RMCAT WG
Internet-Draft
Intended status: Informational
Expires: September 20, 2020

V. Singh
callstats.io
J. Ott
Technical University of Munich
S. Holmer
Google

March 19, 2020

Evaluating Congestion Control for Interactive Real-time Media
draft-ietf-rmcat-eval-criteria-14

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 https://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 20, 2020.

Copyright Notice

Copyright (c) 2020 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 (https://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

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 congestion control algorithms are expected to operate within the envelope of the circuit breakers defined in RFC8083 [RFC8083].

This document only provides the broad set of network parameters and and traffic models for evaluating a new congestion control algorithm. The minimal requirements for congestion control proposals is to produce or present results for the test scenarios described in [I-D.ietf-rmcat-eval-test] (Basic Test Cases), which also defines the specifics for the test cases. Additionally, proponents may produce evaluation results for the wireless test scenarios [I-D.ietf-rmcat-wireless-tests].

This document does not cover application-specific implications of congestion control algorithms and how those could be evaluated. Therefore, no quality metrics are defined for performance evaluation; quality metrics and algorithms to infer those vary between media types. Metrics and algorithms to assess, e.g., quality of experience evolve continuously so that determining suitable choices is left for future work. However, there is consensus that each congestion control algorithm should be able to show that it is useful for interactive video by performing analysis using a real codecs and video sequences and state-of-the-art quality metrics.

Beyond optimizing individual metrics, real-time applications may have further options to trade off performance, e.g., across multiple media; refer to the RMCAT requirements [I-D.ietf-rmcat-cc-requirements] document. Such trade-offs may be defined in the future.

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

This document specifies testing criteria for evaluating congestion control algorithms for RTP media flows. Proposed algorithms are to prove their performance by means of simulation and/or emulation experiments for all the cases described.

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 [tcpdump], wireshark [wireshark]). The following metrics are calculated based on the information in the packet logs:

1. Sending rate, Receiver rate, Goodput (measured at 200ms intervals)

2. Packets sent, Packets received

3. Bytes sent, bytes received

4. Packet delay

5. Packets lost, Packets discarded (from the playout or de-jitter buffer)

6. If using, retransmission or FEC: post-repair loss

7. Self-Fairness and Fairness with respect to cross traffic: Experiments testing a given congestion control proposal must report on relative ratios of the average throughput (measured at coarser time intervals) obtained by each RTP media stream. In the presence of background cross-traffic such as TCP, the report must also include the relative ratio between average throughput of RTP media streams and cross-traffic streams.
   During static periods of a test (i.e., when bottleneck bandwidth is constant and no arrival/departure of streams), these report on relative ratios serve as an indicator of how fair the RTP streams share bandwidth amongst themselves and against cross-traffic streams. The throughput measurement interval should be set at a few values (for example, at 1s, 5s, and 20s) in order to measure fairness across different time scales.
   As a general guideline, the relative ratio between congestion controlled RTP flows with the same priority level and similar

path RTT should be bounded between (0.333 and 3.) For example, see the test scenarios described in [I-D.ietf-rmcat-eval-test].

8. 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.

9. Instability or oscillation in the sending rate: The frequency or number of instances when the sending rate oscillates between an high watermark level and a low watermark level, or vice-versa in a defined time window. For example, the watermarks can be set at 4x interval: 500 Kbps, 2 Mbps, and a time window of 500ms.

10. Bandwidth Utilization, defined as ratio of the instantaneous sending rate to the instantaneous bottleneck capacity. This metric is useful only when a congestion controlled RTP flow is by itself or competing with similar cross-traffic.

Note that the above metrics are all objective application-independent metrics. Refer to Section 3, in [I-D.ietf-netvc-testing] for objective metrics for evaluating codecs.

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

Having a common log format simplifies running analyses across and comparing different measurements. The log file should be tab or comma separated containing the following details:

```
Send or receive timestamp (Unix): <int>.<int>  -- sec.usec decimal
RTP payload type                  <int>        -- decimal
SSRC                              <int>        -- hexadecimal
RTP sequence no                   <int>        -- decimal
RTP timestamp                     <int>        -- decimal
marker bit                        0|1          -- character
Payload size                      <int>        -- # bytes, decimal
```

Each line of the log file should be terminated with CRLF, CR, or LF characters. Empty lines are disregarded.

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).

These data should suffice to compute the media-encoding independent metrics described above. Use of a common log will allow simplified post-processing and analysis across different implementations.

4. List of Network Parameters

The implementors initially are encouraged to choose evaluation settings from the following values:

4.1. One-way Propagation Delay

Experiments are expected to verify that the congestion control is able to work across a broad range of path characteristics, also including challenging situations, for example over trans-continental and/or satellite links. Tests thus account for the following different latencies:

1. Very low latency: 0-1ms

2. Low latency: 50ms

3. High latency: 150ms

4. Extreme latency: 300ms

4.2. End-to-end Loss

Many paths in the Internet today are largely lossless but, with wireless networks and interference, towards remote regions, or in scenarios featuring high/fast mobility, media flows may exhibit substantial packet loss. This variety needs to be reflected appropriately by the tests.

To model a wide range of 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%

4.3. Drop Tail Router Queue Length

Routers should be configured to use Drop Trail queues in the experiments due to their (still) prevalent nature. Experimentation with AQM schemes is encouraged but not mandatory.

The router queue length is measured as the time taken to drain the FIFO queue. It has been noted in various discussions that the queue length in the current deployed Internet varies significantly. While the core backbone network has very short queue length, the home gateways usually have larger queue length. Those various queue lengths can be categorized in the following way:

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

2. Nominal: 300-500ms

3. Buffer-bloated: 1000-2000ms

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

QueueSize (in bytes) = QueueSize (in sec) x Throughput (in bps)/8

4.4. Loss generation model

Many models for generating packet loss are available, some yield correlated, others independent losses; losses can also be extracted from packet traces. As a (simple) minimum loss model with minimal parameterization (i.e., the loss rate), independent random losses must be used in the evaluation.

