specific and the application may be able to modify or remove flows in order allow some useful flows to get enough bandwidth. (Example: not enough bandwidth for low-latency video+audio, but enough for audio-only.)
 - A. It should also handle routing changes and interface changes (WiFi to 3G data, etc) which may radically change the bandwidth available, and react quickly, especially if there is a reduction in available bandwidth.
 - B. The offered load may be less than the available bandwidth at any given moment, and may vary dramatically over time, including dropping to no load and then resuming a high load, such as in a mute operation. The reaction time between a change in the bandwidth available from the algorithm and a change in the offered load is variable, and may be different when increasing versus decreasing.

- C. The algorithm must not overreact to short-term bursts (such as web-browsing) which can quickly saturate a local-bottleneck router or link, but also clear quickly, and should recover quickly when the burst ends. This is inherently at odds with the need to react quickly-enough to avoid queue buildup.
 - D. Similarly periodic bursty flows such as DASH or proprietary media streaming algorithms may compete in bursts with the algorithm, and may not be adaptive within a burst. They are often layered on top of TCP. The algorithm must avoid too much delay buildup during those bursts, and quickly recover. Note that this traffic may be on an access link, or may cause a shift in the location of the bottleneck for the duration of the burst.
2. The algorithm must be fair to other flows, both realtime flows (such as other instances of itself), and TCP flows, both long-lived and bursts such as the traffic generated by a typical web browsing session. Note that 'fair' is a rather hard-to-define term.
- A. Existing flows at a bottleneck must also be fair to new flows to that bottleneck, and must allow new flows to ramp up to a useful share of the bottleneck bandwidth quickly.
3. The algorithm should where possible merge information across multiple RTP streams between the same endpoints, whether or not they're multiplexed on the same ports, in order to allow congestion control of the set of streams together instead of as multiple independent streams. This allows better overall bandwidth management, faster response to changing conditions, and fairer sharing of bandwidth with other network users. Alternatively, it should work with an external bandwidth control framework to coordinate bandwidth usage across a bottleneck, such as draft-welzl-rmcat-coupled-cc [I-D.welzl-rmcat-coupled-cc].
- A. If possible, it should also share information and adaptation with other non-RTP flows between the same endpoints, such as a WebRTC data channel
 - B. The most correlated bandwidth usage would be with other flows on the same 5-tuple, but there may be use in coordinating measurement and control of the local link(s).

- C. Use of information about previous flows, especially on the same 5-tuple, may be useful input to the algorithm, especially to startup performance of a new flow.
- 4. The algorithm should not require any special support from network elements (ECN, etc). As much as possible, it should leverage available information about the incoming flow to provide feedback to the sender. Examples of this information are the ECN, packet arrival times, acknowledgments and feedback, packet timestamps, and packet losses; all of these can provide information about the state of the path and any bottlenecks.
 - A. Extra information could be added to the packets to provide more detailed information on actual send times (as opposed to sampling times), but should not be required.
 - B. When additional input signals such as ECN are available, they should be utilized if possible.
- 5. Since the assumption here is a set of RTP streams, the backchannel typically should be done via RTCP; one alternative would be to include it instead in a reverse RTP channel using header extensions.
 - A. In order to react sufficiently quickly when using RTCP for a backchannel, an RTP profile such as AVPF/SAVPF that allows sufficiently frequent feedback [RFC4585] MUST be used.
 - B. Note that in some cases, backchannel messages may be delayed until the RTCP channel can be allocated enough bandwidth, even under AVPF rules. This may also imply negotiating a higher maximum percentage for RTCP data or allowing RMCAT solutions to violate or modify the rules specified for AVPF.
 - C. Bandwidth for the feedback messages should be minimized (such as via RFC 5506 [RFC5506] to allow RTCP without SR/RR)
 - D. Header extensions would avoid the RTCP timing rules issues, and allow the application to allocate bandwidth as needed for the congestion algorithm.
 - E. Backchannel data should be minimized to avoid taking too much reverse-channel bandwidth (since this will often be used in a bidirectional set of flows). In areas of stability, backchannel data may be sent more infrequently so long as algorithm stability and fairness are maintained. When the channel is unstable or has not yet reached equilibrium after a change, backchannel feedback may be more

frequent and use more reverse-channel bandwidth. This is an area with considerable flexibility of design, and different approaches to backchannel messages and frequency are expected to be evaluated.

6. Flows managed by this algorithm and flows competed against at a bottleneck may have different DSCP markings depending on the type of traffic. A particular bottleneck or section of the network path may or may not honor these markings.
 - A. In WebRTC, a division of packets into 4 classes is envisioned in order of priority: faster-than-audio, audio, video, best-effort, and bulk-transfer. Typically the flows managed by this algorithm would be audio or video in that hierarchy, and feedback flows would be faster-than-audio.
7. The algorithm should sense the unexpected lack of backchannel information as a possible indication of a channel overuse problem and react accordingly to avoid burst events causing a congestion collapse.
8. The algorithm should not starve competing TCP flows, and should as best as possible avoid starvation by TCP flows.
 - A. An algorithm may be more successful at avoiding starvation from short-lived TCP long-lived/saturating TCP flows.
 - B. In order to avoid starvation other goals may need to be compromised (such as delay).
9. The algorithm should be stable and low-delay when faced with active queue management (AQM) algorithms. Also note that these algorithms may apply across multiple queues in the bottleneck, or to a single queue
10. The algorithm should quickly adapt to initial network conditions at the start of a flow. This should occur both if the initial bandwidth is above or below the bottleneck bandwidth.
 - A. The startup adaptation may be faster than adaptation later in a flow. It should allow for both slow-start operation (adapt up) and history-based startup (start at a point expected to be at or below channel bandwidth from historical information, which may need to adapt down quickly if the initial guess is wrong). Starting too low and/or adapting up too slowly can cause a critical point in a personal communication to be poor ("Hello!"). Starting over-

bandwidth causes other problems for user experience, so there's a tension here.

- B. Alternative methods to help startup like probing during setup with dummy data may be useful in some applications; in some cases there will be a considerable gap in time between flow creation and the initial flow of data.
 - C. A flow may need to change adaptation rates due to network conditions or changes in the provided flows (such as unmuting or sending data after a gap).
- 11. It should be evaluated in how it works both with backbone-router bottlenecks, (asymmetric) local-loop bottlenecks, and local-lan (WiFi/etc) bottlenecks, and in competition with varying numbers and types of streams (TCP, TCP variants in use, LEDBAT [I-D.ietf-ledbat-congestion], inflexible VoIP UDP flows).
 - 12. It should be stable if the RTP streams are halted or discontinuous (VAD/DTX).
 - A. After a resumption of RTP data it may adapt more quickly (similar to the start of a flow), and previous bandwidth estimates may need to be aged or thrown away.

3. IANA Considerations

This document makes no request of IANA.

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

4. Security Considerations

An attacker with the ability to delete, delay or insert messages in the flow can fake congestion signals, unless they are passed on a tamper-proof path. Since some possible algorithms depend on the timing of packet arrival, even a traditional protected channel does not fully mitigate such attacks.

An attack that reduces bandwidth is not necessarily significant, since an on-path attacker could break the connection by discarding all packets. Attacks that increase the perceived available bandwidth are conceivable, and need to be evaluated.

Algorithm designers SHOULD consider the possibility of malicious on-path attackers.

5. Acknowledgements

This document is the result of discussions in various fora of the WebRTC effort, in particular on the `rtcp-congestion@alvestrand.no` mailing list. Many people contributed their thoughts to this.

6. References

6.1. Normative References

- [I-D.ietf-rtcweb-overview]
Alvestrand, H., "Overview: Real Time Protocols for Brower-based Applications", draft-ietf-rtcweb-overview-08 (work in progress), September 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [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.

6.2. Informative References

- [I-D.ietf-ledbat-congestion]
Shalunov, S., Hazel, G., Iyengar, J., and M. Kuehlewind, "Low Extra Delay Background Transport (LEDBAT)", draft-ietf-ledbat-congestion-10 (work in progress), September 2012.
- [I-D.jesup-rtp-congestion-reqs]
Jesup, R. and H. Alvestrand, "Congestion Control Requirements For Real Time Media", draft-jesup-rtp-congestion-reqs-00 (work in progress), March 2012.
- [I-D.welzl-rmcatt-coupled-cc]
Welzl, M., Islam, S., and S. Gjessing, "Coupled congestion control for RTP media", draft-welzl-rmcatt-coupled-cc-02 (work in progress), October 2013.
- [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.

Author's Address

Randell Jesup
Mozilla
USA

Email: randell-ietf@jesup.org

RMCAT WG
Internet-Draft
Intended status: Informational
Expires: August 04, 2014

V. Singh
J. Ott
Aalto University
January 31, 2014

Evaluating Congestion Control for Interactive Real-time Media
draft-ietf-rmcat-eval-criteria-00

Abstract

The Real-time Transport Protocol (RTP) is used to transmit media in telephony and video conferencing applications. This document describes the guidelines to evaluate new congestion control algorithms for interactive point-to-point real-time media.

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 August 04, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Metrics	3
3.1. RTP Log Format	5
4. Guidelines	5
4.1. Avoiding Congestion Collapse	5
4.2. Stability	5
4.3. Media Traffic	6
4.4. Start-up Behaviour	6
4.5. Diverse Environments	6
4.6. Varying Path Characteristics	7
4.7. Reacting to Transient Events or Interruptions	7
4.8. Fairness With Similar Cross-Traffic	7
4.9. Impact on Cross-Traffic	7
4.10. Extensions to RTP/RTCP	8
5. Minimum Requirements for Evaluation	8
6. Evaluation Parameters	8
6.1. Bottleneck Traffic Flows	8
6.2. Access Links	9
6.3. Example Bottleneck Link Parameters	9
6.4. DropTail Router Queue Parameters	10
6.5. Media Flow Parameters	11
6.6. Cross-traffic Parameters	11
7. Status of Proposals	11
8. Security Considerations	12
9. IANA Considerations	12
10. Contributors	12
11. Acknowledgements	12
12. References	12
12.1. Normative References	12
12.2. Informative References	13
Appendix A. Application Trade-off	14
A.1. Measuring Quality	14
Appendix B. Proposal to evaluate Self-fairness of RMCAT congestion control algorithm	14
B.1. Evaluation Parameters	15
B.1.1. Media Traffic Generator	15
B.1.2. Bottleneck Link Bandwidth	16
B.1.3. Bottleneck Link Queue Type and Length	16
B.1.4. RMCAT flows and delay legs	16
B.1.5. Impairment Generator	17
B.2. Proposed Passing Criteria	17
B.3. Extensibility of the Experiment	17
Appendix C. Change Log	18
C.1. Changes in draft-ietf-rmcat-eval-criteria-00	18
C.2. Changes in draft-singh-rmcat-cc-eval-04	18

C.3. Changes in draft-singh-rmcat-cc-eval-03	18
C.4. Changes in draft-singh-rmcat-cc-eval-02	19
C.5. Changes in draft-singh-rmcat-cc-eval-01	19
Authors' Addresses	19

1. Introduction

This memo describes the guidelines to help with evaluating new congestion control algorithms for interactive point-to-point real time media. The requirements for the congestion control algorithm are outlined in [I-D.ietf-rmcat-cc-requirements]). This document builds upon previous work at the IETF: Specifying New Congestion Control Algorithms [RFC5033] and Metrics for the Evaluation of Congestion Control Algorithms [RFC5166].

The guidelines proposed in the document are intended to help prevent a congestion collapse, promote fair capacity usage and optimize the media flow's throughput. Furthermore, the proposed algorithms are expected to operate within the envelope of the circuit breakers defined in [I-D.ietf-avtcore-rtp-circuit-breakers].

This document only provides broad-level criteria for evaluating a new congestion control algorithm and the working group should expect a thorough scientific study to make its decision. The results of the evaluation are not expected to be included within the internet-draft but should be cited in the document.

2. Terminology

The terminology defined in RTP [RFC3550], RTP Profile for Audio and Video Conferences with Minimal Control [RFC3551], RTCP Extended Report (XR) [RFC3611], Extended RTP Profile for RTCP-based Feedback (RTP/AVPF) [RFC4585] and Support for Reduced-Size RTCP [RFC5506] apply.

3. Metrics

[RFC5166] describes the basic metrics for congestion control. Metrics that are of interest for interactive multimedia are:

- o Throughput.
- o Minimizing oscillations in the transmission rate (stability) when the end-to-end capacity varies slowly.
- o Delay.
- o Reactivity to transient events.

- o Packet losses and discards.
- o Section 2.1 of [RFC5166] discusses the tradeoff between throughput, delay and loss.

Each experiment is expected to log every incoming and outgoing packet (the RTP logging format is described in Section 3.1). The logging can be done inside the application or at the endpoints using pcap (packet capture, e.g., tcpdump, wireshark). The following are calculated based on the information in the packet logs:

1. Sending rate, Receiver rate, Goodput
2. Packet delay
3. Packet loss
4. If using, retransmission or FEC: residual loss
5. Packets discarded from the playout or de-jitter buffer

[Open issue (1): The "unfairness" test is (measured at 1s intervals):

1. Do not trigger the circuit breaker.
2. Over 3 times or less than 1/3 times the throughput for an RMCAT media stream compared to identical RMCAT streams competing on a bottleneck, for a case when the competing streams have similar RTTs.
3. Over 3 times delay compared to RTT measurements performed before starting the RMCAT flow or for the case when competing with identical RMCAT streams having similar RTTs.

]

[Open issue (2): Possibly using Jain-fairness index.]

Convergence time: the time taken to reach a stable rate at startup, after the available link capacity changes, or when new flows get added to the bottleneck link.

Bandwidth Utilization, defined as ratio of the instantaneous sending rate to the instantaneous bottleneck capacity. This metric is useful when an RMCAT flow is by itself or competing with similar cross-traffic.

From the logs the statistical measures (min, max, mean, standard deviation and variance) for the whole duration or any specific part of the session can be calculated. Also the metrics (sending rate, receiver rate, goodput, latency) can be visualized in graphs as variation over time, the measurements in the plot are at 1 second intervals. Additionally, from the logs it is possible to plot the histogram or CDF of packet delay.

3.1. RTP Log Format

The log file is tab or comma separated containing the following details:

```
Send or receive timestamp (unix)
RTP payload type
SSRC
RTP sequence no
RTP timestamp
marker bit
payload size
```

If the congestion control implements, retransmissions or FEC, the evaluation should report both packet loss (before applying error-resilience) and residual packet loss (after applying error-resilience).

