Network Working Group
Internet-Draft
Intended status: Experimental
Expires: April 11, 2016

X. Zhu
R. Pan
M. Ramalho
S. Mena
P. Jones
J. Fu
Cisco Systems
S. D'Aronco
EPFL
C. Ganzhorn
October 9, 2015

NADA: A Unified Congestion Control Scheme for Real-Time Media
draft-ietf-rmcat-nada-01

Abstract

This document describes NADA (network-assisted dynamic adaptation), a novel congestion control scheme for interactive real-time media applications, such as video conferencing. In the proposed scheme, the sender regulates its sending rate based on either implicit or explicit congestion signaling, in a unified approach. The scheme can benefit from explicit congestion notification (ECN) markings from network nodes. It also maintains consistent sender behavior in the absence of such markings, by reacting to queuing delays and packet losses instead.

Copyright Notice

Table of Contents

1. Introduction

Interactive real-time media applications introduce a unique set of challenges for congestion control. Unlike TCP, the mechanism used for real-time media needs to adapt quickly to instantaneous bandwidth changes, accommodate fluctuations in the output of video encoder rate control, and cause low queuing delay over the network. An ideal scheme should also make effective use of all types of congestion signals, including packet loss, queuing delay, and explicit congestion notification (ECN) [RFC3168] markings. The requirements for the congestion control algorithm are outlined in [I-D.ietf-rmcat-cc-requirements].

This document describes an experimental congestion control scheme called network-assisted dynamic adaptation (NADA). The NADA design benefits from explicit congestion control signals (e.g., ECN markings) from the network, yet also operates when only implicit congestion indicators (delay and/or loss) are available. In addition, it supports weighted bandwidth sharing among competing video flows. The signaling mechanism consists of standard RTP timestamp [RFC3550] and standard RTCP feedback reports.

2. Terminology

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 [RFC2119].

3. System Overview

Figure 1 shows the end-to-end system for real-time media transport that NADA operates in.

```
 +---------+  r_vin  +--------+        +--------+     +----------+
 |  Media  |<--------|  RTP   |        |Network |     |   RTP    |
 | Encoder |========>| Sender |=======>|  Node  |====>| Receiver |
 +---------+  r_vout +--------+ r_send +--------+     +----------+
                         /|\                                |
                          |                                 |
                          +---------------------------------+
                                RTCP Feedback Report
```

Figure 1: System Overview

- o Media encoder with rate control capabilities. It encodes the source media stream into an RTP stream with target bit rate r_vin. The actual output rate from the encoder r_vout may fluctuate around the target r_vin. In addition, the encoder can only change its bit rate at rather coarse time intervals, e.g., once every 0.5 seconds.

- o RTP sender: responsible for calculating the NADA reference rate based on network congestion indicators (delay, loss, or ECN marking reports from the receiver), for updating the video encoder with a new target rate r_vin, and for regulating the actual sending rate r_send accordingly. The RTP sender also provides an RTP timestamp for each outgoing packet.

- o RTP receiver: responsible for measuring and estimating end-to-end delay based on sender RTP timestamp, packet loss and ECN marking ratios, as well as receiving rate (r_recv) of the flow. It calculates the aggregated congestion signal (x_n) that accounts for queuing delay, ECN marking, and packet losses, and determines the mode for sender rate adaptation (rmode) based on whether the flow has encountered any standing non-zero congestion. The receiver sends periodic RTCP reports back to the sender, containing values of x_n, rmode, and r_recv.

- o Network node with several modes of operation. The system can work with the default behavior of a simple drop tail queue. It can also benefit from advanced AQM features such as PIE, FQ-CoDel, RED-based ECN marking, and PCN marking using a token bucket algorithm. Note that network node operation is out of scope for the design of NADA.