It is known that independent loss models may reflect reality poorly and hence more sophisticated loss models could be considered. Suitable models for correlated losses includes the Gilbert-Elliot model [gilbert-elliott] and losses generated by modeling a queue including its (different) drop behaviors.

4.5. Jitter models

This section defines jitter models for the purposes of this document. When jitter is to be applied to both the congestion controlled RTP flow and any competing flow (such as a TCP competing flow), the competing flow will use the jitter definition below that does not allow for re-ordering of packets on the competing flow (see NR-RBPDV definition below).

Jitter is an overloaded term in communications. It is typically used to refer to the variation of a metric (e.g., delay) with respect to some reference metric (e.g., average delay or minimum delay). For example, RFC 3550 jitter is computed as the smoothed difference in packet arrival times relative to their respective expected arrival times, which is particularly meaningful if the underlying packet delay variation was caused by a Gaussian random process.

Because jitter is an overloaded term, we use the term Packet Delay Variation (PDV) instead to describe the variation of delay of individual packets in the same sense as the IETF IPPM WG has defined PDV in their documents (e.g., RFC 3393) and as the ITU-T SG16 has defined IP Packet Delay Variation (IPDV) in their documents (e.g., Y.1540).

Most PDV distributions in packet network systems are one-sided distributions, the measurement of which with a finite number of measurement samples results in one-sided histograms. In the usual packet network transport case, there is typically one packet that transited the network with the minimum delay; a (large) number of packets transit the network within some (smaller) positive variation from this minimum delay, and a (small) number of the packets transit the network with delays higher than the median or average transit time (these are outliers). Although infrequent, outliers can cause significant deleterious operation in adaptive systems and should be considered in rate adaptation designs for RTP congestion control.

In this section we define two different bounded PDV characteristics, 1) Random Bounded PDV and 2) Approximately Random Subject to No-Reordering Bounded PDV.

The former, 1) Random Bounded PDV is presented for information only, while the latter, 2) Approximately Random Subject to No-Reordering Bounded PDV, must be used in the evaluation.

4.5.1. Random Bounded PDV (RBPDV)

The RBPDV probability distribution function (PDF) is specified to be of some mathematically describable function which includes some practical minimum and maximum discrete values suitable for testing. For example, the minimum value, x_min, might be specified as the minimum transit time packet and the maximum value, x_max, might be defined to be two standard deviations higher than the mean.

Since we are typically interested in the distribution relative to the mean delay packet, we define the zero mean PDV sample, z(n), to be z(n) = x(n) - x_mean, where x(n) is a sample of the RBPDV random variable x and x_mean is the mean of x.

We assume here that s(n) is the original source time of packet n and the post-jitter induced emission time, j(n), for packet n is:

j(n) = {[z(n) + x_mean] + s(n)}.

It follows that the separation in the post-jitter time of packets n and n+1 is {[s(n+1)-s(n)] - [z(n)-z(n+1)]}. Since the first term is always a positive quantity, we note that packet reordering at the receiver is possible whenever the second term is greater than the first. Said another way, whenever the difference in possible zero mean PDV sample delays (i.e., [x_max-x_min]) exceeds the inter-departure time of any two sent packets, we have the possibility of packet re-ordering.

There are important use cases in real networks where packets can become re-ordered such as in load balancing topologies and during route changes. However, for the vast majority of cases there is no packet re-ordering because most of the time packets follow the same path. Due to this, if a packet becomes overly delayed, the packets after it on that flow are also delayed. This is especially true for mobile wireless links where there are per-flow queues prior to base station scheduling. Owing to this important use case, we define another PDV profile similar to the above, but one that does not allow for re-ordering within a flow.

4.5.2. Approximately Random Subject to No-Reordering Bounded PDV (NR-RPVD)

No Reordering RPDV, NR-RPVD, is defined similarly to the above with one important exception. Let serial(n) be defined as the serialization delay of packet n at the lowest bottleneck link rate (or other appropriate rate) in a given test. Then we produce all the post-jitter values for j(n) for n = 1, 2, ... N, where N is the length of the source sequence s to be offset-ed. The exception can be stated as follows: We revisit all j(n) beginning from index n=2, and if j(n) is determined to be less than [j(n-1)+serial(n-1)], we redefine j(n) to be equal to [j(n-1)+serial(n-1)] and continue for all remaining n (i.e., n = 3, 4, .. N). This models the case where the packet n is sent immediately after packet (n-1) at the bottleneck link rate. Although this is generally the theoretical minimum in that it assumes that no other packets from other flows are in-between packet n and n+1 at the bottleneck link, it is a reasonable assumption for per flow queuing.

We note that this assumption holds for some important exception cases, such as packets immediately following outliers. There are a multitude of software controlled elements common on end-to-end Internet paths (such as firewalls, ALGs and other middleboxes) which stop processing packets while servicing other functions (e.g., garbage collection). Often these devices do not drop packets, but rather queue them for later processing and cause many of the outliers. Thus NR-RPVD models this particular use case (assuming serial(n+1) is defined appropriately for the device causing the outlier) and thus is believed to be important for adaptation development for congestion controlled RTP streams.

4.5.3. Recommended distribution

Whether Random Bounded PDV or Approximately Random Subject to No-Reordering Bounded PDV, it is recommended that z(n) is distributed according to a truncated Gaussian for the above jitter models:

$$z(n) \sim |\max(\min(N(0, std^2), N\_STD * std), -N\_STD * std)|$$

where N(0, std^2) is the Gaussian distribution with zero mean and standard deviation std. Recommended values:

- o std = 5 ms
- o N_STD = 3

5. Traffic Models

5.1. TCP traffic model

Long-lived TCP flows will download data throughout the session and are expected to have infinite amount of data to send or receive. This roughly applies, for example, when downloading software distributions.