4. Guidelines

A congestion control algorithm should be tested in simulation or a testbed environment, and the experiments should be repeated multiple times to infer statistical significance. The following guidelines are considered for evaluation:

4.1. Avoiding Congestion Collapse

The congestion control algorithm is expected to take an action, such as reducing the sending rate, when it detects congestion. Typically, it should intervene before the circuit breaker [I-D.ietf-avtcore-rtp-circuit-breakers] is engaged.

Does the congestion control propose any changes to (or diverge from) the circuit breaker conditions defined in [I-D.ietf-avtcore-rtp-circuit-breakers].

4.2. Stability

The congestion control should be assessed for its stability when the path characteristics do not change over time. Changing the media encoding rate estimate too often or by too much may adversely affect the application layer performance.

4.3. Media Traffic

The congestion control algorithm should be assessed with different types of media behavior, i.e., the media should contain idle and data-limited periods. For example, periods of silence for audio, varying amount of motion for video, or bursty nature of I-frames.

The evaluation may be done in two stages. In the first stage, the endpoint generates traffic at the rate calculated by the congestion controller. In the second stage, real codecs or models of video codecs are used to mimic application-limited data periods and varying video frame sizes.

4.4. Start-up Behaviour

The congestion control algorithm should be assessed with different start-rates. The main reason is to observe the behavior of the congestion control in different evaluation scenarios, such as when competing with varying amount of cross-traffic or how quickly does the congestion control algorithm achieve a stable sending rate.

[Editor's note: requires a robust definition for unfriendliness and convergence time.]

4.5. Diverse Environments

The congestion control algorithm should be assessed in heterogeneous environments, containing both wired and wireless paths. Examples of wireless access technologies are: 802.11, GPRS, HSPA, or LTE. One of the main challenges of the wireless environments for the congestion control algorithm is to distinguish between congestion induced loss and transmission (bit-error corruption) loss. Congestion control algorithms may incorrectly identify transmission loss as congestion loss and reduce the media encoding rate by too much, which may cause oscillatory behavior and deteriorate the users' quality of experience. Furthermore, packet loss may induce additional delay in networks with wireless paths due to link-layer retransmissions.

4.6. Varying Path Characteristics

The congestion control algorithm should be evaluated for a range of path characteristics such as, different end-to-end capacity and latency, varying amount of cross traffic on a bottleneck link and a router's queue length. For the moment, only DropTail queues are used. However, if new Active Queue Management (AQM) schemes become available, the performance of the congestion control algorithm should be again evaluated.

In an experiment, if the media only flows in a single direction, the feedback path should also be tested with varying amounts of impairments.

The main motivation for the previous and current criteria is to identify situations in which the proposed congestion control is less performant.

4.7. Reacting to Transient Events or Interruptions

The congestion control algorithm should be able to handle changes in end-to-end capacity and latency. Latency may change due to route updates, link failures, handovers etc. In mobile environment the end-to-end capacity may vary due to the interference, fading, handovers, etc. In wired networks the end-to-end capacity may vary due to changes in resource reservation.

4.8. Fairness With Similar Cross-Traffic

The congestion control algorithm should be evaluated when competing with other RTP flows using the same or another candidate congestion control algorithm. The proposal should highlight the bottleneck capacity share of each RTP flow.

[Editor's note: If we define Unfriendliness then that criteria should be applied here.]

4.9. Impact on Cross-Traffic

The congestion control algorithm should be evaluated when competing with standard TCP. Short TCP flows may be considered as transient events and the RTP flow may give way to the short TCP flow to complete quickly. However, long-lived TCP flows may starve out the RTP flow depending on router queue length.

The proposal should also measure the impact on varied number of cross-traffic sources, i.e., few and many competing flows, or mixing various amounts of TCP and similar cross-traffic.

4.10. Extensions to RTP/RTCP

The congestion control algorithm should indicate if any protocol extensions are required to implement it and should carefully describe the impact of the extension.

5. Minimum Requirements for Evaluation

[Editor's Note: If needed, a minimum evaluation criteria can be based on the above guidelines or defined tests/scenarios.]

6. Evaluation Parameters

An evaluation scenario is created from a list of network, link and flow characteristics. The example parameters discussed in the following subsections are meant to aid in creating evaluation scenarios and do not describe an evaluation scenario. The scenario discussed in Appendix B takes into account all these parameters.

6.1. Bottleneck Traffic Flows

The network scenario describes the types of flows sharing the common bottleneck with a single RMCAT flow, they are:

1. A single RMCAT flow by itself.
2. Competing with similar RMCAT flows. These competing flows may use the same algorithm or another candidate RMCAT algorithm.
3. Compete with long-lived TCP.
4. Compete with bursty TCP.
5. Compete with LEDBAT flows.
6. Compete with unresponsive interactive media flows (i.e., not only CBR).

Figure 1 shows an example evaluation topology, where S1..Sn are traffic sources, these sources are either RMCAT or a mixture of traffic flows listed above. R1..Rn are the corresponding receivers. A and B are routers that can be configured to introduce impairments. Access links are in between the sender/receiver and the router, while the bottleneck link is between the Routers A and B.

```

+---+ Access                               Access +---+
|S1 |===== \                             / =====|R1 |
+---+ link  \\\                          // link  +---+

```

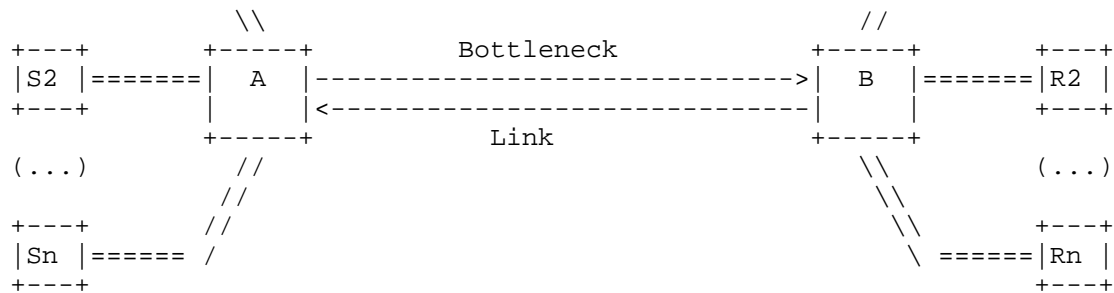


Figure 1: Simple Topology

[Open Issue: Discuss more complex topologies]

6.2. Access Links

The media senders and receivers are typically connected to the bottleneck link, common access links are:

1. Ethernet (LAN)
2. Wireless LAN (WLAN)
3. 3G/LTE

[Open issue: point to a reference containing parameters or traces to model WLAN and 3G/LTE.]

A real-world network typically consists of a mixture of links, the most important aspect is to identify the location of the bottleneck link. The bottleneck link can move from one node to another depending on the amount of cross-traffic or due to the varying link capacity. The design of the experiments should take this into account. In the simplest case the access link may not be the bottleneck link but an intermediate node.

6.3. Example Bottleneck Link Parameters

The bottleneck link carries multiple flows, these flows may be other RMCAT flows or other types of cross-traffic. The experiments should dimension the bottleneck link based on the number of flows and the expected behavior. For example, if 5 media flows are expected to share the bottleneck link equally, the bottleneck link is set to 5 times the desired transmission rate.

If the experiment carries only media in one direction, then the upstream (sender to receiver) bottleneck link carries media packets

while the downstream (receiver to sender) bottleneck carries the feedback packets. The bottleneck link parameters discussed in this section apply only to a single direction, hence the bottleneck link in the reverse direction can choose the same or have different parameters.

The link latency corresponds to the propagation delay of the link, i.e., the time it takes for a packet to traverse the bottleneck link, it does not include queuing delay. In an experiment with several links the experiment should describe if the links add latency or not. It is possible for experiments to have multiple hops with different link latencies. Experiments are expected to verify that the congestion control is able to work in challenging situations, for example over trans-continental and/or satellite links. The experiment should pick link latency values from the following:

1. Very low latency: 0-1ms
2. Low latency: 50ms
3. High latency: 150ms
4. Extreme latency: 300ms

Similarly, to model lossy links, the experiments can choose one of the following loss rates, the fractional loss is the ratio of packets lost and packets sent.

1. no loss: 0%
2. 1%
3. 5%
4. 10%
5. 20%

These fractional losses can be generated using traces, Gilbert-Elliot model, randomly (uncorrelated) loss.

6.4. DropTail Router Queue Parameters

The router queue length is measured as the time taken to drain the FIFO queue, they are:

1. QoS-aware (or short): 70ms

2. Nominal: 500ms

3. Buffer-bloated: 2000ms

However, the size of the queue is typically measured in bytes or packets and to convert the queue length measured in seconds to queue length in bytes:

$$\text{QueueSize (in bytes)} = \text{QueueSize (in sec)} \times \text{Throughput (in bps)} / 8$$

6.5. Media Flow Parameters

The media sources can be modeled in two ways. In the first, the sources always have data to send, i.e., have no data limited intervals and are able to generate the media rate requested by the RMCAT congestion control algorithm. In the second, the traffic generator models the behavior of a media codec, mainly the burstiness (time-varying data produced by a video GOP).

At the beginning of the session, the media sources are configured to start at a given start rate, they are:

1. 200 kbps
2. 800 kbps
3. 1300 kbps
4. 4000 kbps

6.6. Cross-traffic Parameters

Long-lived TCP flows will download data throughout the session and are expected to have infinite amount of data to send or receive.

[Open issue: short-lived/bursty TCP cross-traffic parameters are still TBD.

7. Status of Proposals

Congestion control algorithms are expected to be published as "Experimental" documents until they are shown to be safe to deploy. An algorithm published as a draft should be experimented in simulation, or a controlled environment (testbed) to show its applicability. Every congestion control algorithm should include a note describing the environments in which the algorithm is tested and safe to deploy. It is possible that an algorithm is not recommended for certain environments or perform sub-optimally for the user.

[Editor's Note: Should there be a distinction between "Informational" and "Experimental" drafts for congestion control algorithms in RMCAT. [RFC5033] describes Informational proposals as algorithms that are not safe for deployment but are proposals to experiment with in simulation/testbeds. While Experimental algorithms are ones that are deemed safe in some environments but require a more thorough evaluation (from the community).]

8. Security Considerations

Security issues have not been discussed in this memo.

9. IANA Considerations

There are no IANA impacts in this memo.

10. Contributors

The content and concepts within this document are a product of the discussion carried out in the Design Team.

Michael Ramalho provided the text for the scenario discussed in Appendix B.

11. Acknowledgements

Much of this document is derived from previous work on congestion control at the IETF.

The authors would like to thank Harald Alvestrand, Luca De Cicco, Wesley Eddy, Lars Eggert, Kevin Gross, Vinayak Hegde, Stefan Holmer, Randell Jesup, Piers O'Hanlon, Colin Perkins, Michael Ramalho, Zaheduzzaman Sarker, Timothy B. Terriberry, Michael Welzl, and Mo Zanaty for providing valuable feedback on earlier versions of this draft. Additionally, also thank the participants of the design team for their comments and discussion related to the evaluation criteria.

12. References

12.1. Normative References

- [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.

- [RFC3611] Friedman, T., Caceres, R., and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.
- [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.
- [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.
- [I-D.ietf-rmcat-cc-requirements]
Jesup, R., "Congestion Control Requirements For RMCAT", draft-ietf-rmcat-cc-requirements-00 (work in progress), July 2013.
- [I-D.ietf-avtcore-rtp-circuit-breakers]
Perkins, C. and V. Singh, "RTP Congestion Control: Circuit Breakers for Unicast Sessions", draft-ietf-avtcore-rtp-circuit-breakers-01 (work in progress), October 2012.

12.2. Informative References

- [RFC5033] Floyd, S. and M. Allman, "Specifying New Congestion Control Algorithms", BCP 133, RFC 5033, August 2007.
- [RFC5166] Floyd, S., "Metrics for the Evaluation of Congestion Control Mechanisms", RFC 5166, March 2008.
- [RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.
- [SA4-EVAL]
R1-081955, 3GPP., "LTE Link Level Throughput Data for SA4 Evaluation Framework", 3GPP R1-081955, 5 2008.
- [SA4-LR]
S4-050560, 3GPP., "Error Patterns for MBMS Streaming over UTRAN and GERAN", 3GPP S4-050560, 5 2008.
- [TCP-eval-suite]
Lachlan, A., Marcondes, C., Floyd, S., Dunn, L., Guillier, R., Gang, W., Eggert, L., Ha, S., and I. Rhee, "Towards a Common TCP Evaluation Suite", Proc. PFLDnet. 2008, August 2008.

Appendix A. Application Trade-off

Application trade-off is yet to be defined. see RMCAT requirements [I-D.ietf-rmcat-cc-requirements] document. Perhaps each experiment should define the application's expectation or trade-off.

A.1. Measuring Quality

No quality metric is defined for performance evaluation, it is currently an open issue. However, there is consensus that congestion control algorithm should be able to show that it is useful for interactive video by performing analysis using a real codec and video sequences.

Appendix B. Proposal to evaluate Self-fairness of RMCAT congestion control algorithm

The goal of the experiment discussed in this section is to initially take out as many unknowns from the scenario. Later experiments can define more complex environments, topologies and media behavior. This experiment evaluates the performance of the RMCAT sender competing with other similar RMCAT flows (running the same algorithm or other RMCAT proposals) on the bottleneck link. There are up to 20 RMCAT flows competing for capacity, but the media only flows in one direction, from senders (S1..S20) to receivers (R1..R20) and the feedback packets flow in the reverse direction.

Figure 2 shows the experiment setup and it has subtle differences compared to the simple topology in Figure 1. Groups of 10 receivers are connected to the bottleneck link through two different routers (Router C and D). The rationale for adding these additional routers is to create two delay legs, i.e., two groups of endpoints with different network latencies and measure the performance of the RMCAT congestion control algorithm. If fewer than 10 sources are initialized, all traffic flows experience the same delay because they share the same delay leg.

Router A has a single forward direction bottleneck link (i.e., the bottleneck capacity and delay constraints applies only to the media packets going from the sender to the receiver, the feedback packets are unaffected). Hence, the Round-Trip Time (RTT) is primarily composed of the bottleneck queue delay and any forward path (propagation) latency. The main reason for not applying any constraints on the return path is to provide the best-case performance scenario for the congestion control algorithm. In later experiments, it is possible to add similar capacity and delay constraints on the return path.

