        NADA: A Unified Congestion Control Scheme for Real-Time Media
                        draft-ietf-rmcat-nada-13

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.

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 March 8, 2020.

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]. It highlights that the desired
   congestion control scheme should avoid flow starvation and attain a
   reasonable fair share of bandwidth when competing against other
   flows, adapt quickly, and operate in a stable manner.

   This document describes an experimental congestion control scheme
   called network-assisted dynamic adaptation (NADA). The design of
   NADA 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. Such a
   unified sender behavior distinguishes NADA from other congestion
   control schemes for real-time media. In addition, its core
   congestion control algorithm is designed to guarantee stability for
   path round-trip-times (RTTs) below a prescribed bound (e.g., 250ms
   with default parameter choices). It further supports weighted
   bandwidth sharing among competing video flows with different
   priorities. The signaling mechanism consists of standard RTP
   timestamp [RFC3550] and RTCP feedback reports. The definition of the
   desired RTCP feedback message is described in detail in
   [I-D.ietf-avtcore-cc-feedback-message] so as to support the
   successful operation of several congestion control schemes for real-
   time interactive media.

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. System Overview

   Figure 1 shows the end-to-end system for real-time media transport
   that NADA operates in. Note that there also exist network nodes
   along the reverse (potentially uncongested) path that the RTCP

feedback reports traverse. Those network nodes are not shown in the
figure for sake of brevity.

```
   +---------+  r_vin  +--------+         +--------+      +----------+
   |  Media  |<--------|   RTP  |         |Network |      |   RTP    |
   | Encoder |========>| Sender |=======>|  Node  |====>| Receiver |
   +---------+  r_vout +--------+ r_send +--------+      +----------+
                         /|\                                  |
                          |                                   |
                          +-----------------------------------+
                                   RTCP Feedback Report
```
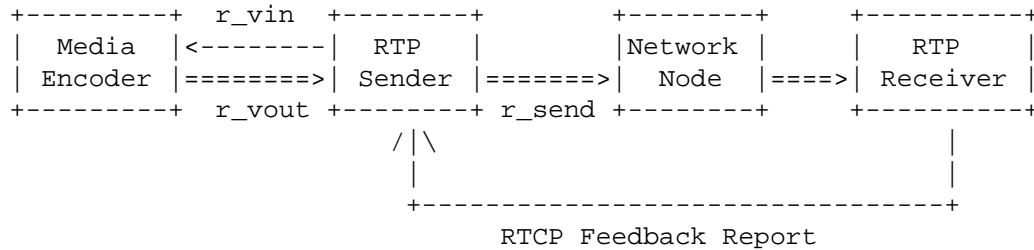
Figure 1: System Overview

o Media encoder with rate control capabilities. It encodes raw
   media (audio and video) frames into a compressed bitstream which
   is later packetized into RTP packets. As discussed in [RFC8593],
   the actual output rate from the encoder r_vout may fluctuate
   around the target r_vin. Furthermore, it is possible that the
   encoder can only react to bit rate changes 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 generates a
   sending timestamp for each outgoing packet.

o RTP receiver: responsible for measuring and estimating end-to-end
   delay (based on sender timestamp), packet loss (based on RTP
   sequence number), ECN marking ratios (based on [RFC6679]), and
   receiving rate (r_recv) of the flow. It calculates the aggregated
   congestion signal (x_curr) that accounts for queuing delay, ECN
   markings, and packet losses. The receiver also 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_curr, 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 [RFC8033], FQ-
   CoDel [RFC8290], ECN marking based on RED [RFC7567], and PCN
   marking using a token bucket algorithm ([RFC6660]). Note that
   network node operation is out of control 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 multiplier (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_curr) 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. Figure 3 also includes the default values for
   choosing the algorithm parameters either to represent a typical
   setting in practical applications or based on theoretical and
   simulation studies. See Section 6.3 for some of the discussions on
   the impact of parameter values. Additional studies in real-world
   settings suggested in Section 8 could gather further insight on how
   to choose and adapt these parameter values in practical deployment.

```
+--------------+----------------------------------------------------+
| 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_ref        | 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_queue      | Estimated queuing delay                            |
| d_tilde      | Equivalent delay after non-linear warping          |
| p_mark       | Estimated packet ECN marking ratio                 |
| p_loss       | Estimated packet loss ratio                        |
| x_curr       | 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_curr - x_prev       |
| rmode        | Rate update mode: (0 = accelerated ramp-up;        |
|              | 1 = gradual update)                                |
| gamma        | Rate increase multiplier in accelerated ramp-up    |
|              | mode                                               |
| loss_int     | Measured average loss interval in packet count     |
| loss_exp     | Threshold value for setting the last observed      |
|              | packet loss to expiration                          |
| 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       |   150Kbps      |
|              | supported by media encoder        |                |
| RMAX         | Maximum rate of application       |   1.5Mbps      |
|              | supported by media encoder        |                |
| XREF         | Reference congestion level        |   10ms         |
| KAPPA        | Scaling parameter for gradual     |   0.5          |
|              | rate update calculation           |                |
| ETA          | Scaling parameter for gradual     |   2.0          |
|              | rate update calculation           |                |
```