4. Core Congestion Control Algorithm

Like TCP-Friendly Rate Control (TFRC) [Floyd-CCR00] [RFC5348], NADA is a rate-based congestion control algorithm. In its simplest form, the sender reacts to the collection of network congestion indicators in the form of an aggregated congestion signal, and operates in one of two modes:

- o Accelerated ramp-up: when the bottleneck is deemed to be underutilized, the rate increases multiplicatively with respect to the rate of previously successful transmissions. The rate increase mutliplier (gamma) is calculated based on observed round-trip-time and target feedback interval, so as to limit self-inflicted queuing delay.

o Gradual rate update: in the presence of non-zero aggregate congestion signal, the sending rate is adjusted in reaction to both its value (x_n) and its change in value (x_diff).

This section introduces the list of mathematical notations and describes the core congestion control algorithm at the sender and receiver, respectively. Additional details on recommended practical implementations are described in Section 5.1 and Section 5.2.

4.1. Mathematical Notations

This section summarizes the list of variables and parameters used in the NADA algorithm.

Notation	Variable Name
t_curr	Current timestamp
t_last	Last time sending/receiving a feedback
delta	Observed interval between current and previous feedback reports: delta = t_curr-t_last
r_n	Reference rate based on network congestion
r_send	Sending rate
r_recv	Receiving rate
r_vin	Target rate for video encoder
r_vout	Output rate from video encoder
d_base	Estimated baseline delay
d_fwd	Measured and filtered one-way delay
d_n	Estimated queueing delay
d_tilde	Equivalent delay after non-linear warping
p_mark	Estimated packet ECN marking ratio
p_loss	Estimated packet loss ratio
x_n	Aggregate congestion signal
x_prev	Previous value of aggregate congestion signal
x_diff	Change in aggregate congestion signal w.r.t. its previous value: x_diff = x_n - x_prev
rmode	Rate update mode: (0 = accelerated ramp-up; 1 = gradual update)
gamma	Rate increase multiplier in accelerated ramp-up mode
rtt	Estimated round-trip-time at sender
buffer_len	Rate shaping buffer occupancy measured in bytes

Figure 2: List of variables.

Notation	Parameter Name	Default Value
PRIO	Weight of priority of the flow	1.0
RMIN	Minimum rate of application supported by media encoder	150 Kbps
RMAX	Maximum rate of application supported by media encoder	1.5 Mbps
X_REF	Reference congestion level	20ms
KAPPA	Scaling parameter for gradual rate update calculation	0.5
ETA	Scaling parameter for gradual rate update calculation	2.0
TAU	Upper bound of RTT in gradual rate update calculation	500ms
DELTA	Target feedback interval	100ms
LOGWIN	Observation window in time for calculating packet summary statistics at receiver	500ms
QEPS	Threshold for determining queuing delay build up at receiver	10ms
QTH	Delay threshold for non-linear warping	100ms
QMAX	Delay upper bound for non-linear warping	400ms
DLOSS	Delay penalty for loss	1.0s
DMARK	Delay penalty for ECN marking	200ms
GAMMA_MAX	Upper bound on rate increase ratio for accelerated ramp-up	20%
QBOUND	Upper bound on self-inflicted queuing delay during ramp up	50ms
FPS	Frame rate of incoming video	30
BETA_S	Scaling parameter for modulating outgoing sending rate	0.1
BETA_V	Scaling parameter for modulating video encoder target rate	0.1
ALPHA	Smoothing factor in exponential smoothing of packet loss and marking ratios	0.1

Figure 3: List of algorithm parameters.