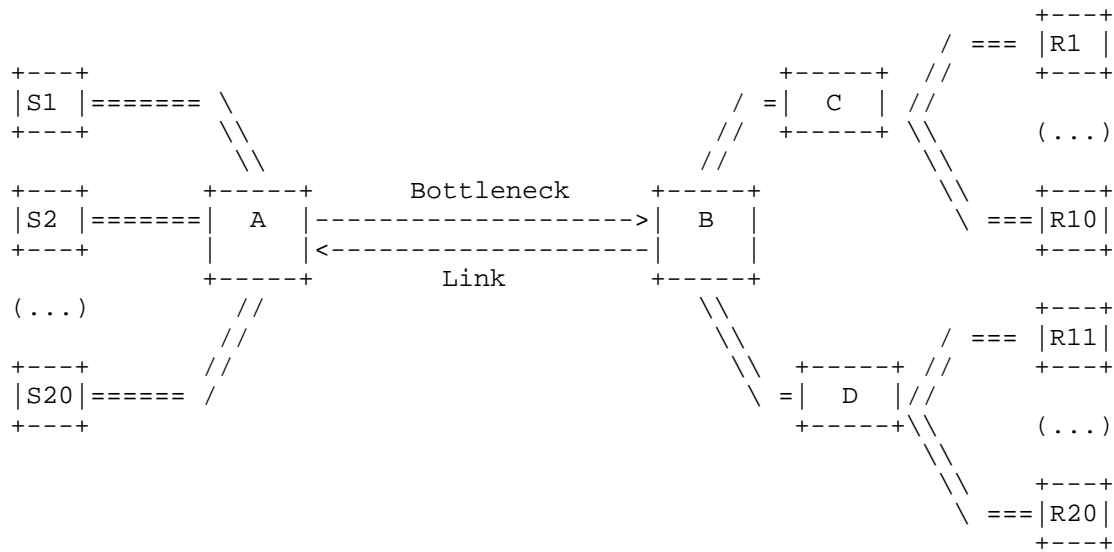


Figure 2: Self-fairness Evaluation Setup

Loss impairments are applied at Router C and Router D, but only to the feedback flows. If the losses are set to 0%, it represents a case where the return path is over-provisioned for all traffic. In later experiments the loss impairments can be added to the media path as well.

The media sources are configured to send infinite amount of data, i.e., the sources always have data to send and have no data limited intervals. Additionally, the media sources are always successful in generating the media rate requested by the RMCAT congestion control algorithm. In this experiment, we avoid the potentially complicated scenario of using media traffic generators that try to model the behavior of media codecs (mainly the burstiness).

B.1. Evaluation Parameters

B.1.1. Media Traffic Generator

The media source always generates at the rate requested by the congestion control and has infinite data to send. Furthermore, the media packet generator is subject to the following constraints:

1. It MUST emit a packet at least once per 100 ms time interval.
2. For low media rate source: when generating data at a rate less than a maximum length MTU every 100 ms would allow (e.g., 120

kbps = 1500 bytes/packet * 10 packets/sec * 8 bits/byte), the RMCAT source must modulate the packet size (RTP payload size) of RTP packets that are sent every 100 ms to attain the desired rate.

3. For high media rate sources: when generating data at a rate greater than a maximum length MTU every 100 ms would allow, the source must do so by sending (approximately) maximum MTU sized packets and adjusting the inter-departure interval to be approximately equal. The intent of this to ensure the data is sent relatively smoothly independent of the bit rate, subject to the first constraint.

B.1.2. Bottleneck Link Bandwidth

The bottleneck link capacity is dimensioned such that each RMCAT flow in an ideal situation with perfectly equal capacity sharing for all the flows on the bottleneck obtains the following throughputs: 200 kbps, 800 kbps, 1.3 Mbps and 4 Mbps.

For example, experiments with five RMCAT flows with an 800 kbps/flow target rate should set the bottleneck link capacity to 4 Mbps.

B.1.3. Bottleneck Link Queue Type and Length

The bottleneck link queue (Router A) is a simple FIFO queue having a buffer length corresponding to 70 ms, 500 ms or 2000 ms (defined in Section 6.4) of delay at the bottleneck link rate (i.e., actual buffer lengths in bytes are dependent on bottleneck link bandwidth).

B.1.4. RMCAT flows and delay legs

Experiments run with 1, 3, 5, 10 and 20 RMCAT sources, they are outlined as follows:

1. Experiments with 1, 3, and 5 RMCAT flows, all RMCAT flows commence simultaneously. A single delay leg is used and the link latency is set to one of the following : 0 ms, 50 ms and 150 ms.
2. For 10 and 20 source experiments where all RMCAT flows begin simultaneously the sources are split evenly into two different bulk delay legs. One leg is set to 0 ms bulk delay leg and the other is set to 150 ms.
3. For 10 and 20 source experiments where the first set will use 0 ms of bulk delay and the second set will use 150 ms bulk delay.
 1. Random starts within interval [0 ms, 500 ms].

2. One "early-coming" flow (i.e., the 1st flow starting and achieving steady-state before the next N-1 simultaneously begin).
3. One "late-coming" flow (i.e., the Nth flow starting after steady-state has occurred for the existing N-1 flows).

These cases assess if there are any early or late-comer advantages or disadvantages for a particular algorithm and to see if any unfairness is reproducible or unpredictable.

[Open issue (A.1): which group does the early and late flow belong to?]

[Open issue (A.2): Start rate for the media flows]

B.1.5. Impairment Generator

Packet loss is created in the reverse path (affects only feedback packets). Cases of 0%, 1%, 5% and 10% are studied for the 1, 3, and 5 RMCAT flow experiments, losses are not applied to flows with 10 or 20 RMCAT flows.

B.2. Proposed Passing Criteria

[Editor's note: there has been little or no discussion on the below criteria, however, they are listed here for the sake of completeness.]

No unfairness is observed, i.e., at steady state each flow attains a throughput between $[B/(3*N), (3*B)/N]$, where B is the link bandwidth and N is the number of flows.

No flow experiences packet loss when queue length is set to 500 ms or greater.

All individual sources must be in their steady state within twenty LRTTs (where LRTT is defined as the RTT associated with the flow with the Largest RTT in the experiment).]

B.3. Extensibility of the Experiment

The above scenario describes only RMCAT sources competing for capacity on the bottleneck link, however, future experiments can use different types of cross-traffic (as described in Section 6.1).

Currently, the forward path (carrying media packets) is characterized to add delay and a fixed bottleneck link capacity, in the future packet losses and capacity changes can be applied to mimic a wireless

link layer (for e.g., WiFi, 3G, LTE). Additionally, only losses are applied to the reverse path (carrying feedback packets), later experiments can apply the same forward path (carrying media packets) impairments to the reverse path.

Appendix C. Change Log

Note to the RFC-Editor: please remove this section prior to publication as an RFC.

C.1. Changes in draft-ietf-rmcat-eval-criteria-00

- o Updated references.
- o Resubmitted as WG draft.

C.2. Changes in draft-singh-rmcat-cc-eval-04

- o Incorporate feedback from IETF 87, Berlin.
- o Clarified metrics: convergence time, bandwidth utilization.
- o Changed fairness criteria to fairness test.
- o Added measuring pre- and post-repair loss.
- o Added open issue of measuring video quality to appendix.
- o clarified use of DropTail and AQM.
- o Updated text in "Minimum Requirements for Evaluation"

C.3. Changes in draft-singh-rmcat-cc-eval-03

- o Incorporate the discussion within the design team.
- o Added a section on evaluation parameters, it describes the flow and network characteristics.
- o Added Appendix with self-fairness experiment.
- o Changed bottleneck parameters from a proposal to an example set.

C.4. Changes in draft-singh-rmcat-cc-eval-02

- o Added scenario descriptions.

C.5. Changes in draft-singh-rmcat-cc-eval-01

- o Removed QoE metrics.
- o Changed stability to steady-state.
- o Added measuring impact against few and many flows.
- o Added guideline for idle and data-limited periods.
- o Added reference to TCP evaluation suite in example evaluation scenarios.

Authors' Addresses

Varun Singh
Aalto University
School of Electrical Engineering
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: varun@comnet.tkk.fi
URI: <http://www.netlab.tkk.fi/~varun/>

Joerg Ott
Aalto University
School of Electrical Engineering
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: jo@comnet.tkk.fi

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

Z. Sarker
I. Johansson
Ericsson AB
February 14, 2014

Evaluation Test Cases for Interactive Real-Time Media over Cellular
Networks
draft-sarker-rmcat-cellular-eval-test-cases-00

Abstract

It is evident that to ensure seamless and robust user experience across all type of access networks multimedia communication suits should adapt to the changing network conditions. There is an ongoing effort in IETF RMCAT working group to standardize rate adaptive algorithm(s) to be used in the real-time interactive communication. In this document test cases are described to evaluate the performances of the proposed endpoint adaptation solutions in a cellular network such as LTE network. It is aimed that the proposed solutions should be evaluated using the test cases defines in this document to select most optimal solutions.

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 August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminologies	4
3. Cellular Network Specific Test Cases	4
3.1. Varying Network Load	5
3.1.1. Network Connection	5
3.1.2. Simulation Setup	6
3.2. Bad Radio Coverage	7
3.2.1. Network connection	8
3.2.2. Simulation Setup	8
4. Desired Evaluation Metrics	8
5. Conclusion	9
6. Acknowledgements	9
7. IANA Considerations	9
8. Security Considerations	9
9. References	9
9.1. Normative References	9
9.2. Informative References	10
Authors' Addresses	10

1. Introduction

Cellular networks are an integral part of the Internet. Mobile devices connected to the cellular networks produces huge amount of media traffic in the Internet. It is important to evaluate the performance of the proposed RMCAT candidates in the cellular network.

A cellular environment is more complicated than a wireline ditto since it seeks to provide services in the context of variable available bandwidth, location dependencies and user mobilities at different speeds. In a cellular network the user may reach the cell edge which may lead to a significant amount of retransmissions to deliver the data from the base station to the destination and vice versa. These network links or radio links will often act as a bottleneck for the rest of the network which will eventually lead to excessive delays or packet drops. An efficient retransmission or link adaptation mechanism can reduce the packet loss probability but there will still be some packet losses and delay variations. Moreover, with increased cell load or handover to a congested cell, transport network level congestion will become even worse. Besides,

there are certain characteristics which make the cellular network different and challenging than other types of access network such as Wi-Fi and wired network. In a cellular network -

- o The bottleneck is often a shared link with relatively few users.
 - * The cost per bit over the shared link varies over time and is different for different users.
 - * Left over/ unused resource can be grabbed by other greedy users.
- o Queues are always per radio bearer hence each user can have many of such queues.
- o Users can experience both Inter and Intra Radio Access Technology (RAT) handovers ("handover" definition in [HO-def-3GPP]).
- o Handover between cells, or change of serving cells (see in [HO-LTE-3GPP] and [HO-UMTS-3GPP]) might cause user plane interruptions which can lead to bursts of packet losses, delay and /or jitter. The exact behavior depends on the type of radio bearer. Typically, the default best effort bearers do not generate packet loss, instead packets are queued up and transmitted once the handover is completed.
- o The network part decides how much the user can transmit.
- o The cellular network has variable link capacity per user
 - * Can vary as fast as a period of milliseconds.
 - * Depends on lots of facts (such as distance, speed, interference, different flows).
 - * Uses complex and smart link adaptation which makes the link behavior ever more dynamic.
 - * The scheduling priority depends on the estimated throughput.
- o Both Quality of Service (QoS) and non-QoS radio bearers can be used.

Hence, a real-time communication application operating in such a cellular network need to cope with shared bottleneck link and variable link capacity, event likes handover, non-congestion related loss, abrupt change in bandwidth (both short term and long term) due to handover, network load and bad radio coverage. Even though 3GPP

define QoS bearers [QoS-3GPP] to ensure high quality user experience, adaptive real-time applications are desired.

Different mobile operators deploy their own cellular network with their own set of network functionalities and policies. Usually, a mobile operator network includes 2G, EDGE, 3G and 4G radio access technologies. Looking at the specifications of such radio technologies it is evident that only 3G and 4G radio technologies can support the high bandwidth requirements from real-time interactive video applications. The future real-time interactive application will impose even greater demand on cellular network performance which makes 4G (and beyond radio technologies) more suitable access technology for such genre of application.

RMCAT evaluation criteria [I-D.ietf-rmcat-eval-criteria] document provides the guideline to perform the evaluation on candidate algorithms and recognize cellular networks to be important access link, however, it does not provides particular test cases to evaluate the performance of the candidate algorithm. In this document we device test cases specifically targeting cellular networks such as LTE networks.

2. Terminologies

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

3. Cellular Network Specific Test Cases

The key factors to define test cases for cellular network are

- o Shared and varying link capacity
- o Mobility
- o Handover

However, for cellular network it is very hard to separate such events from one another as these events are heavily related. Hence instead of devising separate test cases for all those important events we have divided the test case in two categories. It should be noted that in the following test cases the goal is to evaluate the performance of candidate algorithms over radio interface of the cellular network. Hence it is assumed that the radio interface is the bottleneck link between the communicating peers and that the core network does not add any extra congestion in the path. Also the combination of multiple access technologies such as one user has LTE

connection and another has Wi-Fi connection is kept out of the scope of this document. However, later those additional scenarios can also be added in this list of test cases. While defining the test cases we assumed a typical real-time telephony scenario over cellular networks where one real-time session consists of one voice stream and one video stream. We recommend that an LTE network simulator is used for the test cases defined in this document.

3.1. Varying Network Load

The goal of this test is to evaluate the performance of candidate congestion control algorithm under varying network load. The network load variation is created by adding and removing network users a.k.a. User Equipments (UEs) during the simulation. In this test case each of the user/UE in the media session is an RMCAT compliant endpoint. The arrival of users follows a Poisson distribution, which is proportional to the length of the call, so that the number of users per cell is kept fairly constant during the evaluation period. At the beginning of the simulation there should be enough amount of time to warm-up the network. This is to avoid running the evaluation in an empty network where network nodes are having empty buffers, low interference at the beginning of the simulation. This network initialization period is therefore excluded from the evaluation period.

This test case also includes user mobility and competing traffic. The competing traffics includes both same kind of flows (with same adaptation algorithms) and different kind of flows (with different service and congestion control). The investigated congestion control algorithms should show maximum possible network utilization and stability in terms of rate variations, lowest possible end to end frame latency, network latency and Packet Loss Rate (PLR) at different cell load level.

3.1.1. Network Connection

Each mobile user is connected to a fixed user. The connection between the mobile user and fixed user consists of a LTE radio access, an Evolved Packet Core (EPC) and an Internet connection. The mobile user is connected to the EPC using LTE radio access technology which is further connected to the Internet. The fixed user is connected to the Internet via wired connection with no bottleneck (practically infinite bandwidth). The Internet and wired connection in this setup does not add any network impairments to the test, it only adds 10ms of one-way transport propagation delay.