Each short TCP flow is modeled as a sequence of file downloads interleaved with idle periods. Not all short TCP flows start at the same time, i.e., some start in the ON state while others start in the OFF state.

The short TCP flows can be modeled as follows: 30 connections start simultaneously fetching small (30-50 KB) amounts of data, evenly distributed. This covers the case where the short TCP flows are fetching web page resources rather than video files.

The idle period between bursts of starting a group of TCP flows is typically derived from an exponential distribution with the mean value of 10 seconds.

[These values were picked based on the data available at http://httparchive.org/interesting.php as of October 2015].

Many different TCP congestion control schemes are deployed today. Therefore, experimentation with a range of different schemes, especially including CUBIC, is encouraged. Experiments must document in detail which congestion control schemes they tested against and which parameters were used.

5.2. RTP Video model

[RFC8593] describes two types of video traffic models for evaluating candidate algorithms for RTP congestion control. The first model statistically characterizes the behavior of a video encoder, whereas the second model uses video traces.

Sample video test sequences are available at [xiph-seq]. The following two video streams are the recommended minimum for testing: Foreman (CIF sequence) and FourPeople (720p); both come as raw video data to be encoded dynamically. As these video sequences are short (300 and 600 frames, respectively, they shall be stitched together repeatedly until the desired length is reached.

5.3. Background UDP

Background UDP flow is modeled as a constant bit rate (CBR) flow. It will download data at a particular CBR rate for the complete session, or will change to particular CBR rate at predefined intervals. The inter packet interval is calculated based on the CBR and the packet size (is typically set to the path MTU size, the default value can be 1500 bytes).

Note that new transport protocols such as QUIC may use UDP but, due to their congestion control algorithms, will exhibit behavior conceptually similar in nature to TCP flows above and can thus be subsumed by the above, including the division into short- and long-lived flows. As QUIC evolves independently of TCP congestion control algorithms, its future congestion control should be considered as competing traffic as appropriate.

6. Security Considerations

This document specifies evaluation criteria and parameters for assessing and comparing the performance of congestion control protocols and algorithms for real-time communication. This memo itself is thus not subject to security considerations but the protocols and algorithms evaluated may be. In particular, successful operation under all tests defined in this document may suffice for a comparative evaluation but must not be interpreted that the protocol is free of risks when deployed on the Internet as briefly described in the following by example.

Such evaluations are expected to be carried out in controlled environments for limited numbers of parallel flows. As such, these evaluations are by definition limited and will not be able to systematically consider possible interactions or very large groups of communicating nodes under all possible circumstances, so that careful protocol design is advised to avoid incidentally contributing traffic that could lead to unstable networks, e.g., (local) congestion collapse.

This specification focuses on assessing the regular operation of the protocols and algorithms under considerations. It does not suggest checks against malicious use of the protocols -- by the sender, the receiver, or intermediate parties, e.g., through faked, dropped, replicated, or modified congestion signals. It is up to the protocol specifications themselves to ensure that authenticity, integrity, and/or plausibility of received signals are checked and the appropriate actions (or non-actions) are taken.

7. IANA Considerations

There are no IANA impacts in this memo.

8. 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 Jitter model.

9. Acknowledgments

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

The authors would like to thank Harald Alvestrand, Anna Brunstrom, Luca De Cicco, Wesley Eddy, Lars Eggert, Kevin Gross, Vinayak Hegde, Randell Jesup, Mirja Kuehlewind, Karen Nielsen, Piers O'Hanlon, Colin Perkins, Michael Ramalho, Zaheduzzaman Sarker, Timothy B. Terriberry, Michael Welzl, Mo Zanaty, and Xiaoqing Zhu 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.

10. References

10.1. Normative References

[I-D.ietf-rmcat-cc-requirements]
Jesup, R. and Z. Sarker, "Congestion Control Requirements for Interactive Real-Time Media", draft-ietf-rmcat-cc-requirements-09 (work in progress), December 2014.

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

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

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

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

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, DOI 10.17487/RFC5506, April 2009, <https://www.rfc-editor.org/info/rfc5506>.

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

[RFC8593] Zhu, X., Mena, S., and Z. Sarker, "Video Traffic Models for RTP Congestion Control Evaluations", RFC 8593, DOI 10.17487/RFC8593, May 2019, <https://www.rfc-editor.org/info/rfc8593>.

10.2. Informative References

[gilbert-elliott] Hasslinger, G. and O. Hohlfeld, "The Gilbert-Elliott Model for Packet Loss in Real Time Services on the Internet", 14th GI/ITG Conference - Measurement, Modelling and Evalutation of Computer and Communication Systems , 3 2008.

[I-D.ietf-netvc-testing] Daede, T., Norkin, A., and I. Brailovskiy, "Video Codec Testing and Quality Measurement", draft-ietf-netvc-testing-09 (work in progress), January 2020.

[I-D.ietf-rmcat-eval-test] Sarker, Z., Singh, V., Zhu, X., and M. Ramalho, "Test Cases for Evaluating RMCAT Proposals", draft-ietf-rmcat-eval-test-10 (work in progress), May 2019.

[I-D.ietf-rmcat-wireless-tests] Sarker, Z., Zhu, X., and J. Fu, "Evaluation Test Cases for Interactive Real-Time Media over Wireless Networks", draft-ietf-rmcat-wireless-tests-11 (work in progress), March 2020.

[RFC5033] Floyd, S. and M. Allman, "Specifying New Congestion Control Algorithms", BCP 133, RFC 5033, DOI 10.17487/RFC5033, August 2007, <https://www.rfc-editor.org/info/rfc5033>.