4.2. Receiver-Side Algorithm

The receiver-side algorithm can be outlined as below:

```
On initialization:
  set d_base = +INFINITY
  set p_loss = 0
  set p_mark = 0
  set r_recv = 0
  set both t_last and t_curr as current time

On receiving a media packet:
  obtain current timestamp t_curr
  obtain from packet header sending time stamp t_sent
  obtain one-way delay measurement: d_fwd = t_curr - t_sent
  update baseline delay: d_base = min(d_base, d_fwd)
  update queuing delay:  d_n = d_fwd - d_base
  update packet loss ratio estimate p_loss
  update packet marking ratio estimate p_mark
  update measurement of receiving rate r_recv

On time to send a new feedback report (t_curr - t_last > DELTA):
  calculate non-linear warping of delay d_tilde if packet loss exists
  calculate aggregate congestion signal x_n
  determine mode of rate adaptation for sender: rmode
  send RTCP feedback report containing values of: rmode, x_n, and r_recv
  update t_last = t_curr
```

In order for a delay-based flow to hold its ground when competing against loss-based flows (e.g., loss-based TCP), it is important to distinguish between different levels of observed queuing delay. For instance, a moderate queuing delay value below 100ms is likely self-inflicted or induced by other delay-based flows, whereas a high queuing delay value of several hundreds of milliseconds may indicate the presence of a loss-based flow that does not refrain from increased delay.

When packet losses are observed, the estimated queuing delay follows a non-linear warping inspired by the delay-adaptive congestion window backoff policy in [Budzisz-TON11]:

```
             / d_n,                  if     d_n<QTH;
             |
             |     (QMAX - d_n)^4
d_tilde = <  QTH ----------------, if QTH<d_n<QMAX  (1)
             |     (QMAX - QTH)^4
             |
             \  0,                   otherwise.
```

Here, the queuing delay value is unchanged when it is below the first threshold QTH; it is scaled down following a non-linear curve when its value falls between QTH and QMAX; above QMAX, the high queuing delay value no longer counts toward congestion control.

The aggregate congestion signal is:

```
x_n = d_tilde + p_mark*DMARK + p_loss*DLOSS.      (2)
```

Here, DMARK is prescribed delay penalty associated with ECN markings and DLOSS is prescribed delay penalty associated with packet losses. The value of DLOSS and DMARK does not depend on configurations at the network node, but does assume that ECN markings, when available, occur before losses. Furthermore, the values of DLOSS and DMARK need to be set consistently across all NADA flows for them to compete fairly.

In the absence of packet marking and losses, the value of x_n reduces to the observed queuing delay d_n. In that case the NADA algorithm operates in the regime of delay-based adaptation.

Given observed per-packet delay and loss information, the receiver is also in a good position to determine whether the network is underutilized and recommend the corresponding rate adaptation mode for the sender. The criteria for operating in accelerated ramp-up mode are:

o No recent packet losses within the observation window LOGWIN; and

o No build-up of queuing delay: d_fwd-d_base < QEPS for all previous delay samples within the observation window LOGWIN.

Otherwise the algorithm operates in graduate update mode.

4.3. Sender-Side Algorithm

The sender-side algorithm is outlined as follows:

```
  on initialization:
    set r_n = RMIN
    set rtt = 0
    set x_prev = 0
    set t_last and t_curr as current time

  on receiving feedback report:
    obtain current timestamp: t_curr
    obtain values of rmode, x_n, and r_recv from feedback report
    update estimation of rtt
    measure feedback interval: delta = t_curr - t_last
    if rmode == 0:
      update r_n following accelerated ramp-up rules
    else:
      update r_n following gradual update rules
    clip rate r_n within the range of [RMIN, RMAX]
    x_prev = x_n
    t_last = t_curr
```

In accelerated ramp-up mode, the rate r_n is updated as follows:

```
                               QBOUND
    gamma = min(GAMMA_MAX, -----------)     (3)
                             rtt+DELTA

    r_n  =  (1+gamma) r_recv            (4)
```

The rate increase multiplier gamma is calculated as a function of upper bound of self-inflicted queuing delay (QBOUND), round-trip-time (rtt), and target feedback interval DELTA. It has a maximum value of GAMMA_MAX. The rationale behind (3)-(4) is that the longer it takes for the sender to observe self-inflicted queuing delay build-up, the more conservative the sender should be in increasing its rate, hence the smaller the rate increase multiplier.