3.1.2. Simulation Setup

The desired simulation setup as follows-

1. Radio environment
 - A. Deployment : 3GPP case 1[Deployment]
 - B. Antenna: Multiple-Input and Multiple-Output (MIMO)
 - C. Mobility: [3km/h, 30km/h]
 - D. Transmission bandwidth: 10Mhz
 - E. Number of cells: multi cell deployment (3 Cells per Base Station (BS) * 7 BS) = 21 cells
 - F. Cell radius: 166.666 Meters
 - G. Scheduler: Proportional fair with nopriority
 - H. Bearer: Default bearer for all traffic.
 - I. Active Queue Management (AQM) settings: AQM [on,off]
2. End to end Round Trip Time (RTT): on avg. {40ms,.....,150ms}
3. User arrival model: Poisson arrival model
4. User intensity = {0.7,.....,10.5}
5. Simulation duration: 91s
6. Evaluation period : 30s-60s
7. Media traffic model
 1. Media Type: Video
 - a. Number of Media source per user: One (1)
 - b. Media direction: [Uplink, Downlink]
 - c. Media duration per user: 30s
 - d. Media codec : Variable BitRate (VBR)
 - e. Bitrate :

- Maximum bitrate: nominal 1.5 Mbps
- Minimum bitrate: nominal 150kbps
- f. Adaptation:
 - Bitrate: 150kbps-1.5mbps
 - Resolution: 144p-720p(1080p)
 - Framerate: 10fps-30fps

2. Media Type : Voice

- a. Number of Media source per user: One (1)
- b. Media direction: {Uplink, Downlink}
- c. Media duration per user: 30s
- d. Media codec: Constant BitRate (CBR)
- e. Bitrate :
 - Maximum bitrate: 20 kbps
- f. Adaptation: off

8. Other traffic model: Maximum of 2Mbps/cell (web browsing or FTP traffic)

3.2. Bad Radio Coverage

The goal of this test is to evaluate the performance of candidate congestion control algorithm when users visit part of the network with bad radio coverage. The scenario is created by using larger cell radius than previous test case. In this test case each of the user/UE in the media session is an RMCAT compliant endpoint. The arrival of users follows a Poisson distribution, which is proportional to the length of the call, so that the number of users per cell is kept fairly constant during the evaluation period. At the beginning of the simulation there should be enough amount of time to warm-up the network. This is to avoid running the evaluation in an empty network where network nodes are having empty buffers, low interference at the beginning of the simulation. This network initialization period is therefore excluded from the evaluation period.

This test case also includes user mobility and competing traffic. The competing traffics includes same kind of flows (with same adaptation algorithms) . The investigated congestion control algorithms should show maximum possible network utilization and stability in terms of rate variations, lowest possible end to end frame latency, network latency and Packet Loss Rate (PLR) at different cell load level.

3.2.1. Network connection

Same as defined in Section 3.1.1

3.2.2. Simulation Setup

The desired simulation setup is same as Varying Network Load test case defined in Section 3.1 except following changes-

1. Radio environment : Same as defined in Section 3.1.2 except followings
 - A. Deployment : 3GPP case 3[Deployment]
 - B. Cell radius: 577.3333 Meters
2. User intensity = {0.7,.....,7.0}
3. Media traffic model: Same as defined in Section 3.1.2
4. Other traffic model: None

4. Desired Evaluation Metrics

RMCAT evaluation criteria document [I-D.ietf-rmcat-eval-criteria] defines metrics to be used to evaluate candidate algorithms. However, looking at the nature and distinction of cellular networks we recommend at minimum following metrics to be used to evaluate the performance of the candidate algorithms for the test cases defined in this document.

The desired metrics are-

- o Average cell throughput (for all cells), shows cell utilizations
- o Application sending and receiving bitrate, goodput
- o Packet Loss Rate (PLR)
- o End to end Media frame delay

- o Transport delay
- o Algorithm stability in terms of rate variation

[Editor's note: This document needs to define the metrics clearly (or provide references to the respective definition) and the procedure of measuring those.]

5. Conclusion

This document defines two test cases that are considered important for cellular networks. Moreover, this document also provides a framework to define more additional test cases for cellular network.

6. Acknowledgements

We would like to thank Tomas Frankkila, Magnus Westerlund, Kristofer Kristofer Sandlund for their valuable comments while writing this draft.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

Security issues have not been discussed in this memo.

9. References

9.1. Normative References

[Deployment]

TS 25.814, 3GPP., "Physical layer aspects for evolved Universal Terrestrial Radio Access (UTRA)", October 2006, <http://www.3gpp.org/ftp/specs/archive/25_series/25.814/25814-710.zip>.

[HO-LTE-3GPP]

TS 36.331, 3GPP., "E-UTRA- Radio Resource Control (RRC); Protocol specification", December 2011, <http://www.3gpp.org/ftp/specs/archive/36_series/36.331/36331-990.zip>.

[HO-UMTS-3GPP]

TS 25.331, 3GPP., "Radio Resource Control (RRC); Protocol specification", December 2011, <http://www.3gpp.org/ftp/specs/archive/25_series/25.331/25331-990.zip>.

[HO-def-3GPP]

TR 21.905, 3GPP., "Vocabulary for 3GPP Specifications", December 2009, <http://www.3gpp.org/ftp/specs/archive/21_series/21.905/21905-940.zip>.

[I-D.ietf-rmcat-eval-criteria]

Singh, V. and J. Ott, "Evaluating Congestion Control for Interactive Real-time Media", draft-ietf-rmcat-eval-criteria-00 (work in progress), January 2014.

[QoS-3GPP]

TS 23.203, 3GPP., "Policy and charging control architecture", June 2011, <http://www.3gpp.org/ftp/specs/archive/23_series/23.203/23203-990.zip>.

[RFC2119]

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

9.2. Informative References

[I-D.ietf-rmcat-cc-requirements]

Jesup, R., "Congestion Control Requirements For RMCAT", draft-ietf-rmcat-cc-requirements-02 (work in progress), February 2014.

Authors' Addresses

Zaheduzzaman Sarker
Ericsson AB
Laboratoriegraend 11
Luleae 97753
Sweden

Phone: +46 107173743
Email: zaheduzzaman.sarker@ericsson.com

Ingemar Johansson
Ericsson AB
Laboratoriegraend 11
Luleae 97753
Sweden

Phone: +46 10 7143042
Email: ingemar.s.johansson@ericsson.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

Z. Sarker
Ericsson AB
V. Singh
Aalto University
X. Zhu
M. A. Ramalho
Cisco Systems
February 14, 2014

Test Cases for Evaluating RMCAT Proposals
draft-sarker-rmcat-eval-test-00

Abstract

The Real-time Transport Protocol (RTP) is used to transmit media in multimedia telephony applications, these applications are typically required to implement congestion control. The RMCAT working group is currently working on candidate algorithms for such interactive real-time multimedia applications. This document describes the test cases to be used in the evaluation of the performance of those candidate algorithms.

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 August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Basic Structure of Test cases	3
4. Basic Test Cases	7
4.1. Variable Available Capacity	7
4.2. Maximum Media Bit Rate is Greater than Link Capacity	10
4.3. Competing Flows with same RMCAT Algorithm	13
4.4. RMCAT Flow competing with a long TCP Flow	16
4.5. RMCAT Flow competing with short TCP Flows	19
4.6. Congested Feedback Link	22
4.7. Round Trip Time Fairness	27
4.8. Media Pause and Resume	29
4.9. Explicit Congestion Notification Usage	31
5. Wireless Access Links	31
5.1. Cellular Network Specific Test Cases	31
5.2. Wi-Fi Network Specific Test Cases	31
6. Security Considerations	31
7. IANA Considerations	31
8. Acknowledgements	31
9. References	32
9.1. Normative References	32
9.2. Informative References	33
Authors' Addresses	33

1. Introduction

This memo describes a set of test cases for evaluating candidate RMCAT congestion control algorithm proposals, it is based on the guidelines enumerated in [I-D.ietf-rmcat-eval-criteria] and requirements discussed in [I-D.ietf-rmcat-cc-requirements]. The test cases cover basic usage scenarios and are described using a common structure, which allows for additional test cases to be added to those described herein to accommodate other topologies and/or the modeling of different link characteristics. Each test case incorporates the metrics, evaluation guidelines and parameters described in [I-D.ietf-rmcat-eval-criteria]. It is the intention of this work to capture the consensus of the RMCAT working group participants regarding the test cases upon which the performance of the candidate RMCAT proposals should be evaluated.

2. Terminology

The terminology defined in RTP [RFC3550], RTP Profile for Audio and Video Conferences with Minimal Control [RFC3551], RTCP Extended Report (XR) [RFC3611], Extended RTP Profile for RTCP-based Feedback (RTP/AVPF) [RFC4585], Support for Reduced-Size RTCP [RFC5506], and RTP Circuit Breaker algorithm [I-D.ietf-avtcore-rtcp-circuit-breakers] apply.

3. Basic Structure of Test cases

All test cases in this document follow a basic structure allowing implementers to describe a new test scenarios without repeatedly explaining common attributes. The structure includes a general description section that describes the test case and motivations, additionally it defines a set of attributes that characterize the testbed, i.e., the network path between communicating peers and the diverse traffic sources.

- o Define the test case:

- * General description: describes the motivation and the goals of the test case.

- * Additionally, describe the desired rate adaptation behaviour.

- * Define a checklist to evaluate the desired behaviour: this indicates the minimum set of metrics (e.g., link utilization, media sending rate) that a proposed algorithm needs to measure to validate the expected rate adaptation behaviour. It should also indicate the time granularity (e.g., averaged over 10ms, 100ms, or 1s) for measuring certain metrics. Typical measurement interval is 200ms.

- o Define testbed topology: every test case needs to define an evaluation testbed topology. Figure 1 shows such an evaluation topology. In this evaluation topology, S1..Sn are traffic sources, these sources either generate media traffic and use an RMCAT candidate congestion control algorithm or use a different congestion control algorithm than that one used in the media source. R1..Rn are the corresponding receivers. A test case can have one or more such traffic source (S) and corresponding receiver (R). The path from the source to destination is denoted as upstream and the path from a destination to a source is denoted as downstream. It is possible for a testbed to have different path characteristics in either directions. The following basic structure of test case has been described from the perspective of endpoints attached on the left-hand side of Figure 1.

O

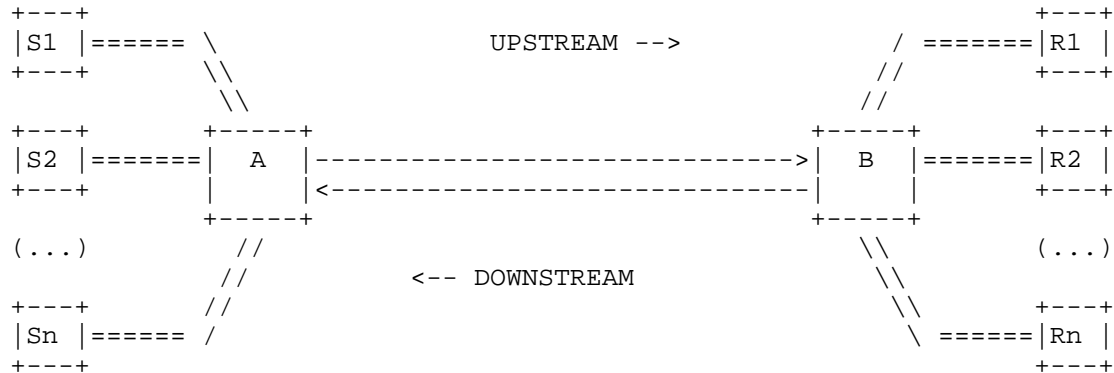


Figure 1: Example of A Testbed Topology

- o In a laboratory testbed environment there exists large amount of traffic on the network path between the endpoints, which may be contributed by other devices connected to such a testbed. It is recommended to not route non-test traffic through the testbed.
- o Define testbed attributes:
 - * Duration: defines the duration of the test.
 - * Path characteristics: defines the end-to-end transport level path characteristics of the testbed in a particular test case. Two sets of attributes describes the path characteristics, one for the upstream path and the other for the downstream path. The path characteristics for a particular path direction is applicable to all the Sources "S" sending traffic on that path. If only one attribute is specified, it is used for both path directions.
 - + Path direction: upstream or downstream.
 - + Bottleneck-link capacity: defines minimum capacity of the end-to-end path
 - + One-way propagation delay: describes the end-to-end latency along the path when network queues are empty i.e., the time it takes for a packet to go from the sender to the receiver, it does not include any queuing delay.
 - + Maximum end-to-end jitter: defines maximum jitter can be observed along the path.

- o Define testbed attributes:

- * Duration: defines the duration of the test.
- * Path characteristics: defines the end-to-end transport level path characteristics of the testbed in a particular test case. Two sets of attributes describes the path characteristics, one for the upstream path and the other for the downstream path. The path characteristics for a particular path direction is applicable to all the Sources "S" sending traffic on that path. If only one attribute is specified, it is used for both path directions.
 - + Path direction: upstream or downstream.
 - + Bottleneck-link capacity: defines minimum capacity of the end-to-end path
 - + One-way propagation delay: describes the end-to-end latency along the path when network queues are empty i.e., the time it takes for a packet to go from the sender to the receiver, it does not include any queuing delay.
 - + Maximum end-to-end jitter: defines maximum jitter can be observed along the path.

- * Path characteristics: defines the end-to-end transport level path characteristics of the testbed in a particular test case. Two sets of attributes describes the path characteristics, one for the upstream path and the other for the downstream path. The path characteristics for a particular path direction is applicable to all the Sources "S" sending traffic on that path. If only one attribute is specified, it is used for both path directions.
- + Path direction: upstream or downstream.
- + Bottleneck-link capacity: defines minimum capacity of the end-to-end path
- + One-way propagation delay: describes the end-to-end latency along the path when network queues are empty i.e., the time it takes for a packet to go from the sender to the receiver, it does not include any queuing delay.
- + Maximum end-to-end jitter: defines maximum jitter can be observed along the path.

- + Path direction: upstream or downstream.
- + Bottleneck-link capacity: defines minimum capacity of the end-to-end path
- + One-way propagation delay: describes the end-to-end latency along the path when network queues are empty i.e., the time it takes for a packet to go from the sender to the receiver, it does not include any queuing delay.
- + Maximum end-to-end jitter: defines maximum jitter can be observed along the path.

- ```
+ Bottleneck-link capacity: defines minimum capacity of the
 end-to-end path
```

- + One-way propagation delay: describes the end-to-end latency along the path when network queues are empty i.e., the time it takes for a packet to go from the sender to the receiver, it does not include any queuing delay.

- + Maximum end-to-end jitter: defines maximum jitter can be observed along the path.