[RFC5166] Floyd, S., Ed., "Metrics for the Evaluation of Congestion Control Mechanisms", RFC 5166, DOI 10.17487/RFC5166, March 2008, <https://www.rfc-editor.org/info/rfc5166>.

[tcpdump] "Homepage of tcpdump and libpcap", https://www.tcpdump.org/index.html .

[wireshark] "Homepage of Wireshark", https://www.wireshark.org .

[xiph-seq] Daede, T., "Video Test Media Set", https://media.xiph.org/video/derf/ .

Appendix A. Change Log

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

A.1. Changes in draft-ietf-rmcat-eval-criteria-07

Updated the draft according to the discussion at IETF-101.

- o Updated the discussion on fairness. Thanks to Xiaoqing Zhu for providing text.
- o Fixed a simple loss model and provided pointers to more sophisticated ones.
- o Fixed the choice of the jitter model.

A.2. Changes in draft-ietf-rmcat-eval-criteria-06

- o Updated Jitter.

A.3. Changes in draft-ietf-rmcat-eval-criteria-05

- o Improved text surrounding wireless tests, video sequences, and short-TCP model.

A.4. Changes in draft-ietf-rmcat-eval-criteria-04

- o Removed the guidelines section, as most of the sections are now covered: wireless tests, video model, etc.
- o Improved Short TCP model based on the suggestion to use httparchive.org.

A.5. Changes in draft-ietf-rmcat-eval-criteria-03

- o Keep-alive version.
- o Moved link parameters and traffic models from eval-test

A.6. Changes in draft-ietf-rmcat-eval-criteria-02

- o Incorporated fairness test as a working test.
- o Updated text on mimimum evaluation requirements.

A.7. Changes in draft-ietf-rmcat-eval-criteria-01

- Removed Appendix B.
- Removed Section on Evaluation Parameters.

A.8. Changes in draft-ietf-rmcat-eval-criteria-00

- Updated references.
- Resubmitted as WG draft.

A.9. Changes in draft-singh-rmcat-cc-eval-04

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

A.10. Changes in draft-singh-rmcat-cc-eval-03

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

A.11. Changes in draft-singh-rmcat-cc-eval-02

- Added scenario descriptions.

A.12. Changes in draft-singh-rmcat-cc-eval-01

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

Authors' Addresses

Varun Singh
CALLSTATS I/O Oy
Runeberginkatu 4c A 4
Helsinki 00100
Finland

Email: varun.singh@iki.fi
URI: https://www.callstats.io/about

Joerg Ott
Technical University of Munich
Faculty of Informatics
Boltzmannstrasse 3
Garching bei Muenchen, DE 85748
Germany

Email: ott@in.tum.de

Stefan Holmer
Google
Kungsbron 2
Stockholm 11122
Sweden

Email: holmer@google.com

Network Working Group X. Zhu
Internet-Draft S. Mena
Intended status: Informational Cisco Systems
Expires: August 23, 2019 Z. Sarker
Ericsson AB
February 19, 2019

Video Traffic Models for RTP Congestion Control Evaluations
draft-ietf-rmcat-video-traffic-model-07

Abstract

This document describes two reference video traffic models for evaluating RTP congestion control algorithms. The first model statistically characterizes the behavior of a live video encoder in response to changing requests on the target video rate. The second model is trace-driven and emulates the output of actual encoded video frame sizes from a high-resolution test sequence. Both models are designed to strike a balance between simplicity, repeatability, and authenticity in modeling the interactions between a live video traffic source and the congestion control module. Finally, the document describes how both approaches can be combined into a hybrid model.

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 https://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 23, 2019.

Copyright Notice

Copyright (c) 2019 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 (https://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

When evaluating candidate congestion control algorithms designed for real-time interactive media, it is important to account for the characteristics of traffic patterns generated from a live video encoder. Unlike synthetic traffic sources that can conform perfectly to the rate changing requests from the congestion control module, a live video encoder can be sluggish in reacting to such changes. The output rate of a live video encoder also typically deviates from the target rate due to uncertainties in the encoder rate control process.

Consequently, end-to-end delay and loss performance of a real-time media flow can be further impacted by rate variations introduced by the live encoder.

On the other hand, evaluation results of a candidate RTP congestion control algorithm should mostly reflect the performance of the congestion control module and somewhat decouple from peculiarities of any specific video codec. It is also desirable that evaluation tests are repeatable, and be easily duplicated across different candidate algorithms.

One way to strike a balance between the above considerations is to evaluate congestion control algorithms using a synthetic video traffic source model that captures key characteristics of the behavior of a live video encoder. The synthetic traffic model should also contain tunable parameters so that it can be flexibly adjusted to reflect the wide variations in real-world live video encoder behaviors. To this end, this draft presents two reference models. The first is based on statistical modeling. The second is driven by frame size and interval traces recorded from a real-world encoder. The draft also discusses the pros and cons of each approach, as well as how both approaches can be combined into a hybrid model.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Desired Behavior of A Synthetic Video Traffic Model

A live video encoder employs encoder rate control to meet a target rate by varying its encoding parameters, such as quantization step size, frame rate, and picture resolution, based on its estimate of the video content (e.g., motion and scene complexity). In practice, however, several factors prevent the output video rate from perfectly conforming to the input target rate.

Due to uncertainties in the captured video scene, the output rate typically deviates from the specified target. In the presence of a significant change in target rate, the encoder’s output frame sizes sometimes fluctuate for a short, transient period of time before the output rate converges to the new target. Finally, while most of the frames in a live session are encoded in predictive mode (i.e., P-frames in [H264]), the encoder can occasionally generate a large intra-coded frame (i.e., I-frame as defined in [H264]) or a frame partially containing intra-coded blocks in an attempt to recover from losses, to re-sync with the receiver, or during the transient period of responding to target rate or spatial resolution changes.