In gradual update mode, the rate r_n is updated as:

```
    x_offset = x_n - PRIO*X_REF*RMAX/r_n           (5)

    x_diff   = x_n - x_prev                         (6)

                       delta    x_offset
    r_n = r_n - KAPPA*-------*------------*r_n
                        TAU       TAU

                           x_diff
                - KAPPA*ETA*---------*r_n            (7)
                            TAU
```

The rate changes in proportion to the previous rate decision. It is affected by two terms: offset of the aggregate congestion signal from its value at equilibrium (x_offset) and its change (x_diff). Calculation of x_offset depends on maximum rate of the flow (RMAX), its weight of priority (PRIO), as well as a reference congestion signal (X_REF). The value of X_REF is chosen that the maximum rate of RMAX can be achieved when the observed congestion signal level is below PRIO*X_REF.

At equilibrium, the aggregated congestion signal stablizes at x_n = PRIO*X_REF*RMAX/r_n. This ensures that when multiple flows share the same bottleneck and observe a common value of x_n, their rates at equilibrium will be proportional to their respective priority levels (PRIO) and maximum rate (RMAX).

As mentioned in the sender-side algorithm, the final rate is clipped within the dynamic range specified by the application:

```
        r_n = min(r_n, RMAX)          (8)

        r_n = max(r_n, RMIN)          (9)
```

The above operations ignore many practical issues such as clock synchronization between sender and receiver, filtering of noise in delay measurements, and base delay expiration. These will be addressed in later sections describing practical implementation of the NADA algorithm.

5. Practical Implementation of NADA

5.1. Receiver-Side Operation

The receiver continuously monitors end-to-end per-packet statistics in terms of delay, loss, and/or ECN marking ratios. It then aggregates all forms of congestion indicators into the form of an equivalent delay and periodically reports this back to the sender.

In addition, the receiver tracks the receiving rate of the flow and includes that in the feedback message.

5.1.1. Estimation of one-way delay and queuing delay

The delay estimation process in NADA follows a similar approach as in earlier delay-based congestion control schemes, such as LEDBAT [RFC6817]. NADA estimates the forward delay as having a constant base delay component plus a time varying queuing delay component. The base delay is estimated as the minimum value of one-way delay observed over a relatively long period (e.g., tens of minutes), whereas the individual queuing delay value is taken to be the difference between one-way delay and base delay.

The individual sample values of queuing delay should be further filtered against various non-congestion-induced noise, such as spikes due to processing "hiccup" at the network nodes. Current implementation employs a 15-tab minimum filter over per-packet queuing delay estimates.

5.1.2. Estimation of packet loss/marking ratio

The receiver detects packet losses via gaps in the RTP sequence numbers of received packets. Packets arriving out-of-order are discarded, and count towards losses. The instantaneous packet loss ratio p_inst is estimated as the ratio between the number of missing packets over the number of total transmitted packets within the recent observation window LOGWIN. The packet loss ratio p_loss is obtained after exponential smoothing:

    p_loss = ALPHA*p_inst + (1-ALPHA)*p_loss.   (10)

The filtered result is reported back to the sender as the observed packet loss ratio p_loss.

Estimation of packet marking ratio p_mark follows the same procedure as above. It is assumed that ECN marking information at the IP header can be passed to the transport layer by the receiving endpoint.

5.1.3. Estimation of receiving rate

It is fairly straighforward to estimate the receiving rate r_recv. NADA maintains a recent observation window with time span of LOGWIN, and simply divides the total size of packets arriving during that window over the time span. The receiving rate (r_recv) is included as part of the feedback report.

5.2. Sender-Side Operation

Figure 4 provides a detailed view of the NADA sender. Upon receipt of an RTCP feedback report from the receiver, the NADA sender calculates the reference rate r_n as specified in Section 4.3. It further adjusts both the target rate for the live video encoder r_vin and the sending rate r_send over the network based on the updated value of r_n and rate shaping buffer occupancy buffer_len.