- + Bottleneck queue type: for example, Droptail, FQ-CoDel, or PIE.
- + Bottleneck queue size: defines size of queue in terms of queuing time when the queue is full (in milliseconds).
- + Path loss ratio: characterize the non-congested losses observed on the end-to-end path. MUST describe the loss pattern or loss model.
- \* Application-related: defines the media-related behaviour for implementing the test case
  - + Media Source: defines the characteristics of the media sources present. When using more than one media source, the different attributes are enumerated separately for each different media source.
    - Media type: Video/Voice/Application/Text
    - Media flow direction: upstream, downstream or both.
    - Number of media sources: defines the total number of media sources
    - Media codec: Constant Bit Rate (CBR) or Variable Bit Rate (VBR)
    - Media source behaviour: describes the media encoder behavior. It MUST define the main parameters that affect the adaptation behaviour. This may include but not limited to:
      - o Adaptability: describes the adaptation options. For example, in case of video it defines the range of bitrate adaptation, frame rate adaptation, video resolution adaptation. In case of voice, it defines the range of bitrate adaptation, sampling rate, frame size.
      - o Output variation : for VBR encoder, it defines the encoder output variation from the average target rate. For example on average the encoder output may vary between 5% to 15% above or below the average target bit rate.
      - o Encoder's responsiveness to a new bit rate request: value typically between 10ms to 1000ms.

- Media content: describes the chosen media sequences; For example, test sequences are available at: [xiph-seq] [HEVC-seq].
- Media timeline: describes the point when the media source is introduced and removed from the testbed. For example, the media source may begin transmitting when the test case begins or a few seconds after, etc.
- Startup behaviour: the media starts at a defined bit rate, which may be the minimum, maximum bit rate, or a value in between (in Kbps).
- + Competing traffic source: describes the characteristics of the competing traffic source, the different types of competing flows are enumerated in [I-D.ietf-rmcat-eval-criteria].
  - Traffic direction: Upstream, downstream or both.
  - Type of sources: defines the types of competing traffic sources. Types of competing traffic flows are listed in [I-D.ietf-rmcat-eval-criteria]. For example, the number of TCP flows connected to a web browser, the mean size and distribution of the content downloaded.
  - Number of sources: defines the total number of competing sources of each type.
  - Congestion control: enumerate the congestion control used by each type of competing traffic.
  - Traffic timeline: describes when the competing traffic is added and removed from the test case.
- \* Additional attributes: describes attributes essential for implementing a test case which are not included in the above structure. These attributes MUST be well defined, so that other implementers are able to implement it.

Any attribute can have a set of values (enclosed within "[ ]"). Each member value of such a set MUST be treated as different value for the same attribute. It is desired to run separate tests for each such attribute value.

The test cases described in this document follow the above structure.

## 4. Basic Test Cases

### 4.1. Variable Available Capacity

In this test case the bottleneck-link capacity between the two endpoints varies over time. This test is designed to measure the responsiveness of the candidate algorithm. This test tries to address the requirement 1(a) in [I-D.ietf-rmcat-cc-requirements], which requires the algorithm to adapt the flow(s) and provide lower end-to-end latency when there exists:

- o an intermediate bottleneck
- o change in available capacity due to interface change and/or routing change.
- o persistent network load due to competing traffic

It should be noted that the exact variation in available capacity due to any of the above depends on the under-lying technologies. Hence, we describe a set of known factors, which may be extended to devise a more specific test case targeting certain behaviour in a certain network environment.

Expected behavior: the candidate algorithms is expected to detect the variation in available capacity and adapt the media stream(s) accordingly. The flows stabilize around their maximum bitrate as the as the maximum link capacity is large enough to accommodate the flows. When the available capacity drops, the flow(s) adapts by decreasing its sending bit rate, and when congestion disappears, the flow(s) are again expected to ramp up.

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics at a fine enough time granularity:

1. Flow level:
  - a. End-to-end delay
  - b. Losses observed at the receiving endpoint
  - c. Feedback message overhead
2. Transport level:
  - a. Bandwidth utilization



## b. Queue length (ms):

- + average over the length of the session
- + 5 and 95 percentile
- + median, maximum, minimum

Testbed Topology: Two (2) media sources S1 and S2 are connected to their corresponding R1 and R2. The media traffic is transported over the upstream path and corresponding feedback/control traffic is transported over the downstream path.

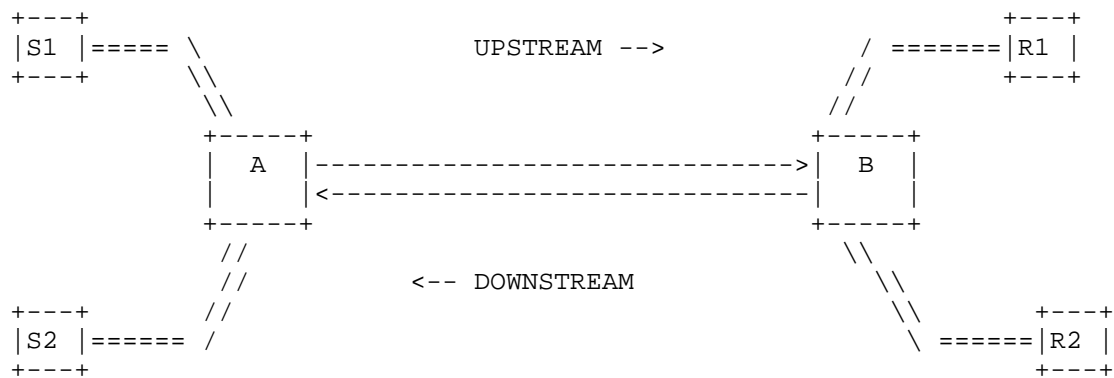


Figure 2: Testbed Topology for Variable Available Capacity

## Testbed attributes:

- o Test duration: 120s
- o Path characteristics:
  - \* Path direction: Upstream and downstream.
  - \* Bottleneck-link capacity: 4Mbps.
  - \* One-Way propagation delay: [10ms, 100ms].
  - \* Maximum end-to-end jitter: 30ms.
  - \* Bottleneck queue type: Drop tail.
  - \* Bottleneck queue size: 300ms.
  - \* path loss ratio: 0%.

- o Application-related:
  - \* Media Source:
    - + Media type: Video
    - + Media direction: Upstream.
    - + Number of media sources: Two (2).
    - + Media codec: VBR
    - + Media source behaviour:
      - Adaptability:
        - o Bit rate range: 150 Kbps - 1.5 Mbps
        - o Frame Resolution: 144p - 720p (or 1080p)
        - o Frame rate: 10fps - 30fps
      - Variation from target bitrate: +/-5%
      - Responsiveness to new bit rate request: 100ms
    - + Media content: Foreman media sequence.
    - + Media timeline:
      - Start time: 0s.
      - End time: 119s.
  - \* Media startup behaviour: [200Kbps, 1500Kbps].
  - \* Competing traffic:
    - + Number of sources : Zero (0)
- o Test specific setup:
  - \* Number of bandwidth variations: Three (4)
  - \* Variation pattern:
    - + Sequence number: 1

- + Path direction: Upstream
- + Bottleneck Capacity: 2Mbps.
- + Start time: 30s
- \* Variation pattern:
  - + Sequence number: 2
  - + Path direction: Upstream
  - + Bottleneck Capacity: 3.5Mbps
  - + Start time: 50s
- \* Variation pattern:
  - + Sequence number: 3
  - + Path direction: Upstream
  - + Bottleneck Capacity: 1Mbps
  - + Start time: 70s
- \* Variation pattern:
  - + Sequence number: 4
  - + Path direction: Upstream
  - + Bottleneck Capacity: 2Mbps
  - + Start time: 100s

#### 4.2. Maximum Media Bit Rate is Greater than Link Capacity

In this case, the application will attempt to ramp up to its maximum bit rate, since the link capacity is limited to a value lower, the congestion control scheme is expected to stabilize the sending bit rate close to the available bottleneck capacity. This situation can occur when the endpoints are connected via thin long networks even though the advertised capacity of the access network may be higher. The test case addresses the requirement 1 and 10 of the [I-D.ietf-rmcat-cc-requirements].

Expected behavior: the candidate algorithm is expected to detect the path capacity constraint, converges to bottleneck link's capacity and adapt the flow to avoid unwanted oscillation when the sending bit rate is approaching the bottleneck link's capacity. The oscillations occur when the media flow(s) attempts to reach its maximum bit rate, overshoots the usage of the available bottleneck capacity, to rectify it reduces the bit rate and starts to ramp up again.

Testbed topology: One media source S1 is connected to corresponding R1. The media traffic is transported over the upstream path and corresponding feedback/control traffic is transported over the downstream path.

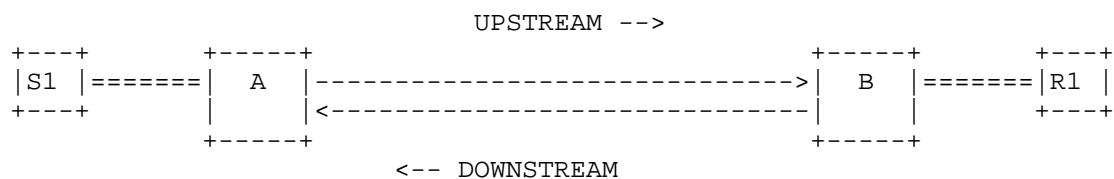


Figure 3: Testbed Topology for Limited Link Capacity

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics at a fine enough time granularity:

1. Flow level:
  - a. End-to-end delay.
  - b. RTP packet losses observed at the receiving endpoint.
  - c. Variation in sending bit rate and goodput. Mainly observing the frequency and magnitude of oscillations.
  - d. Convergence time.
  - e. Feedback message overhead.
2. Transport level:
  - a. Bandwidth utilization.
  - b. Queue length (ms):
    - + average over the length of the session
    - + 5 and 95 percentile

- + median

Testbed attributes:

- o Test duration: 60s
- o Path characteristics:
  - \* Path direction: Upstream and downstream.
  - \* Bottleneck-link capacity: 1Mbps
  - \* One-Way propagation delay: [10ms,100 ms]
  - \* Maximum end-to-end jitter: 30ms.
  - \* Bottleneck queue type: Droptail. Additional tests with other AQM schemes are recommended: FQ-CoDel, PIE
  - \* Bottleneck size: 300ms
  - \* Path loss ratio: 0%
- o Application-related:
  - \* Media Source:
    - + Media type: Video
    - + Media direction: Upstream.
    - + Number of media sources: One (1)
    - + Media codec: VBR
    - + Media source behaviour:
      - Adaptability:
        - o Bit rate range: 150 Kbps - 1.5 Mbps
        - o Frame Resolution: 144p - 720p (or 1080p)
        - o Frame rate: 10fps - 30fps
      - Variation from target bitrate: +/-5%
      - Responsiveness to new bit rate request: 100ms

- + Media content: Foreman video sequence
- + Media timeline:
  - Start time: 0s.
  - End time: 59s.
- \* Media startup behaviour: [200Kbps, 1500Kbps].
- \* Competing traffic:
  - + Number of sources : Zero (0)
- o Test specific setup: None

#### 4.3. Competing Flows with same RMCAT Algorithm

In this test case, more than one RMCAT media flow shares the bottleneck link and each of them uses the same congestion control algorithm. This is a typical scenario wherein a real-time interactive application sends more than one media flows to the same destination and these flows are multiplexed over the same port. In such a scenario it is likely that the flows will be routed via the same path and need to share the available bandwidth amongst themselves. For the sake of simplicity it is assumed that there are no other competing traffic sources in the bottleneck link and that there is sufficient capacity to accommodate all the flows individually. While this appears to be a variant of the test case defined in Section 4.1 , however it tests the capacity sharing distribution of the candidate algorithm. Whereas, the previous test case measures the stability and responsiveness of the candidate algorithm. This test case particularly addresses the requirements 2,3 and 10 in [I-D.ietf-rmcat-cc-requirements].

Expected behavior: It is expected that the competing flows will converge to an optimum bit rate to accommodate all the flows with minimum possible latency and loss. Specifically, the test introduces three media flows at different time instances, when the second flow appears there should still be room to accommodate another flow on the bottleneck link. Lastly, when the third flow appears the bottleneck link should be saturated.

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics metrics at a fine enough time granularity:

1. Flow level:

- a. End-to-end delay.
  - b. RTP packet losses observed at the receiving endpoint.
  - c. Variation in sending bit rate and goodput. Mainly observing the frequency and magnitude of oscillations.
  - d. Convergence time.
  - e. Feedback message overhead.
2. Transport level:
- a. Bandwidth utilization.
  - b. Queue length (ms):
    - + average over the length of the session
    - + 5 and 95 percentile
    - + median

Testbed topology: Three media sources S1, S2, S3 are connected to respective R1, R2, R3. The media traffic is transported over the upstream path and corresponding feedback/control traffic is transported over the downstream path.

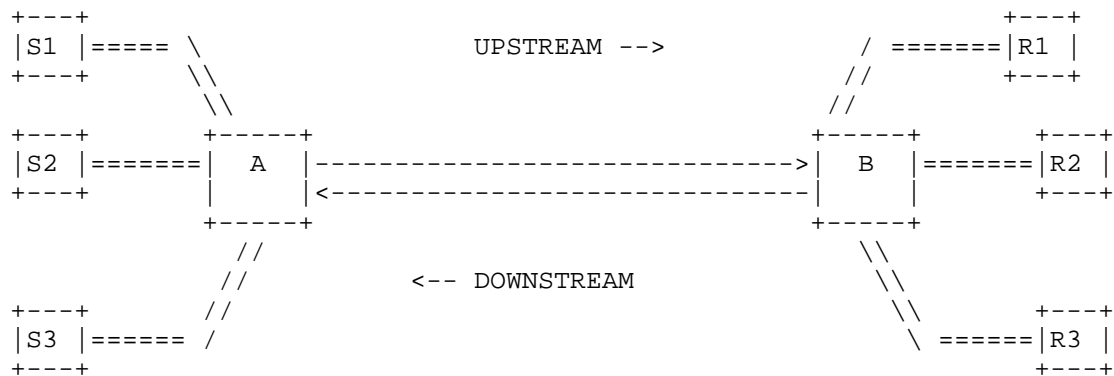


Figure 4: Testbed Topology for Multiple RMCAT Flows

Testbed attributes:

- o Test duration: 60s

- o Path characteristics:
  - \* Path direction: Upstream, Downstream
  - \* Bottleneck-link capacity: 3.5Mbps
  - \* One-Way propagation delay: [10ms, 50ms]
  - \* Maximum end to end jitter: 30ms
  - \* Bottleneck queue type: Droptail
  - \* Bottleneck queue size: 300ms
  - \* Link loss ratio: 0.0%
- o Application-related:
  - \* Media Source:
    - + Media type: Video
    - + Media direction: Upstream
    - + Number of media sources: Three (3)
    - + Media codec: VBR
    - + Media source behaviour:
      - Adaptability:
        - o Bit rate range: 150 Kbps - 1.5 Mbps
        - o Frame Resolution: 144p - 720p (or 1080p)
        - o Frame rate: 10fps - 30fps
      - Variation from target bitrate: +/-5%
      - Responsiveness to new bit rate request: 100ms
    - + Media content: Foreman video sequence
    - + Media timeline: New media flows are added sequentially, at short time intervals. See test specific setup below.
    - + Media startup behaviour: 200Kbps.