Hence, a synthetic video source should have the following capabilities:

- o To change bitrate. This includes the ability to change framerate and/or spatial resolution or to skip frames upon request.
- o To fluctuate around the target bitrate specified by the congestion control module.
- o To show a delay in convergence to the target bitrate.
- o To generate intra-coded or repair frames on demand.

While there exist many different approaches in developing a synthetic video traffic model, it is desirable that the outcome follows a few common characteristics, as outlined below.

- o Low computational complexity: The model should be computationally lightweight, otherwise it defeats the whole purpose of serving as a substitute for a live video encoder.
- o Temporal pattern similarity: The individual traffic trace instances generated by the model should mimic the temporal pattern of those from a real video encoder.
- o Statistical resemblance: The synthetic traffic source should match the outcome of the real video encoder in terms of statistical characteristics, such as the mean, variance, peak, and autocorrelation coefficients of the bitrate. It is also important that the statistical resemblance should hold across different time scales, ranging from tens of milliseconds to sub-seconds.
- o A wide range of coverage: The model should be easily configurable to cover a wide range of codec behaviors (e.g., with either fast or slow reaction time in live encoder rate control) and video content variations (e.g., ranging from high to low motion).

These distinct behavior features can be characterized via simple statistical modeling or a trace-driven approach. Section 5 and Section 6 provide an example of each approach, respectively. Section 7 discusses how both models can be combined together.

4. Interactions Between Synthetic Video Traffic Source and Other Components at the Sender

Figure 1 depicts the interactions of the synthetic video traffic source with other components at the sender, such as the application, the congestion control module, the media packet transport module, etc. Both reference models --- as described later in Section 5 and Section 6 --- follow the same set of interactions.

The synthetic video source dynamically generates a sequence of dummy video frames with varying size and interval. These dummy frames are processed by other modules in order to transmit the video stream over the network. During the lifetime of a video transmission session, the synthetic video source will typically be required to adapt its encoding bitrate, and sometimes the spatial resolution and frame rate.

In this model, the synthetic video source module has a group of incoming and outgoing interface calls that allow for interaction with other modules. The following are some of the possible incoming interface calls --- marked as (a) in Figure 1 --- that the synthetic video traffic source may accept. The list is not exhaustive and can be complemented by other interface calls if necessary.

- Target bitrate R_v: target bitrate request measured in bits per second (bps). Typically, the congestion control module calculates the target bitrate and updates it dynamically over time. Depending on the congestion control algorithm in use, the update requests can either be periodic (e.g., once per second), or on-demand (e.g., only when a drastic bandwidth change over the network is observed).

- Target frame rate FPS: the instantaneous frame rate measured in frames-per-second at a given time. This depends on the native camera capture frame rate as well as the target/preferred frame rate configured by the application or user.

- Target frame resolution XY: the 2-dimensional vector indicating the preferred frame resolution in pixels. Several factors govern the resolution requested to the synthetic video source over time. Examples of such factors include the capturing resolution of the native camera and the display size of the destination screen. The target frame resolution also depends on the current target bitrate R_v, since it does not make sense to pair very low spatial resolutions with very high bitrates, and vice-versa.

- o Instant frame skipping: the request to skip the encoding of one or several captured video frames, for instance when a drastic decrease in available network bandwidth is detected.

- o On-demand generation of intra (I) frame: the request to encode another I frame to avoid further error propagation at the receiver when severe packet losses are observed. This request typically comes from the error control module. It can be initiated either by the sender or by the receiver via Full Intra Request (FIR) messages as defined in [RFC5104].

An example of outgoing interface call --- marked as (b) in Figure 1 --- is the rate range [R_min, R_max]. Here, R_min and R_max are meant to capture the dynamic rate range and actual live video encoder is capable of generating given the input video content. This typically depends on the video content complexity and/or display type (e.g., higher R_max for video contents with higher motion complexity, or for displays of higher resolution). Therefore, these values will not change with R_v but may change over time if the content is changing.

```
                         +-------------+
                         |             |  dummy encoded
                         |  Synthetic  |   video frames
                         |    Video    | -------------->
                         |   Source    |
                         |             |
                         +--------+----+
                            /|\    |
                             |     |
          -------------------+     +-------------------->
             interface from           interface to
            other modules (a)       other modules (b)
```

Figure 1: Interaction between synthetic video encoder and other modules at the sender

5. A Statistical Reference Model

This section describes one simple statistical model of the live video encoder traffic source. Figure 2 summarizes the list of tunable parameters in this statistical model. A more comprehensive survey of popular methods for modeling video traffic source behavior can be found in [Tanwir2013].

Notation	Parameter Name	Example Value
R_v	Target bitrate request	1 Mbps
FPS	Target frame rate	30 Hz
tau_v	Encoder reaction latency	0.2 s
K_d	Burst duration of the transient period	8 frames
K_B	Burst frame size during the transient period	13.5 KBytes*
t0	Reference frame interval 1/FPS	33 ms
B0	Reference frame size R_v/8/FPS	4.17 KBytes
SCALE_t	Scaling parameter of the zero-mean Laplacian distribution describing deviations in normalized frame interval (t-t0)/t0	0.15
SCALE_B	Scaling parameter of the zero-mean Laplacian distribution describing deviations in normalized frame size (B-B0)/B0	0.15
R_min	minimum rate supported by video encoder type or content activity	150 Kbps
R_max	maximum rate supported by video encoder type or content activity	1.5 Mbps

* Example value of K_B for a video stream encoded at 720p and 30 frames per second, using H.264/AVC encoder.

Figure 2: List of tunable parameters in a statistical video traffic source model.

5.1. Time-damped response to target rate update

While the congestion control module can update its target bitrate request R_v at any time, the statistical model dictates that the encoder will only react to such changes tau_v seconds after a previous rate transition. In other words, when the encoder has reacted to a rate change request at time t, it will simply ignore all subsequent rate change requests until time t+tau_v.