The NADA sender behavior stays the same in the presence of all types of congestion indicators: delay, loss, and ECN marking. This unified approach allows a graceful transition of the scheme as the network shifts dynamically between light and heavy congestion levels.

```
                      +----------------+
                      |  Calculate     | <---- RTCP report
                      | Reference Rate |
                      -----------------+
                              | r_n
                 +------------+-------------+
                 |                          |
                \|/                        \|/
         +-----------------+           +---------------+
         | Calculate Video |           |   Calculate   |
         |  Target Rate    |           | Sending Rate  |
         +-----------------+           +---------------+
             |        /|\                 /|\      |
       r_vin |         |                   |       |
            \|/        +-------------------+       |
         +----------+          | buffer_len        |  r_send
         |  Video   | r_vout  -----------+        \|/
         |  Encoder |-------->   |||||||||=================>
         +----------+         -----------+    RTP packets
                             Rate Shaping Buffer
```

Figure 4: NADA Sender Structure

5.2.1. Rate shaping buffer

The operation of the live video encoder is out of the scope of the design for the congestion control scheme in NADA. Instead, its behavior is treated as a black box.

A rate shaping buffer is employed to absorb any instantaneous mismatch between encoder rate output r_vout and regulated sending rate r_send. Its current level of occupancy is measured in bytes and is denoted as buffer_len.

A large rate shaping buffer contributes to higher end-to-end delay, which may harm the performance of real-time media communications. Therefore, the sender has a strong incentive to prevent the rate shaping buffer from building up. The mechanisms adopted are:

- To deplete the rate shaping buffer faster by increasing the sending rate r_send; and

- To limit incoming packets of the rate shaping buffer by reducing the video encoder target rate r_vin.

5.2.2. Adjusting video target rate and sending rate

The target rate for the live video encoder deviates from the network congestion control rate r_n based on the level of occupancy in the rate shaping buffer:

```
r_vin = r_n - BETA_V*8*buffer_len*FPS.     (11)
```

The actual sending rate r_send is regulated in a similar fashion:

```
r_send = r_n + BETA_S*8*buffer_len*FPS.    (12)
```

In (11) and (12), the first term indicates the rate calculated from network congestion feedback alone. The second term indicates the influence of the rate shaping buffer. A large rate shaping buffer nudges the encoder target rate slightly below -- and the sending rate slightly above -- the reference rate r_n.

Intuitively, the amount of extra rate offset needed to completely drain the rate shaping buffer within the duration of a single video frame is given by 8*buffer_len*FPS, where FPS stands for the frame rate of the video. The scaling parameters BETA_V and BETA_S can be tuned to balance between the competing goals of maintaining a small rate shaping buffer and deviating the system from the reference rate point.

6. Discussions and Further Investigations

6.1. Choice of delay metrics

The current design works with relative one-way-delay (OWD) as the main indication of congestion. The value of the relative OWD is obtained by maintaining the minimum value of observed OWD over a relatively long time horizon and subtract that out from the observed absolute OWD value. Such an approach cancels out the fixed difference between the sender and receiver clocks. It has been widely adopted by other delay-based congestion control approaches

such as [RFC6817]. As discussed in [RFC6817], the time horizon for tracking the minimum OWD needs to be chosen with care: it must be long enough for an opportunity to observe the minimum OWD with zero queuing delay along the path, and sufficiently short so as to timely reflect "true" changes in minimum OWD introduced by route changes and other rare events.

The potential drawback in relying on relative OWD as the congestion signal is that when multiple flows share the same bottleneck, the flow arriving late at the network experiencing a non-empty queue may mistakenly consider the standing queuing delay as part of the fixed path propagation delay. This will lead to slightly unfair bandwidth sharing among the flows.

Alternatively, one could move the per-packet statistical handling to the sender instead and use relative round-trip-time (RTT) in lieu of relative OWD, assuming that per-packet acknowledgements are available. The main drawback of RTT-based approach is the noise in the measured delay in the reverse direction.