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	
DFILT	Bound on filtering delay	120ms	
GAMMA_MAX	Upper bound on rate increase ratio for accelerated ramp-up	0.5	
QBOUND	Upper bound on self-inflicted queuing delay during ramp up	50ms	
MULTILOSS	Multiplier for self-scaling the expiration threshold of the last observed loss (loss_exp) based on measured average loss interval (loss_int)	7.0	
QTH	Delay threshold for invoking non-linear warping	50ms	
LAMBDA	Scaling parameter in the exponent of non-linear warping	0.5	
PLRREF	Reference packet loss ratio	0.01	
PMRREF	Reference packet marking ratio	0.01	
DLOSS	Reference delay penalty for loss when packet loss ratio is at PLRREF	10ms	
DMARK	Reference delay penalty for ECN marking when packet marking is at PMRREF	2ms	
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 and their default values.

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 in milliseconds

   On receiving a media packet:
     obtain current timestamp t_curr from system clock
     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_queue = 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 current aggregate congestion signal x_curr
     determine mode of rate adaptation for sender: rmode
     send feedback containing values of: rmode, x_curr, 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, over wired connections, a moderate queuing delay value on
   the order of tens of milliseconds 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.

   If the last observed packet loss is within the expiration window of
   loss_exp (measured in terms of packet counts), the estimated queuing
   delay follows a non-linear warping:

$$
d\_tilde = \begin{cases} d\_queue, & \text{if } d\_queue < QTH; \\ QTH \exp\left(-LAMBDA \dfrac{(d\_queue - QTH)}{QTH}\right), & \text{otherwise.} \end{cases} \quad (1)
$$

In (1), the queuing delay value is unchanged when it is below the first threshold QTH; otherwise it is scaled down following a non-linear curve. This non-linear warping is inspired by the delay-adaptive congestion window backoff policy in [Budzisz-TON11], so as to "gradually nudge" the controller to operate based on loss-induced congestion signals when competing against loss-based flows. The exact form of the non-linear function has been simplified with respect to [Budzisz-TON11]. The value of the threshold QTH should be carefully tuned for different operational environments, so as to avoid potential risks of prematurely discounting the congestion signal level. Typically, a higher value of QTH is required in a noisier environment (e.g., over wireless connections, or where the video stream encounters many time-varying background competing traffic) so as to stay robust against occasional non-congestion-induced delay spikes. Additional insights on how this value can be tuned or auto-tuned should be gathered from carrying out experimental studies in different real-world deployment scenarios.

The value of loss_exp is configured to self-scale with the average packet loss interval loss_int with a multiplier MULTILOSS:

$$
loss\_exp = MULTILOSS * loss\_int.
$$

Estimation of the average loss interval loss_int, in turn, follows Section 5.4 of the TCP Friendly Rate Control (TFRC) protocol [RFC5348].

In practice, it is recommended to linearly interpolate between the warped (d_tilde) and non-warped (d_queue) values of the queuing delay during the transitional period lasting for the duration of loss_int.

The aggregate congestion signal is:

$$
x\_curr = d\_tilde + DMARK*\left(\frac{p\_mark}{PMRREF}\right)^2 + DLOSS*\left(\frac{p\_loss}{PLRREF}\right)^2. \quad (2)
$$

Here, DMARK is prescribed reference delay penalty associated with ECN markings at the reference marking ratio of PMRREF; DLOSS is prescribed reference delay penalty associated with packet losses at the reference packet loss ratio of PLRREF. The value of DLOSS and DMARK does not depend on configurations at the network node. Since ECN-enabled active queue management schemes typically mark a packet before dropping it, the value of DLOSS SHOULD be higher than that of DMARK. Furthermore, the values of DLOSS and DMARK need to be set consistently across all NADA flows sharing the same bottleneck link, so that they can compete fairly.