5.2. Temporary burst and oscillation during the transient period

The output bitrate R_o during the period [t, t+tau_v] is considered to be in a transient state when reacting to abrupt changes in target rate. Based on observations from video encoder output data, the encoder reaction to a new target bitrate request can be characterized by high variations in output frame sizes. It is assumed in the model that the overall average output bitrate R_o during this transient period matches the target bitrate R_v. Consequently, the occasional burst of large frames is followed by smaller-than-average encoded frames.

This temporary burst is characterized by two parameters:

- burst duration K_d: number of frames in the burst event; and
- burst frame size K_B: size of the initial burst frame which is typically significantly larger than average frame size at steady state.

It can be noted that these burst parameters can also be used to mimic the insertion of a large on-demand I frame in the presence of severe packet losses. The values of K_d and K_B typically depend on the type of video codec, spatial and temporal resolution of the encoded stream, as well as the video content activity level.

5.3. Output rate fluctuation at steady state

The output bitrate R_o during steady state is modeled as randomly fluctuating around the target bitrate R_v. The output traffic can be characterized as the combination of two random processes denoting the frame interval t and output frame size B over time, as the two major sources of variations in the encoder output. For simplicity, the deviations of t and B from their respective reference levels are modeled as independent and identically distributed (i.i.d) random variables following the Laplacian distribution [Papoulis]. More specifically:

- Fluctuations in frame interval: the intervals between adjacent frames have been observed to fluctuate around the reference interval of t0 = 1/FPS. Deviations in normalized frame interval DELTA_t = (t-t0)/t0 can be modeled by a zero-mean Laplacian distribution with scaling parameter SCALE_t. The value of SCALE_t dictates the "width" of the Laplacian distribution and therefore the amount of fluctuation in actual frame intervals (t) with respect to the reference frame interval t0.

- Fluctuations in frame size: the output encoded frame sizes also tend to fluctuate around the reference frame size B0=R_v/8/FPS. Likewise, deviations in the normalized frame size DELTA_B = (B-B0)/B0 can be modeled by a zero-mean Laplacian distribution with scaling parameter SCALE_B. The value of SCALE_B dictates the "width" of this second Laplacian distribution and correspondingly the amount of fluctuations in output frame sizes (B) with respect to the reference target B0.

Both values of SCALE_t and SCALE_B can be obtained via parameter fitting from empirical data captured for a given video encoder. Example values are listed in Figure 2 based on empirical data presented in [IETF-Interim].

5.4. Rate range limit imposed by video content

The output bitrate R_o is further clipped within the dynamic range [R_min, R_max], which in reality are dictated by scene and motion complexity of the captured video content. In the proposed statistical model, these parameters are specified by the application.

6. A Trace-Driven Model

The second approach for modeling a video traffic source is trace-driven. This can be achieved by running an actual live video encoder on a set of chosen raw video sequences and using the encoder's output traces for constructing a synthetic video source. With this approach, the recorded video traces naturally exhibit temporal fluctuations around a given target bitrate request R_v from the congestion control module.

The following list summarizes the main steps of this approach:

1. Choose one or more representative raw video sequences.

2. Encode the sequence(s) using an actual live video encoder. Repeat the process for a number of bitrates. Keep only the sequence of frame sizes for each bitrate.

3. Construct a data structure that contains the output of the previous step. The data structure should allow for easy bitrate lookup.

4. Upon a target bitrate request R_v from the controller, look up the closest bitrates among those previously stored. Use the

frame size sequences stored for those bitrates to approximate the frame sizes to output.

5. The output of the synthetic video traffic source contains "encoded" frames with dummy contents but with realistic sizes.

In the following, Section 6.1 explains the first three steps (1-3), Section 6.2 elaborates on the remaining two steps (4-5). Finally, Section 6.3 briefly discusses the possibility to extend the trace-driven model for supporting time-varying frame rate and/or time-varying frame resolution.

6.1. Choosing the video sequence and generating the traces

The first step is a careful choice of a set of video sequences that are representative of the target use cases for the video traffic model. For the example use case of interactive video conferencing, it is recommended to choose a sequence with content that resembles a "talking head", e.g. from a news broadcast or recording of an actual video conferencing call.

The length of the chosen video sequence is a tradeoff. If it is too long, it will be difficult to manage the data structures containing the traces. If it is too short, there will be an obvious periodic pattern in the output frame sizes, leading to biased results when evaluating congestion control performance. It has been empirically determined that a sequence 2 to 4 minutes in length sufficiently avoids the periodic pattern.

Given the chosen raw video sequence, denoted S, one can use a live encoder, e.g. some implementation of [H264] or [HEVC], to produce a set of encoded sequences. As discussed in Section 3, the output bitrate of the live encoder can be achieved by tuning three input parameters: quantization step size, frame rate, and picture resolution. In order to simplify the choice of these parameters for a given target rate, one can typically assume a fixed frame rate (e.g. 30 fps) and a fixed resolution (e.g., 720p) when configuring the live encoder. See Section 6.3 for a discussion on how to relax these assumptions.

Following these simplifications, the chosen encoder can be configured to start at a constant target bitrate, then vary the quantization step size (internally via the video encoder rate controller) to meet various externally specified target rates. It can be further assumed the first frame is encoded as an I-frame and the rest are P-frames (see, e.g., [H264] for definitions of I- and P-frames). For live encoding, the encoder rate control algorithm typically does not use knowledge of frames in the future when encoding a given frame.