Note that the choice of either delay metric (relative OWD vs. RTT) involves no change in the proposed rate adaptation algorithm. Therefore, comparing the pros and cons regarding which delay metric to adopt can be kept as an orthogonal direction of investigation.

6.2. Method for delay, loss, and marking ratio estimation

Like other delay-based congestion control schemes, performance of NADA depends on the accuracy of its delay measurement and estimation module. Appendix A in [RFC6817] provides an extensive discussion on this aspect.

The current recommended practice of simply applying a 15-tab minimum filter suffices in guarding against processing delay outliers observed in wired connections. For wireless connections with a higher packet delay variation (PDV), more sophisticated techniques on de-noising, outlier rejection, and trend analysis may be needed.

More sophisticated methods in packet loss ratio calculation, such as that adopted by [Floyd-CCR00], will likely be beneficial. These alternatives are currently under investigation.

6.3. Impact of parameter values

In the gradual rate update mode, the parameter TAU indicates the upper bound of round-trip-time (RTT) in feedback control loop. Typically, the observed feedback interval delta is close to the target feedback interval DELTA, and the relative ratio of delta/TAU

versus ETA dictates the relative strength of influence from the aggregate congestion signal offset term (x_offset) versus its recent change (x_diff), respectively. These two terms are analogous to the integral and proportional terms in a proportional-integral (PI) controller. The recommended choice of TAU=500ms, DELTA=100ms and ETA = 2.0 corresponds to a relative ratio of 1:10 between the gains of the integral and proportional terms. Consequently, the rate adaptation is mostly driven by the change in the congestion signal with a long-term shift towards its equilibrium value driven by the offset term. Finally, the scaling parameter KAPPA determines the overall speed of the adaptation and needs to strike a balance between responsiveness and stability.

The choice of the target feedback interval DELTA needs to strike the right balance between timely feedback and low RTCP feedback message counts. A target feedback interval of DELTA=100ms is recommended, corresponding to a feedback bandwidth of 16Kbps with 200 bytes per feedback message --- less than 0.1% overhead for a 1 Mbps flow. Furthermore, both simulation studies and frequency-domain analysis have established that a feedback interval below 250ms will not break up the feedback control loop of NADA congestion control.

In calculating the non-linear warping of delay in (1), the current design uses fixed values of QTH and QMAX. It is possible to adapt the value of both based on past observations of queuing delay in the presence of packet losses.

In calculating the aggregate congestion signal x_n, the choice of DMARK and DLOSS influence the steady-state packet loss/marking ratio experienced by the flow at a given available bandwidth. Higher values of DMARK and DLOSS result in lower steady-state loss/marking ratios, but are more susceptible to the impact of individual packet loss/marking events. While the value of DMARK and DLOSS are fixed and predetermined in the current design, a scheme for automatically tuning these values based on desired bandwidth sharing behavior in the presence of other competing loss-based flows (e.g., loss-based TCP) is under investigation.

[Editor's note: Choice of start value: is this in scope of congestion control, or should this be decided by the application?]

6.4. Sender-based vs. receiver-based calculation

In the current design, the aggregated congestion signal x_n is calculated at the receiver, keeping the sender operation completely independent of the form of actual network congestion indications (delay, loss, or marking). Alternatively, one can move the logics of (1) and (2) to the sender. Such an approach requires slightly higher

overhead in the feedback messages, which should contain individual fields on queuing delay (d_n), packet loss ratio (p_loss), packet marking ratio (p_mark), receiving rate (r_recv), and recommended rate adaptation mode (rmode).

6.5. Incremental deployment

One nice property of NADA is the consistent video endpoint behavior irrespective of network node variations. This facilitates gradual, incremental adoption of the scheme.

To start off with, the proposed congestion control mechanism can be implemented without any explicit support from the network, and relies solely on observed one-way delay measurements and packet loss ratios as implicit congestion signals.