- \* Competing traffic:
  - + Number of sources : Zero (0)
- o Test specific setup:
  - \* Media flow timeline:
    - + Flow no: One (1)
    - + Start time: 0s
    - + End time: 59s
  - \* Media flow appearance:
    - + Flow no: Two (2)
    - + Start time: 20s
    - + End time: 59s
  - \* Media flow appearance:
    - + Flow no: Three (3)
    - + Start time: 40s
    - + End time: 59s

#### 4.4. RMCAT Flow competing with a long TCP Flow

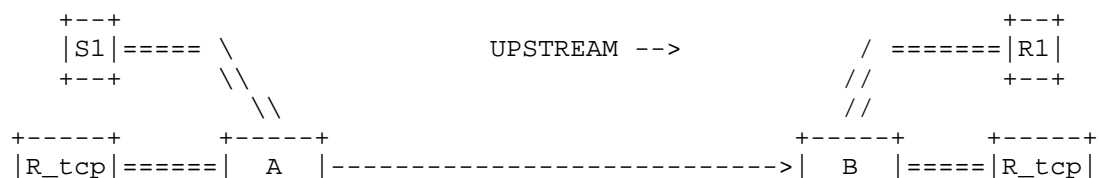
In this test case, one or more RMCAT media flow shares the bottleneck link with at least one long lived TCP flows. Long lived TCP flows download data throughout the session and are expected to have infinite amount of data to send and receive. This is a scenario wherein a multimedia application co-exists with a large file download. The test case measures the adaptivity of the candidate algorithm to competing traffic, it addresses the requirement 8 in [I-D.ietf-rmcat-cc-requirements].

Expected behavior: depending on the convergence observed in test case 4.1 and 4.2, the candidate algorithm may be able to avoid congestion collapse. In the worst case, the media stream will fall to the minimum media bit rate.

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics at a fine enough time granularity:

1. Flow level:
  - a. End-to-end delay for the RMCAT flow.
  - b. RTP packet losses observed at the receiving endpoint.
  - c. Variation in sending bit rate and goodput. Mainly observing the frequency and magnitude of oscillations.
  - d. Variation in the sending rate of the TCP flow
  - e. TCP throughput.
  - f. Convergence time.
  - g. Feedback message overhead.
2. Transport level:
  - a. Bandwidth utilization.
  - b. Queue length (ms):
    - + average over the length of the session
    - + 5 and 95 percentile

Testbed topology: One (1) media source S1 is connected to corresponding R1, but both endpoints are additionally receiving and sending data, respectively. The media traffic (S1->S2) is transported over the upstream path and corresponding feedback/control traffic is transported over the downstream path. Likewise media traffic (S2->S1) is transported over the downstream path and corresponding feedback/control traffic is transported over the upstream path. The TCP uploads traffic over upstream path and downloads over downstream path. Hence, TCP traffic exits in both path directions.



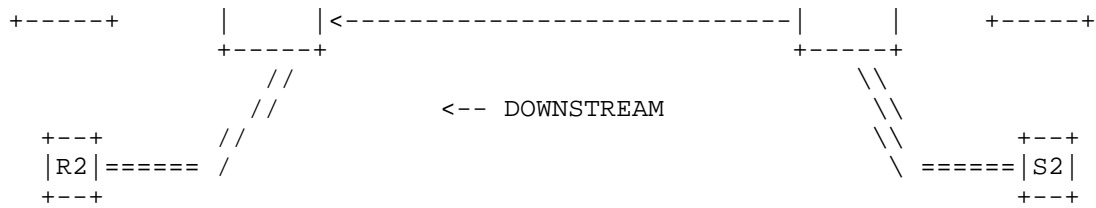


Figure 5: Testbed Topology for TCP vs RMCAT Flows

## Testbed attributes:

- o Test duration: 120s
- o Path characteristics:
  - \* Path direction: Upstream, Downstream
  - \* Bottleneck-link capacity: 2Mbps
  - \* One-Way propagation delay: [10ms, 150ms]
  - \* Maximum end to end jitter: 30ms
  - \* Bottleneck queue type: Droptail, but would benefit from running the same test with different AQM schemes: FQ-Codel, or PIE.
  - \* Bottleneck queue size: [20ms, 250ms, 1000ms]
  - \* Path loss ratio: 0.0%
- o Application-related:
  - \* Media Source:
    - + Media type: Video
    - + Media direction: Upstream and Downstream
    - + Number of media sources: One (1)
    - + Media codec: VBR
    - + Media source behaviour:
      - Adaptability:
        - o Bit rate range: 150 Kbps - 1.5 Mbps

- o Frame Resolution: 144p - 720p (or 1080p)
  - o Frame rate: 10fps - 30fps
  - Variation from target bitrate: +/-5%
  - Responsiveness to new bit rate request: 100ms
- + Media content: Foreman video sequence
- + Media timeline:
  - Start time: 5s.
  - End time: 119s.
- + Media startup behaviour: [200Kbps, 1500Kbps].
- \* Competing traffic:
  - + Number and Types of sources : one (1), long-lived TCP
  - + Traffic direction : Downstream
  - + Congestion control: Default TCP congestion control.
  - + Traffic timeline:
    - Start time: 0s.
    - End time: 119s.
- o Test specific setup: None

#### 4.5. RMCAT Flow competing with short TCP Flows

In this test case, one or more RMCAT media flow shares the bottleneck link with at multiple short-lived TCP flows. Short-lived TCP flows resemble the on/off pattern observed in the web traffic, wherein clients (browsers) connect to a server and download a resource (typically a webpage, few images, text files, etc.) using several TCP connections (up to 4). This scenario shows the performance of the multimedia application when several browser windows are active. The test case measures the adaptivity of the candidate algorithm to competing web traffic, it addresses the requirements 2 in [I-D.ietf-rmcate-cc-requirements].

Depending on the number of short TCP flows, the cross-traffic either appears as a short burst flow or resembles a long TCP flow. The intention of this test is to observe the impact of short-term burst on the behaviour of the candidate algorithm.

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics at a fine enough time granularity:

1. Flow level:
  - a. End-to-end delay for the RMCAT flow.
  - b. RTP packet losses observed at the receiving endpoint.
  - c. Variation in sending bit rate and goodput. Mainly observing the frequency and magnitude of oscillations.
  - d. Variation in the sending rate of the TCP flow.
  - e. TCP throughput.
  - f. Convergence time.
  - g. Feedback message overhead.
2. Transport level:
  - a. Bandwidth utilization.
  - b. Queue length (ms):
    - + average over the length of the session
    - + 5 and 95 percentile

Testbed topology: One (1) media source S1 is connected to corresponding R1, but both endpoints are additionally receiving and sending data, respectively. The media traffic (S1->S2) is transported over the upstream path and corresponding feedback/control traffic is transported over the downstream path. Likewise media traffic (S2->S1) is transported over the downstream path and corresponding feedback/control traffic is transported over the upstream path. The TCP uploads traffic over upstream path and downloads over downstream path. Hence, TCP traffic exits in both path directions. The topology described here is similar to the one described in Figure 5.

## Testbed attributes:

- o Test duration: 300s
- o Path characteristics:
  - \* Path direction: Upstream, Downstream
  - \* Bottleneck-link capacity: 2.0Mbps
  - \* One-Way propagation delay: [10ms, 150ms]
  - \* Maximum end to end jitter: 30ms
  - \* Bottleneck queue type: Droptail, but would benefit from running the same test with different AQM schemes: FQ-Codel, or PIE.
  - \* Bottleneck queue size: 300ms
  - \* Path loss ratio: 0.0%
- o Application-related:
  - \* Media Source:
    - + Media type: Video
    - + Media direction: Upstream and Downstream
    - + Number of media sources: One (1)
    - + Media codec: VBR
    - + Media source behaviour:
      - Adaptability:
        - o Bit rate range: 150 Kbps - 1.5 Mbps
        - o Frame Resolution: 144p - 720p (or 1080p)
        - o Frame rate: 10fps - 30fps
      - Variation from target bitrate: +/-5%
      - Responsiveness to new bit rate request: 100ms
    - + Media content: Foreman video sequence

- + Media timeline:
  - Start time: 0s.
  - End time: 299s.
- + Media startup behaviour: [200Kbps, 1500Kbps].
- \* Competing traffic:
  - + Number and Types of sources : Ten (10), short-lived TCP flows.
  - + Traffic direction : Downstream
  - + Congestion algorithm: Default TCP Congestion control.
  - + Traffic timeline: Each short TCP flow is modeled as a sequence of file downloads interleaved with idle periods. See test specific setup. Not all short TCPs start at the same time, 2 start in the ON state while 8 start in an OFF state. The model for the idle times for the OFF state is discussed in the Short-TCP model.
- o Test specific setup:
  - \* Short-TCP traffic model:
    - + File sizes: uniform distribution between 100KB to 1MB
    - + Idle period: the duration of the OFF state is derived from an exponential distribution with the mean value of 10 seconds.

#### 4.6. Congested Feedback Link

RMCAT WG has been chartered to define algorithms for RTP hence it is assumed that RTCP, RTP header extension or such would be used as signalling means for the adaptation algorithm in the backchannel. Due to asymmetry nature of the link between communicating peers it is possible to observe lack such backchannel information due to impaired backchannel link (even when forward channel might be unimpaired). This test case is designed to observe candidate congestion control behaviour in such an event. This test case addresses requirement number 5 and in particular, requirement number 7.

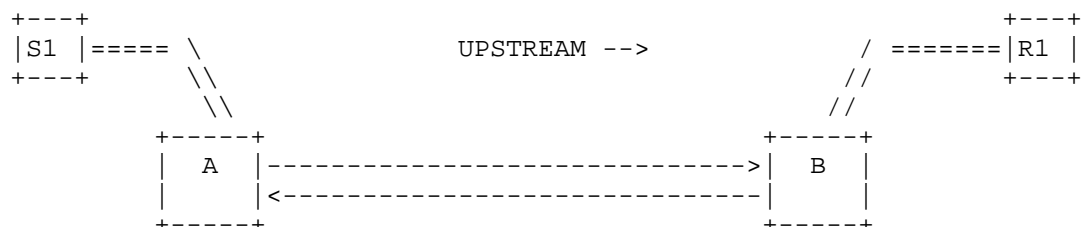
It is expected that the candidate algorithms should cope with the lack of backchannel information and adapt to minimize the performance degradation of media flows in the forward channel.

It SHOULD be noted that for this test case log is needed for the reference case where the downstream channel has no impairments

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics at a fine enough time granularity:

1. Flow level:
  - a. End-to-end delay
  - b. Losses observed at the receiving endpoint
  - c. Feedback message overhead
2. Transport level:
  - a. Bandwidth utilization
  - b. Queue length (ms):
    - + average over the length of the session
    - + 5 and 95 percentile

Testbed topology: One (1) media source S1 is connected to corresponding R1, but both endpoints are additionally receiving and sending data, respectively. The media traffic (S1->S2) is transported over the upstream path and corresponding feedback/control traffic is transported over the downstream path. Likewise media traffic (S2->S1) is transported over the downstream path and corresponding feedback/control traffic is transported over the upstream path.





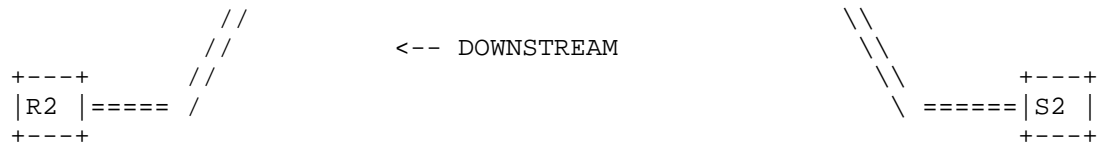


Figure 6: Testbed Topology for Congested Feedback Link

## Testbed attributes:

- o Test duration: 60s
- o Path characteristics:
  - \* Path direction: Upstream and downstream.
  - \* Bottleneck-link capacity: 2Mbps.
  - \* One-Way propagation delay: [10ms, 100ms].
  - \* Maximum end-to-end jitter: 30ms.
  - \* Bottleneck queue type: Drop tail.
  - \* Bottleneck queue size: 300ms.
  - \* path loss ratio: 0%.
- o Application-related:
  - \* Media Source:
    - + Media type: Video
    - + Media direction: Upstream
    - + Number of media sources: One (1).
    - + Media codec: VBR
    - + Media source behaviour:
      - Adaptability:
        - o Bit rate range: 150 Kbps - 1.5 Mbps
        - o Frame Resolution: 144p - 720p (or 1080p)

- o Frame rate: 10fps - 30fps
  - Variation from target bitrate: +/-5%
  - Responsiveness to new bit rate request: 100ms
- + Media content: Foreman media sequence
- + Media timeline:
  - Start time: 0s.
  - End time: 59s.
- \* Media Source:
  - + Media type: Video
  - + Media direction: Downstream
  - + Number of media sources: One (1).
  - + Media codec: VBR
  - + Media source behaviour:
    - Adaptability:
      - o Bit rate range: 150 Kbps - 1.5 Mbps
      - o Frame Resolution: 144p - 720p (or 1080p)
      - o Frame rate: 10fps - 30fps
    - Variation from target bitrate: +/-5%
    - Responsiveness to new bit rate request: 100ms
  - + Media content: Foreman media sequence
  - + Media timeline:
    - Start time: 0s.
    - End time: 59s.
- \* Media startup behaviour: 200Kbps.