Given the minimum and maximum bitrates at which the synthetic codec is to operate (denoted as R_min and R_max, see Section 4), the entire range of target bitrates can be divided into n_s steps. This leads to a encoding bitrate ladder of (n_s + 1) choices equally spaced apart by the step length l = (R_max - R_min)/n_s. The following simple algorithm is used to encode the raw video sequence.

```
r = R_min
while r <= R_max do
    Traces[r] = encode_sequence(S, r, e)
    r = r + l
```

The function encode_sequence takes as input parameters, respectively, a raw video sequence (S), a constant target rate (r), and an encoder rate control algorithm (e); it returns a vector with the sizes of frames in the order they were encoded. The output vector is stored in a map structure called Traces, whose keys are bitrates and whose values are vectors of frame sizes.

The choice of a value for the number of bitrate steps n_s is important, since it determines the number of vectors of frame sizes stored in the map Traces. The minimum value one can choose for n_s is 1; the maximum value depends on the amount of memory available for holding the map Traces. A reasonable value for n_s is one that results in steps of length l = 200 kbps. The next section will discuss further the choice of step length l.

Finally, note that, as mentioned in previous sections, R_min and R_max may be modified after the initial sequences are encoded. Henceforth, for notational clarity, we refer to the bitrate range of the trace file as [Rf_min, Rf_max]. The algorithm described in the next section also covers the cases when the current target bitrate is less than Rf_min, or greater than Rf_max.

6.2. Using the traces in the synthetic codec

The main idea behind the trace-driven synthetic codec is that it mimics the rate adaptation behavior of a real live codec upon dynamic updates of the target bitrate request R_v by the congestion control module. It does so by switching to a different frame size vector stored in the map Traces when needed.

6.2.1. Main algorithm

The main algorithm for rate adaptation in the synthetic codec maintains two variables: r_current and t_current.

o The variable r_current points to one of the keys of map Traces. Upon a change in the value of R_v, typically because the congestion controller detects that the network conditions have changed, r_current is updated based on R_v as follows:

```
R_ref = min (Rf_max, max(Rf_min, R_v))

r_current = r
such that
    (r in keys(Traces)  and
     r <= R_ref  and
    (not(exists) r' in keys(Traces) such that r <r'<= R_ref))
```

o The variable t_current is an index to the frame size vector stored in Traces[r_current]. It is updated every time a new frame is due. It is assumed that all vectors stored in Traces have the same size, denoted as size_traces. The following equation governs the update of t_current:

```
if t_current < SkipFrames then
    t_current = t_current + 1
else
    t_current = ((t_current + 1 - SkipFrames)
                 % (size_traces-SkipFrames)) + SkipFrames
```

where operator % denotes modulo, and SkipFrames is a predefined constant that denotes the number of frames to be skipped at the beginning of frame size vectors after t_current has wrapped around. The point of constant SkipFrames is avoiding the effect of periodically sending a large I-frame followed by several smaller-than-average P-frames. A typical value of SkipFrames is 20, although it could be set to 0 if one is interested in studying the effect of sending I-frames periodically.

The initial value of r_current is set to R_min, and the initial value of t_current is set to 0.

When a new frame is due, its size can be calculated following one of the three cases below:

a) Rf_min <= R_v < Rf_max: the output frame size is calculated via linear interpolation of the frame sizes appearing in Traces[r_current] and Traces[r_current + 1]. The interpolation is done as follows:

```
size_lo = Traces[r_current][t_current]
size_hi = Traces[r_current + l][t_current]
distance_lo = (R_v - r_current) / l
framesize = size_hi*distance_lo + size_lo*(1-distance_lo)
```

b) R_v < Rf_min: the output frame size is calculated via scaling with respect to the lowest bitrate Rf_min in the trace file, as follows:

```
w = R_v / Rf_min
framesize = max(fs_min, factor * Traces[Rf_min][t_current])
```

c) R_v >= Rf_max: the output frame size is calculated by scaling with respect to the highest bitrate Rf_max in the trace file, as follows:

```
w = R_v / Rf_max
framesize = min(fs_max, w * Traces[Rf_max][t_current])
```

In cases b) and c), floating-point arithmetic is used for computing the scaling factor w. The resulting value of the instantaneous frame size (framesize) is further clipped within a reasonable range between fs_min (e.g., 10 bytes) and fs_max (e.g., 1MB).

6.2.2. Notes to the main algorithm

Note that the main algorithm as described above can be further extended to mimic some additional typical behaviors of a live video encoder. Two examples are given below:

o I-frames on demand: The synthetic codec can be extended to simulate the sending of I-frames on demand, e.g., as a reaction to losses. To implement this extension, the codec's incoming interface (see (a) in Figure 1) is augmented with a new function to request a new I-frame. Upon calling such function, t_current is reset to 0.

o Variable step length l between R_min and R_max: In the main algorithm, the step length l is fixed for ease of explanation. However, if the range [R_min, R_max] is very wide, it is also possible to define a set of intermediate encoding rates with variable step length. The rationale behind this modification is that the difference between 400 kbps and 600 kbps as target bitrate is much more significant than the difference between 4400 kbps and 4600 kbps. For example, one could define steps of length 200 Kbps under 1 Mbps, then steps of length 300 Kbps between 1 Mbps and 2 Mbps; 400 Kbps between 2 Mbps and 3 Mbps, and so on.

6.3. Varying frame rate and resolution

The trace-driven synthetic codec model explained in this section is relatively simple due to the choice of fixed frame rate and frame resolution. The model can be extended further to accommodate variable frame rate and/or variable spatial resolution.

When the encoded picture quality at a given bitrate is low, one can potentially decrease either the frame rate (if the video sequence is currently in low motion) or the spatial resolution in order to improve quality-of-experience (QoE) in the overall encoded video. On the other hand, if target bitrate increases to a point where there is no longer a perceptible improvement in the picture quality of individual frames, then one might afford to increase the spatial resolution or the frame rate (useful if the video is currently in high motion).

Many techniques have been proposed to choose over time the best combination of encoder quantization step size, frame rate, and spatial resolution in order to maximize the quality of live video codecs [Ozer2011][Hu2010]. Future work may consider extending the trace-driven codec to accommodate variable frame rate and/or resolution.