When ECN is enabled at the network nodes with RED-based marking, the receiver can fold its observations of ECN markings into the calculation of the equivalent delay. The sender can react to these explicit congestion signals without any modification.

Ultimately, networks equipped with proactive marking based on token bucket level metering can reap the additional benefits of zero standing queues and lower end-to-end delay and work seamlessly with existing senders and receivers.

7. Implementation Status

The NADA scheme has been implemented in [ns-2] and [ns-3] simulation platforms. Extensive ns-2 simulation evaluations of an earlier version of the draft are documented in [Zhu-PV13]. Evaluation results of the current draft over several test cases in [I-D.ietf-rmcat-eval-test] have been presented at recent IETF meetings [IETF-90][IETF-91].

The scheme has also been implemented and evaluated in a lab setting as described in [IETF-90]. Preliminary evaluation results of NADA in single-flow and multi-flow scenarios have been presented in [IETF-91].

8. IANA Considerations

This document makes no request of IANA.

9. Acknowledgements

The authors would like to thank Randell Jesup, Luca De Cicco, Piers O'Hanlon, Ingemar Johansson, Stefan Holmer, Cesar Ilharco Magalhaes, Safiqul Islam, Mirja Kuhlewind, and Karen Elisabeth Egede Nielsen for their valuable questions and comments on earlier versions of this draft.

10. References

10.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <http://www.rfc-editor.org/info/rfc3168>.

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

[I-D.ietf-rmcat-eval-criteria] Singh, V. and J. Ott, "Evaluating Congestion Control for Interactive Real-time Media", draft-ietf-rmcat-eval-criteria-03 (work in progress), March 2015.

[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-02 (work in progress), September 2015.

[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.

10.2. Informative References

[RFC2309] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, DOI 10.17487/RFC2309, April 1998, <http://www.rfc-editor.org/info/rfc2309>.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, DOI 10.17487/RFC5348, September 2008, <http://www.rfc-editor.org/info/rfc5348>.

[RFC6660] Briscoe, B., Moncaster, T., and M. Menth, "Encoding Three Pre-Congestion Notification (PCN) States in the IP Header Using a Single Diffserv Codepoint (DSCP)", RFC 6660, DOI 10.17487/RFC6660, July 2012, <http://www.rfc-editor.org/info/rfc6660>.

[RFC6817] Shalunov, S., Hazel, G., Iyengar, J., and M. Kuehlewind, "Low Extra Delay Background Transport (LEDBAT)", RFC 6817, DOI 10.17487/RFC6817, December 2012, <http://www.rfc-editor.org/info/rfc6817>.

[Floyd-CCR00] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "Equation-based Congestion Control for Unicast Applications", ACM SIGCOMM Computer Communications Review vol. 30, no. 4, pp. 43-56, October 2000.

[Budzisz-TON11] Budzisz, L., Stanojevic, R., Schlote, A., Baker, F., and R. Shorten, "On the Fair Coexistence of Loss- and Delay-Based TCP", IEEE/ACM Transactions on Networking vol. 19, no. 6, pp. 1811-1824, December 2011.

[Zhu-PV13] Zhu, X. and R. Pan, "NADA: A Unified Congestion Control Scheme for Low-Latency Interactive Video", in Proc. IEEE International Packet Video Workshop (PV'13) San Jose, CA, USA, December 2013.

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

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

[IETF-90] Zhu, X., Ramalho, M., Ganzhorn, C., Jones, P., and R. Pan, "NADA Update: Algorithm, Implementation, and Test Case Evalua6on Results", July 2014, <https://tools.ietf.org/agenda/90/slides/slides-90-rmcat-6.pdf>.

[IETF-91] Zhu, X., Pan, R., Ramalho, M., Mena, S., Ganzhorn, C., Jones, P., and S. D'Aronco, "NADA Algorithm Update and Test Case Evaluations", November 2014, <http://www.ietf.org/proceedings/interim/2014/11/09/rmcat/slides/slides-interim-2014-rmcat-1-2.pdf>.