- \* Competing traffic:
  - + Number of sources : Zero (0)
- o Test specific setup:
  - \* Number of bandwidth variation: Two (2)
  - \* Variation pattern:
    - + Sequence number: 1
    - + Path direction: Upstream
    - + Bottleneck Capacity: 1Mbps
    - + Start time: 20s
  - \* Variation pattern:
    - + Sequence number: 2
    - + Path direction: Upstream
    - + Bottleneck Capacity: 600Kbps
    - + Start time: 30s
  - \* Variation pattern:
    - + Sequence number: 3
    - + Path direction: Upstream
    - + Bottleneck Capacity: 2Mbps
    - + Start time: 50s
  - \* Variation pattern:
    - + Sequence number: 1
    - + Path direction: Downstream
    - + Bottleneck Capacity: 800kbps
    - + Start time: 30s

- \* Variation pattern:
  - + Sequence number: 2
  - + Path direction: Downstream
  - + Bottleneck Capacity: 2Mbps
  - + Start time: 50s

#### 4.7. Round Trip Time Fairness

In this test case, more than one RMCAT media flow shares the bottleneck link, but the end-to-end path latency for each RMCAT flow is different. For the sake of simplicity it is assumed that there are no other competing traffic sources in the bottleneck link and that there is sufficient capacity to accommodate all the flows. While this appears to be a variant of the test case 4.2, it tests the capacity sharing distribution of the candidate algorithm under different RTTs. This test case particularly addresses the requirements 2 [I-D.ietf-rmcat-cc-requirements].

It is expected that the competing flows will converge to an optimum bit rate to accommodate all the flows with minimum possible latency and loss. Specifically, the test introduces five media flows at the same time instance.

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics at a fine enough time granularity:

1. Flow level:
  - a. End-to-end delay.
  - b. RTP packet losses observed at the receiving endpoint.
  - c. Variation in sending bit rate and goodput. Mainly observing the frequency and magnitude of oscillations.
  - d. Convergence time.
2. Transport level:
  - a. Bandwidth utilization.
  - b. Queue length (ms):

- + average over the length of the session
- + 5 an 95 percentile

Testbed Topology: Five (5) media sources S1,S2,...,S5 are connected to their corresponding media sinks R1,R2,...,R5. The media traffic is transported over the upstream path and corresponding feedback/control traffic is transported over the downstream path. The topology is the same as test case 4.3 defined in Section 4.3. The end-to-end path delay for S1-R1 is 10ms, S2-R2 is 25ms, S3-R3 is 50ms, S4-R4 is 100ms, S5-R5 is 150ms.

Testbed attributes:

- o Test duration: 60s
- o Path characteristics:
  - \* Path direction: Upstream, Downstream
  - \* Number of bottlenecks: One (1)
  - \* Bottleneck link capacity: 3.0Mbps
  - \* One-Way propagation delay for each path is: 10ms, 25ms, 50ms, 100ms, 150ms.
  - \* Maximum end to end jitter: 30ms
  - \* Bottleneck queue type: Droptail
  - \* Bottleneck queue size: 300ms
  - \* Link loss ratio: 0.0%
- o Application-related:
  - \* Media Source:
    - + Media direction: Upstream
    - + Number of media sources: Five (5)
    - + Encoder configuration:
      - Bit rate generation: VBR
      - Bit rate range: 150 Kbps - 1.5 Mbps

- Frame Resolution: 144p-720p (or 1080p)
- Frame rate: 10fps-30fps
- Variation from target bit rate: +/-5%
- Responsiveness to new bit rate request: 60ms
- + Media content: Foreman video sequence
- + Media timeline:
  - Start time: 0s.
  - End time: 59s.
- + Media startup behaviour: [200 Kbps, 1500 Kbps].
- \* Competing traffic:
  - + Number of sources : Zero (0)
- o Test specific setup: None

#### 4.8. Media Pause and Resume

In this test case, more than one real-time interactive media flows share the link bandwidth and all flows reach to a steady state by utilizing the link capacity in an optimum way. At these stage one of the media flow is paused for a moment. This event will result in more available bandwidth for the rest of the flows and as they are on a shared link. When the paused media flow will resume it would no longer have the same bandwidth share on the link. It has to make it way through the other existing flows in the link to achieve a fair share of the link capacity. This test case is important specially for real-time interactive media which consists of more than one media flows and can pause/resume media flow at any point of time during the session. This test case directly addresses the requirement number 1.B in [I-D.ietf-rmcat-cc-requirements]. One can think it as a variation of test case 4.3. However, it is different as the candidate algorithms can use different strategies to increase it s efficiency, for example the fairness, convergence time, reduce oscillation etc, by capitalizing the fact that they have previous information of the link.

To evaluate the performance of the candidate algorithms it is expected to log enough information to visualize the following metrics at a fine enough time granularity:

1. Flow level:
  - a. End-to-end delay.
  - b. RTP packet losses observed at the receiving endpoint.
  - c. Variation in sending bit rate and goodput. Mainly observing the frequency and magnitude of oscillations.
  - d. Convergence time.
  - e. Feedback message overhead.
2. Transport level:
  - a. Bandwidth utilization.
  - b. Queue length (ms):
    - + average over the length of the session
    - + 5 and 95 percentile

Testbed Topology: Same as test case 4.3 defined in Section 4.3

Testbed attributes: The general description of the testbed parameters are same as test case 4.3 with changes in the test specific setup as below-

- o Other test specific setup:
  - \* Media flow timeline:
    - + Flow no: One (1)
    - + Start time: 0s
    - + Flow duration: 59s
    - + Pause time: not required
    - + Resume time: not required
  - \* Media flow appearance:
    - + Flow no: Two (2)
    - + Start time: 0s

- + Flow duration: 59s
- + Pause time: 20s
- + Resume time: 30s
- \* Media flow appearance:
  - + Flow no: Three (3)
  - + Start time: 0s
  - + Flow duration: 59s
  - + Pause time: not required
  - + Resume time: not required

#### 4.9. Explicit Congestion Notification Usage

TBD

#### 5. Wireless Access Links

##### 5.1. Cellular Network Specific Test Cases

Additional cellular network specific test cases are define in  
[I-D.draft-sarker-rmcat-cellular-eval-test-cases]

##### 5.2. Wi-Fi Network Specific Test Cases

TBD

[Editor's Note: We should encourage people to come up with possible  
WiFi Network specific test cases]

#### 6. Security Considerations

Security issues have not been discussed in this memo.

#### 7. IANA Considerations

There are no IANA impacts in this memo.

#### 8. Acknowledgements

Much of this document is derived from previous work on congestion  
control at the IETF.



The content and concepts within this document are a product of the discussion carried out in the Design Team.

## 9. References

### 9.1. Normative References

- [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.
- [RFC3611] Friedman, T., Caceres, R., and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.
- [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.
- [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.
- [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.
- [I-D.ietf-avtcore-rtp-circuit-breakers] Perkins, C. and V. Singh, "RTP Congestion Control: Circuit Breakers for Unicast Sessions", draft-ietf-avtcore-rtp-circuit-breakers-01 (work in progress), October 2012.
- [I-D.ietf-rmcat-eval-criteria] Singh, V. and J. Ott, "Evaluating Congestion Control for Interactive Real-time Media", draft-ietf-rmcat-eval-criteria-00 (work in progress), January 2014.
- [I-D.ietf-rmcat-cc-requirements] Jesup, R., "Congestion Control Requirements For RMCAT", draft-ietf-rmcat-cc-requirements-00 (work in progress), July 2013.
- [I-D.draft-sarker-rmcat-cellular-eval-test-cases]

Sarker, Z., "Evaluation Test Cases for Interactive Real-Time Media over Cellular Networks", , <<http://www.ietf.org/id/draft-sarker-rmcat-cellular-eval-test-cases-00.txt>>.

## 9.2. Informative References

- [I-D.ietf-rtcweb-use-cases-and-requirements]  
Holmberg, C., Hakansson, S., and G. Eriksson, "Web Real-Time Communication Use-cases and Requirements", draft-ietf-rtcweb-use-cases-and-requirements-10 (work in progress), December 2012.
- [RFC5033] Floyd, S. and M. Allman, "Specifying New Congestion Control Algorithms", BCP 133, RFC 5033, August 2007.
- [RFC5166] Floyd, S., "Metrics for the Evaluation of Congestion Control Mechanisms", RFC 5166, March 2008.
- [RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.
- [SA4-EVAL]  
R1-081955, 3GPP., "LTE Link Level Throughput Data for SA4 Evaluation Framework", 3GPP R1-081955, 5 2008.
- [SA4-LR] S4-050560, 3GPP., "Error Patterns for MBMS Streaming over UTRAN and GERAN", 3GPP S4-050560, 5 2008.
- [xiph-seq]  
Xiph.org, , "Video Test Media",  
<http://media.xiph.org/video/derf/> , .
- [HEVC-seq]  
HEVC, , "Test Sequences",  
[http://www.netlab.tkk.fi/~varun/test\\_sequences/](http://www.netlab.tkk.fi/~varun/test_sequences/) , .
- [TCP-eval-suite]  
Lachlan, A., Marcondes, C., Floyd, S., Dunn, L., Guillier, R., Gang, W., Eggert, L., Ha, S., and I. Rhee, "Towards a Common TCP Evaluation Suite", Proc. PFLDnet. 2008, August 2008.

## Authors' Addresses

Zaheduzzaman Sarker  
Ericsson AB  
Luleae, SE 977 53  
Sweden

Phone: +46 10 717 37 43  
Email: zaheduzzaman.sarker@ericsson.com

Varun Singh  
Aalto University  
School of Electrical Engineering  
Otakaari 5 A  
Espoo, FIN 02150  
Finland

Email: varun@comnet.tkk.fi  
URI: <http://www.netlab.tkk.fi/~varun/>

Xiaoqing Zhu  
Cisco Systems  
510 McCarthy Blvd  
Milpitas, CA 95134  
USA

Email: xiaoqzhu@cisco.com

Michael A. Ramalho  
Cisco Systems, Inc.  
8000 Hawkins Road  
Sarasota, FL 34241  
USA

Phone: +1 919 476 2038  
Email: mramalho@cisco.com

RMCAT WG  
Internet-Draft  
Intended status: Informational  
Expires: September 4, 2014

M. Zanaty  
Cisco  
V. Singh  
Aalto University  
S. Nandakumar  
Cisco  
March 3, 2014

RTP Application Interaction with Congestion Control  
draft-zanaty-rmcat-app-interaction-00

Abstract

Interactive real-time media applications that use the Real-time Transport Protocol (RTP) over the User Datagram Protocol (UDP) must use congestion control techniques above the UDP layer since it provides none. This memo describes the interactions and interfaces necessary between the application components that relate to congestion control, including the RTP layer, the higher-level media codec control layer, and the lower-level transport interface and the congestion control layer.

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 September 4, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|                                               |    |
|-----------------------------------------------|----|
| 1. Introduction . . . . .                     | 2  |
| 2. Key Words for Requirements . . . . .       | 3  |
| 3. Conceptual Model . . . . .                 | 4  |
| 4. Implementation Model . . . . .             | 5  |
| 5. Interfaces and Interactions . . . . .      | 6  |
| 5.1. Config - Codec Interactions . . . . .    | 6  |
| 5.2. Codec - RTP Interactions . . . . .       | 6  |
| 5.3. Codec - CC Interactions . . . . .        | 7  |
| 5.4. RTP - CC Interactions . . . . .          | 9  |
| 5.5. CC - UDP Interactions . . . . .          | 9  |
| 5.6. CC - Shared State Interactions . . . . . | 10 |
| 6. Acknowledgements . . . . .                 | 10 |
| 7. IANA Considerations . . . . .              | 10 |
| 8. Security Considerations . . . . .          | 10 |
| 9. References . . . . .                       | 11 |
| 9.1. Normative References . . . . .           | 11 |
| 9.2. Informative References . . . . .         | 12 |
| Authors' Addresses . . . . .                  | 13 |

## 1. Introduction

Interactive real-time media applications most commonly use RTP [RFC3550] over UDP [RFC0768]. Since UDP provides no form of congestion control, which is essential for any application deployed on the Internet, these RTP applications have historically implemented one of the following options at the application layer to address their congestion control requirements.

1. For media with relatively low packet rates and bit rates, such as many speech codecs, some applications use a simple form of congestion control that stops transmission permanently or temporarily after observing significant packet loss over a significant period of time, similar to the RTP circuit breakers [I-D.ietf-avtcore-rtp-circuit-breakers].
2. Some applications have no explicit congestion control, despite the clear requirements in RTP and its profiles AVP [RFC3551] and AVPF [RFC4585], under the expectation that users will terminate

media flows that are significantly impaired by congestion (in essence, human circuit breakers).

3. For media with substantially higher packet rates and bit rates, such as many video codecs, various non-standard congestion control techniques are often used to adapt transmission rate based on receiver feedback.
4. Some experimental applications use standardized techniques such as TCP-Friendly Rate Control (TFRC) [RFC5348]. However, for various reasons, these have not been widely deployed.

The RTP Media Congestion Avoidance Techniques (RMCAT) working group was chartered to standardize appropriate and effective congestion control for RTP applications. It is expected such applications will migrate from the above historical solutions to the RMCAT solution(s).

The RMCAT requirements [I-D.ietf-rmcat-cc-requirements] include low delay, reasonably high throughput, fast reaction to capacity changes including routing or interface changes, stability without over-reaction or oscillation, fair bandwidth sharing with other instances of itself and TCP flows, sharing information across multiple flows when possible [I-D.welzl-rmcat-coupled-cc], and performing as well or better in networks which support Active Queue Management (AQM), Explicit Congestion Notification (ECN), or Differentiated Services Code Points (DSCP).

In order to meet these requirements, interactions are necessary between the application's congestion controller, the RTP layer, media codecs, other components, and the OS's UDP stack. This memo discusses these interactions, presents a conceptual model of the required interfaces based on a simplified application decomposition, and proposes specific information exchange across these interfaces along with corresponding component behavior.

Note that RTP can also operate over other transports with integrated congestion control such as TCP [RFC5681] and DCCP [RFC4340], but that is beyond the scope of RMCAT and this memo.

## 2. Key Words for Requirements

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

### 3. Conceptual Model

It is useful to decompose an RTP application into several components to facilitate understanding and discussion of where congestion control functions operate, and how they interface with the other components. The conceptual model in Figure 1 consists of the following components.

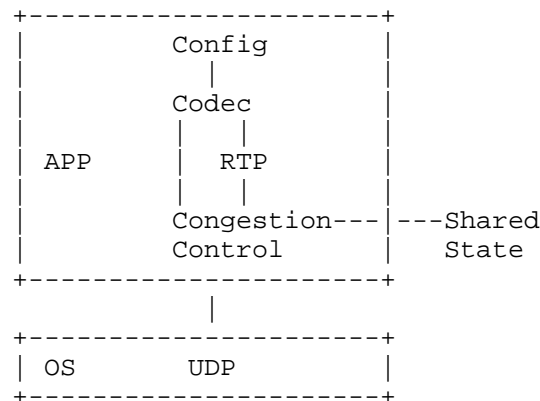


Figure 1

- o APP: Application containing one or more RTP streams and the corresponding media codecs and congestion controllers. For example, a WebRTC browser.
- o Config: Media and transport configuration specified by the application (for example, using the constraints API in a WebRTC Javascript application), after media and transport parameters have been negotiated with the remote peer (for example, using SDP or other protocols). This determines the initial static configuration negotiated in session establishment. The dynamic state may differ due to congestion or other factors, but still must conform to limits established in the config.
- o Codec: Media encoder/decoder or other source/sink for the RTP payload. The codec may be, for example, a simple monaural audio format, a complex scalable video codec with several dependent layers, or a source/sink with no live encoding/decoding such as a mixer which selectively switches or forwards streams rather than mixes media.
- o RTP: Standard RTP stack functions, including RTCP, but excluding possible new extensions specific to congestion control (CC) such as circuit breakers [I-D.ietf-avtcore-rtp-circuit-breakers], TMMBR

[RFC5104] or REMB [I-D.alvestrand-remcat-remb] (for receiver-side CC), ACK/NACK [RFC4585] (for sender-side CC), absolute timestamps (for sender-side or receiver-side CC), etc.