From the perspective of congestion control, varying the spatial resolution typically requires a new intra-coded frame to be generated, thereby incurring a temporary burst in the output traffic pattern. The impact of frame rate change tends to be more subtle: reducing frame rate from high to low leads to sparsely spaced larger encoded packets instead of many densely spaced smaller packets. Such difference in traffic profiles may still affect the performance of congestion control, especially when outgoing packets are not paced by the media transport module. Investigation of varying frame rate and resolution are left for future work.

7. Combining The Two Models

It is worthwhile noting that the statistical and trace-driven models each have their own advantages and drawbacks. Both models are fairly simple to implement. It takes significantly greater effort to fit the parameters of a statistical model to actual encoder output data. In contrast, it is straightforward for a trace-driven model to obtain encoded frame size data. Once validated, the statistical model is more flexible in mimicking a wide range of encoder/content behaviors by simply varying the corresponding parameters in the model. In this regard, a trace-driven model relies -- by definition -- on additional data collection efforts for accommodating new codecs or video contents.

In general, the trace-driven model is more realistic for mimicking the ongoing, steady-state behavior of a video traffic source with fluctuations around a constant target rate. In contrast, the statistical model is more versatile for simulating the behavior of a video stream in transient, such as when encountering sudden rate changes. It is also possible to combine both methods into a hybrid model. In this case, the steady-state behavior is driven by traces during steady state and the transient-state behavior is driven by the statistical model.

```
                             transient +----------------+
                               state   | Generate next  |
                               +------>| K_d transient  |
            +-----------------+  /     |     frames     |
      R_v   | Compare against | /      +----------------+
     ------>|    previous     |/
            | target bitrate  |\
            +-----------------+ \      +----------------+
                                 \     | Generate next  |
                               +------>|  frame from    |
                                steady |     trace      |
                                 state +----------------+
```

Figure 3: A hybrid video traffic model

As shown in Figure 3, the video traffic model operates in a transient state if the requested target rate R_v is substantially different from the previous target, or else it operates in steady state. During the transient state, a total of K_d frames are generated by the statistical model, resulting in one (1) big burst frame with size K_B followed by K_d-1 smaller frames. When operating at steady state, the video traffic model simply generates a frame according to the trace-driven model given the target rate, while modulating the frame interval according to the distribution specified by the statistical model. One example criterion for determining whether the traffic model should operate in a transient state is whether the rate change exceeds 10% of the previous target rate. Finally, as this model follows transient-state behavior dictated by the statistical model, upon a substantial rate change, the model will follow the time-damping mechanism as defined in Section 5.1, which is governed by parameter tau_v.

8. Implementation Status

The statistical, trace-driven, and hybrid models as described in this draft have been implemented as a stand-alone, platform-independent synthetic traffic source module. It can be easily integrated into network simulation platforms such as [ns-2] and [ns-3], as well as testbeds using a real network. The stand-alone traffic source module is available as an open source implementation at [Syncodecs].

9. IANA Considerations

There are no IANA impacts in this memo.

10. Security Considerations

The synthetic video traffic models as described in this draft do not impose any security threats. They are designed to mimic realistic traffic patterns for evaluating candidate RTP-based congestion control algorithms, so as to ensure stable operations of the network. It is RECOMMENDED that candidate algorithms be tested using the video traffic models presented in this draft before wide deployment over the Internet. If the generated synthetic traffic flows are sent over the Internet, they also need to be congestion controlled.

11. References

11.1. Normative References

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

[H264] ITU-T Recommendation H.264, "Advanced video coding for generic audiovisual services", May 2003, <https://www.itu.int/rec/T-REC-H.264>.

[HEVC] ITU-T Recommendation H.265, "High efficiency video coding", April 2013, <https://www.itu.int/rec/T-REC-H.265>.

[Hu2010] Hu, H., Ma, Z., and Y. Wang, "Optimization of Spatial, Temporal and Amplitude Resolution for Rate-Constrained Video Coding and Scalable Video Adaptation", in Proc. 19th IEEE International Conference on Image Processing, (ICIP'12), September 2012.

[IETF-Interim] Zhu, X., Mena, S., and Z. Sarker, "Update on RMCAT Video Traffic Model: Trace Analysis and Model Update", April 2017, <https://www.ietf.org/proceedings/interim-2017-rmcat-01/slides/slides-interim-2017-rmcat-01-sessa-update-on-video-traffic-model-draft-00.pdf>.

[ns-2] "The Network Simulator - ns-2", <http://www.isi.edu/nsnam/ns/>.

[ns-3] "The Network Simulator - ns-3", <https://www.nsnam.org/>.

[Ozer2011] Ozer, J., "Video Compression for Flash, Apple Devices and HTML5", ISBN 13:978-0976259503, 2011.

[Papoulis] Papoulis, A., "Probability, Random Variables and Stochastic Processes", 2002.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <https://www.rfc-editor.org/info/rfc5104>.

[Syncodecs] Mena, S., D'Aronco, S., and X. Zhu, "Syncodecs: Synthetic codecs for evaluation of RMCAT work", <https://github.com/cisco/syncodecs>.

[Tanwir2013] Tanwir, S. and H. Perros, "A Survey of VBR Video Traffic Models", IEEE Communications Surveys and Tutorials, vol. 15, no. 5, pp. 1778-1802., October 2013.

Authors' Addresses

Xiaoqing Zhu
Cisco Systems
12515 Research Blvd., Building 4
Austin, TX 78759
USA

Email: xiaoqzhu@cisco.com

Sergio Mena de la Cruz
Cisco Systems
EPFL, Quartier de l’Innovation, Batiment E
Ecublens, Vaud 1015
Switzerland

Email: semena@cisco.com

Zaheduzzaman Sarker
Ericsson AB
Luleae, SE 977 53
Sweden

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