Appendix A. Network Node Operations

NADA can work with different network queue management schemes and does not assume any specific network node operation. As an example, this appendix describes three variants of queue management behavior at the network node, leading to either implicit or explicit congestion signals.

In all three flavors described below, the network queue operates with the simple first-in-first-out (FIFO) principle. There is no need to maintain per-flow state. The system can scale easily with a large number of video flows and at high link capacity.

A.1. Default behavior of drop tail queues

In a conventional network with drop tail or RED queues, congestion is inferred from the estimation of end-to-end delay and/or packet loss. Packet drops at the queue are detected at the receiver, and contributes to the calculation of the aggregated congestion signal x_n. No special action is required at network node.

A.2. RED-based ECN marking

In this mode, the network node randomly marks the ECN field in the IP packet header following the Random Early Detection (RED) algorithm [RFC2309]. Calculation of the marking probability involves the following steps:

```
on packet arrival:
    update smoothed queue size q_avg as:
        q_avg = w*q + (1-w)*q_avg.

    calculate marking probability p as:

       / 0,                    if q < q_lo;
       |
       |        q_avg - q_lo
  p= <  p_max*--------------, if q_lo <= q < q_hi;
       |         q_hi - q_lo
       |
       \ p = 1,                if q >= q_hi.
```

Here, q_lo and q_hi corresponds to the low and high thresholds of queue occupancy. The maximum marking probability is p_max.

The ECN markings events will contribute to the calculation of an equivalent delay x_n at the receiver. No changes are required at the sender.

A.3. Random Early Marking with Virtual Queues

Advanced network nodes may support random early marking based on a token bucket algorithm originally designed for Pre-Congestion Notification (PCN) [RFC6660]. The early congestion notification (ECN) bit in the IP header of packets are marked randomly. The marking probability is calculated based on a token-bucket algorithm originally designed for the Pre-Congestion Notification (PCN) [RFC6660]. The target link utilization is set as 90%; the marking probability is designed to grow linearly with the token bucket size when it varies between 1/3 and 2/3 of the full token bucket limit.

* upon packet arrival, meter packet against token bucket (r,b);

* update token level b_tk;

* calculate the marking probability as:

```
       / 0,                     if b-b_tk < b_lo;
       |
       |          b-b_tk-b_lo
  p = <  p_max* --------------, if b_lo<= b-b_tk <b_hi;
       |           b_hi-b_lo
       |
       \ 1,                     if b-b_tk>=b_hi.
```

Here, the token bucket lower and upper limits are denoted by b_lo and b_hi, respectively. The parameter b indicates the size of the token bucket. The parameter r is chosen to be below capacity, resulting in slight under-utilization of the link. The maximum marking probability is p_max.

The ECN markings events will contribute to the calculation of an equivalent delay x_n at the receiver. No changes are required at the sender. The virtual queuing mechanism from the PCN-based marking algorithm will lead to additional benefits such as zero standing queues.

Authors' Addresses

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

Email: xiaoqzhu@cisco.com

Rong Pan
Cisco Systems
3625 Cisco Way
San Jose, CA 95134
USA

Email: ropan@cisco.com

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

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

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

Email: semena@cisco.com

Paul E. Jones
Cisco Systems
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Email: paulej@packetizer.com

Jiantao Fu
Cisco Systems
707 Tasman Drive
Milpitas, CA 95035
USA

Email: jianfu@cisco.com

Stefano D'Aronco
Ecole Polytechnique Federale de Lausanne
EPFL STI IEL LTS4, ELD 220 (Batiment ELD), Station 11
Lausanne CH-1015
Switzerland

Email: stefano.daronco@epfl.ch

Charles Ganzhorn
7900 International Drive, International Plaza, Suite 400
Bloomington, MN 55425
USA

Email: charles.ganzhorn@gmail.com