- o Congestion Control: All functions directly responsible for congestion control, including a possible new RTP/RTCP extensions, send rate computation (for sender-side CC), receive rate computation (for receiver-side CC), other statistics, and control of the UDP sockets.
- o Shared State: Storage and exchange of congestion control state for multiple flows within the application and beyond it.
- o OS: Operating System containing the UDP socket interface and other network functions such as ECN, DSCP, physical interface events, interface-level traffic shaping, etc.

#### 4. Implementation Model

There are advantages and drawbacks to implementing congestion control in the application layer. It avoids OS dependencies and allows for rapid experimentation, evolution and optimization for each application. However, it also puts the burden on all applications, which raises the risks of improper or divergent implementations. One motivation of this memo is to mitigate such risks by giving proper guidance on how the application components relating to congestion control should interact.

Another drawback of congestion control in the application layer is that any decomposition, including the one presented in Figure 1, is purely conceptual and illustrative, since implementations have differing designs and decompositions. Conversely, this can be viewed as an advantage to distribute congestion control functions wherever expedient without rigid interfaces. For example, they may be distributed within the RTP stack itself, so the separate components in Figure 1 are combined into a single RTP+CC component as shown in Figure 2.



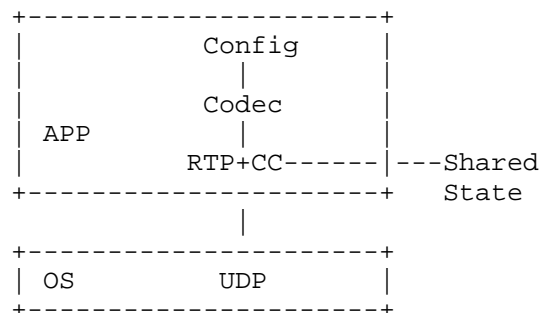


Figure 2

The conceptual model in Figure 1 will be used throughout this memo to establish clearer boundaries between functions. But actual implementations may be closer to the looser model in [Singh12].

## 5. Interfaces and Interactions

### 5.1. Config - Codec Interactions

The primary interactions between the config and the codec that are relevant to congestion control are the multiplexing of media streams [I-D.ietf-mmusic-sdp-bundle-negotiation] and RTP/RTCP [RFC5761] on the same UDP port, and the negotiated RTCP attributes such as reduced size [RFC5506], codec control [RFC5104], transmission time [RFC5450], etc.

The config also establishes limits for the codec such as maximum bit rate and other codec-specific parameters. For example, a video codec config often sets limits on maximum resolution and frame rate as well as bit rate.

### 5.2. Codec - RTP Interactions

Packetization of codec frames into RTP packets can be an important interaction. Some network interfaces may benefit from small packet sizes well below the MTU, while others may benefit from large packets approaching the MTU. Equalizing packet sizes of a frame may also be beneficial in some cases, rather than a combination of large and small packets. For example, in some FEC schemes, the FEC bandwidth overhead depends on the largest source packet size. Equalizing the source packet sizes can yield lower overhead than a combination of large and small packets.

### 5.3. Codec - CC Interactions

Allowed Rate (from CC to Codec): The max transmit rate allowed over the next time interval. The time interval may be specified or may use a default, for example, one second. The rate may be specified in bytes or packets or both. The rate must never exceed permanent limits established in session signaling such as the SDP bandwidth attribute [RFC4566] nor temporary limits in RTCP such as TMMBR [RFC5104] or REMB [I-D.alvestrand-rmcat-remb]. This is the most important interface among all components, and is always required in any RMCAT solution. In the simplest possible solution, it may be the only CC interface required.

Media Elasticity (from Codec to CC): Many live media encoders are highly elastic, often able to achieve any target bit rate within a wide range, by adapting the media quality. For example, a video encoder may support any bit rate within a range of a few tens or hundreds of kbps up to several Mbps, with rate changes registering as fast as the next video frame, although there may be limitations in the frequency of changes. Other encoders may be less elastic, supporting a narrower rate range, coarser granularity of rate steps, slower reaction to rate changes, etc. Other media, particularly some audio codecs, may be fully inelastic with a single fixed rate. CC can beneficially use codec elasticity, if provided, to plan Allowed Rate changes, especially when there are multiple flows sharing CC state and bandwidth.

Startup Ramp (from Codec to CC, and from CC to Codec): Startup is an important moment in a conversation. Rapid rate adaptation during startup is therefore important. The codec should minimize its startup media rate as much as possible without adversely impacting the user experience, and support a strategy for rapid rate ramp. The CC should allow the highest startup media rate as possible without adversely impacting network conditions, and also support rapid rate ramp until stabilizing on the available bandwidth. Startup can be viewed as a negotiation between the codec and the CC. The codec requests a startup rate and ramp, and the CC responds with the allowable parameters which may be lower/slower. The RMCAT requirements also include the possibility of bandwidth history to further accelerate or even eliminate startup ramp time. While this is highly desirable from an application viewpoint, it may be less acceptable to network operators, since it is in essence a gamble on current congestion state matching historical state, with the potential for significant congestion contribution if the gamble was wrong. Note that startup can often commence before user interaction or conversation to reduce the chance of clipped media.

Delay Tolerance (from Codec to CC): An ideal CC will always minimize delay and target zero. However, real solutions often need a real non-zero delay tolerance. The codec should provide an absolute delay tolerance, perhaps expressed as an impairment factor to mix with other metrics.

Loss Tolerance (from Codec to CC): An ideal CC will always minimize packet loss and target zero. However, real solutions often need a real non-zero loss tolerance. The codec should provide an absolute loss tolerance, perhaps expressed as an impairment factor to mix with other metrics. Note this is unrecoverable post-repair loss after retransmission or forward error correction.

Throughput Sensitivity (from Codec to CC): An ideal CC will always maximize throughput. However, real solutions often need a trade-off between throughput and other metrics such as delay or loss. The codec should provide throughput sensitivity, perhaps expressed as an impairment factor (for low throughputs) to mix with other metrics.

Rate Stability (from Codec to CC): The CC algorithm must strike a balance between rate stability and fast reaction to changes in available bandwidth. The codec should provide its preference for rate stability versus fast and frequent reaction to rate changes, perhaps expressed as an impairment factor (for high rate variance over short timescales) to mix with other metrics.

Forward Error Correction (FEC): Simple FEC schemes like XOR Parity codes [RFC5109] may not handle consecutive or burst loss well. More complex FEC schemes like Reed-Solomon [RFC6865] or Raptor [RFC6330] codes are more effective at handling bursty loss. The sensitivity to packet loss therefore depends on the media (source) encoding as well as the FEC (channel) encoding, and this sensitivity may differ for different loss patterns like random, periodic, or consecutive loss. Expressing this sensitivity to the congestion controller may help it choose the right balance between optimizing for throughput versus low loss.

Probing for Available Bandwidth: FEC can also be used to probe for additional available bandwidth, if the application desires a higher target rate than the current rate. FEC is preferable to synthetic probes since any contribution to congestion by the FEC probe will not impact the post-repair loss rate of the source media flow while synthetic probes may adversely affect the loss rate [Nagy14].

#### 5.4. RTP - CC Interactions

**RTP Circuit Breakers:** The intent behind RTP circuit breakers [I-D.ietf-avtcore-rtp-circuit-breakers] is to provide a kill switch of last resort, not true congestion control. The breakers should never trip when an effective congestion control is operating. This may impose some boundaries on RMCAT solutions to ensure the congestion control never approaches situations which may trigger the breakers.

**RTCP Feedback:** The primary method of communicating CC information is RTCP.

**RTP Header Extensions:** While RTCP is likely to be the primary carrier of CC feedback, the RMCAT requirements also include the possibility of using RTP header extensions in bidirectional flows for CC feedback. Transmission time [RFC5450], or possibly absolute time, also use header extensions, as would any per packet priority markings expected to survive across different networks and administrative domains.

#### 5.5. CC - UDP Interactions

**Pacing / Shaping:** Simple pacing / shaping strategies delay the transmission of packets to equalize inter-packet time intervals, assuming the bottleneck is most sensitive to packet rate. More complex pacing strategies may go beyond simple even distribution of transmission times. For example, Sprout [Winstein13] attempts to predict the optimal transmission time (and rate) with the highest probability of success for each packet based on channel statistics. Pacing may be always on, or adaptively enabled / disabled based on congestion state to minimize delay. Pacing may be performed within the CC for a single flow or across multiple flows. It may also be performed across all or selective traffic over the network interface if the OS supports interface-level traffic shaping.

**Detection of Transport Capabilities:** The CC can query the OS for useful transport capabilities such as ECN, DSCP, traffic shaping, etc. This may also aid upper layers in making better decisions such as whether or not to multiplex media streams. For example, if audio can be given differentiated network treatment from video when using separate ports.

**ECN:** If the OS and transport path support ECN, the CC can react faster than a loss-based CC and more reliably to congestion onset and abatement.

DSCP: If the OS and transport path support DSCP, the CC can map per-packet priority from RTP header extensions to DSCP (and layer 2 QoS if available) for better network handling under congestion.

AQM: If AQM is present in the bottleneck, and working effectively, there should be little or no excess delay observed when varying the transmission rate. The loss of such delay signals may hinder the performance of congestion control algorithms that are highly dependent on delay variation for adapting transmission rate. If the application has knowledge of the presence of AQM, through any means which are beyond the scope of this memo, it should communicate this to the CC. The CC may use this to alter its signal collection and rate adaptation strategies. The CC must not rely solely on this as a reliable indicator. It must continue to monitor statistics to validate this application hint, and react appropriately if the statistics suggest different network behavior.

#### 5.6. CC - Shared State Interactions

Multiple Flows: Sharing state across multiple flows within the application can yield better CC decisions. Sharing state across even more flows beyond the application can yield even better CC decisions. The actual benefits and mechanisms of state sharing and coupled CC are described in [I-D.welzl-rmc-cat-coupled-cc].

Weighted Fairness: An important consideration in CC of multiple flows is their relative application-specified weights. Within an application, it is likely the different flows have different rate requirements, so equal bandwidth sharing may not be fair nor desirable, and weighted fairness may be required.

#### 6. Acknowledgements

The RMCAT design team discussions contributed to this memo.

#### 7. IANA Considerations

This memo includes no request to IANA.

#### 8. Security Considerations

Amplification attacks often use UDP traffic to launch denial of service attacks. Attackers may attempt to subvert congestion control protocols in UDP applications to launch amplification attacks by signaling more bandwidth than is actually available. For example, sending a victim a forged REMB or a few fast ACKs may result in the victim sending a high rate RTP stream. Attacks on conference servers could lead to further amplification if it distributes the streams to

many others. One mitigation is to use SRTCP for congestion control messages where supported. Even if SRTCP is only authenticated not encrypted, SRTCP packets should always pass authentication checks before any message contents are interpreted. Non-secure RTCP should be avoided where possible.

## 9. References

### 9.1. Normative References

- [I-D.alvestrand-rmcat-remb]  
Alvestrand, H., "RTCP message for Receiver Estimated Maximum Bitrate", draft-alvestrand-rmcat-remb-03 (work in progress), October 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-05 (work in progress), February 2014.
- [I-D.ietf-mmusic-sdp-bundle-negotiation]  
Holmberg, C., Alvestrand, H., and C. Jennings, "Multiplexing Negotiation Using Session Description Protocol (SDP) Port Numbers", draft-ietf-mmusic-sdp-bundle-negotiation-05 (work in progress), October 2013.
- [I-D.ietf-rmcat-cc-requirements]  
Jesup, R., "Congestion Control Requirements For RMCAT", draft-ietf-rmcat-cc-requirements-02 (work in progress), February 2014.
- [I-D.ietf-rmcat-eval-criteria]  
Singh, V. and J. Ott, "Evaluating Congestion Control for Interactive Real-time Media", draft-ietf-rmcat-eval-criteria-00 (work in progress), January 2014.
- [I-D.welzl-rmcat-coupled-cc]  
Welzl, M., Islam, S., and S. Gjessing, "Coupled congestion control for RTP media", draft-welzl-rmcat-coupled-cc-02 (work in progress), October 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [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.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, 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.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.
- [RFC5450] Singer, D. and H. Desineni, "Transmission Time Offsets in RTP Streams", RFC 5450, March 2009.
- [RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.
- [RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, April 2010.

## 9.2. Informative References

- [Nagy14] Nagy, M., Singh, V., Ott, J., and L. Eggert, "Congestion Control using FEC for Conversational Multimedia Communication", Proc. of 5th ACM International Conference on Multimedia Systems , 3 2014.
- [RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.
- [RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, March 2006.
- [RFC5109] Li, A., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, December 2007.
- [RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, September 2008.
- [RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

- [RFC6330] Luby, M., Shokrollahi, A., Watson, M., Stockhammer, T., and L. Minder, "RaptorQ Forward Error Correction Scheme for Object Delivery", RFC 6330, August 2011.
- [RFC6865] Roca, V., Cunche, M., Lacan, J., Bouabdallah, A., and K. Matsuzono, "Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME", RFC 6865, February 2013.
- [Singh12] Singh, V., Ott, J., and C. Perkins, "Congestion Control for Interactive Media: Control Loops & APIs", Proc. of IAB /IRTF Workshop on Congestion Control for Interactive RTC , 7 2012.
- [Winstein13] Winstein,, K., Sivaraman,, A., and H. Balakrishnan, "Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks", Proc. of the 10th USENIX Symposium on Networked Systems Design and Implementation , 4 2013.

## Authors' Addresses

Mo Zanaty  
Cisco  
Raleigh, NC  
USA

Email: mzanaty@cisco.com

Varun Singh  
Aalto University  
Espoo, FIN  
Finland

Email: varun@comnet.tkk.fi

Suhas Nandakumar  
Cisco  
San Jose, CA  
USA

Email: snandaku@cisco.com