In the absence of packet marking and losses, the value of x_curr reduces to the observed queuing delay d_queue. 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_ref = RMIN
     set rtt = 0
     set x_prev = 0
     set t_last and t_curr as current system clock time

   on receiving feedback report:
     obtain current timestamp from system clock: t_curr
     obtain values of rmode, x_curr, and r_recv from feedback report
     update estimation of rtt
     measure feedback interval: delta = t_curr - t_last
     if rmode == 0:
       update r_ref following accelerated ramp-up rules
     else:
       update r_ref following gradual update rules

     clip rate r_ref within the range of minimum rate (RMIN)
     and maximum rate (RMAX).
     x_prev = x_curr
     t_last = t_curr
```

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

$$gamma = \min\left(GAMMA\_MAX, \frac{QBOUND}{rtt+DELTA+DFILT}\right) \quad (3)$$

$$r\_ref = \max(r\_ref, (1+gamma)\, r\_recv) \quad (4)$$

The rate increase multiplier gamma is calculated as a function of upper bound of self-inflicted queuing delay (QBOUND), round-trip-time (rtt), target feedback interval (DELTA) and bound on filtering delay for calculating d_queue (DFILT). 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_ref is updated as:

```
    x_offset = x_curr - PRIO*XREF*RMAX/r_ref         (5)

    x_diff    = x_curr - x_prev                      (6)

                     delta     x_offset
    r_ref = r_ref - KAPPA*-------*------------*r_ref
                      TAU        TAU

                        x_diff
             - KAPPA*ETA*---------*r_ref              (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 (XREF). The value of XREF is chosen so that the maximum rate
of RMAX can be achieved when the observed congestion signal level is
below PRIO*XREF.

At equilibrium, the aggregated congestion signal stabilizes at x_curr
= PRIO*XREF*RMAX/r_ref. This ensures that when multiple flows share
the same bottleneck and observe a common value of x_curr, their rates
at equilibrium will be proportional to their respective priority
levels (PRIO) and the range between minimum and maximum rate. Values
of the minimum rate (RMIN) and maximum rate (RMAX) will be provided
by the media codec, for instance, as outlined by
[I-D.ietf-rmcat-cc-codec-interactions]. In the absence of such
information, NADA sender will choose a default value of 0 for RMIN,
and 3Mbps for RMAX.

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

```
    r_ref = min(r_ref, RMAX)        (8)
    r_ref = max(r_ref, 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 Section 5.

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]. For experimental implementations, instead of relying on
   RTP timestamps and the transmission time offset RTP header extension
   [RFC5450], the NADA sender can generate its own timestamp based on
   local system clock and embed that information in the transport packet
   header. The NADA receiver 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. By re-
   estimating the base delay periodically, one can avoid the potential
   issue of base delay expiration, whereby an earlier measured base
   delay value is no longer valid due to underlying route changes or
   cumulative timing difference introduced by the clock rate skew
   between sender and receiver. All delay estimations are based on
   sender timestamps with a recommended granularity of 100 microseconds
   or finer.

   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. Therefore, in
   addition to calculating the value of queuing delay using d_queue =
   d_fwd - d_base, as expressed in Section 5.1, current implementation
   further employs a minimum filter with a window size of 15 samples
   over per-packet queuing delay values.

5.1.2. Estimation of packet loss/marking ratio

   The receiver detects packet losses via gaps in the RTP sequence
   numbers of received packets. For interactive real-time media
   application with stringent latency constraint (e.g., video
   conferencing), the receiver avoids the packet re-ordering delay by
   treating out-of-order packets as 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:

$$p\_loss = ALPHA*p\_inst + (1-ALPHA)*p\_loss. \quad (10)$$

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 receiving endpoint, e.g., by following
the mechanism described in [RFC6679].

5.1.3. Estimation of receiving rate

It is fairly straightforward 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) can be
calculated at either the sender side based on the per-packet feedback
from the receiver, or 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_ref 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_ref 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
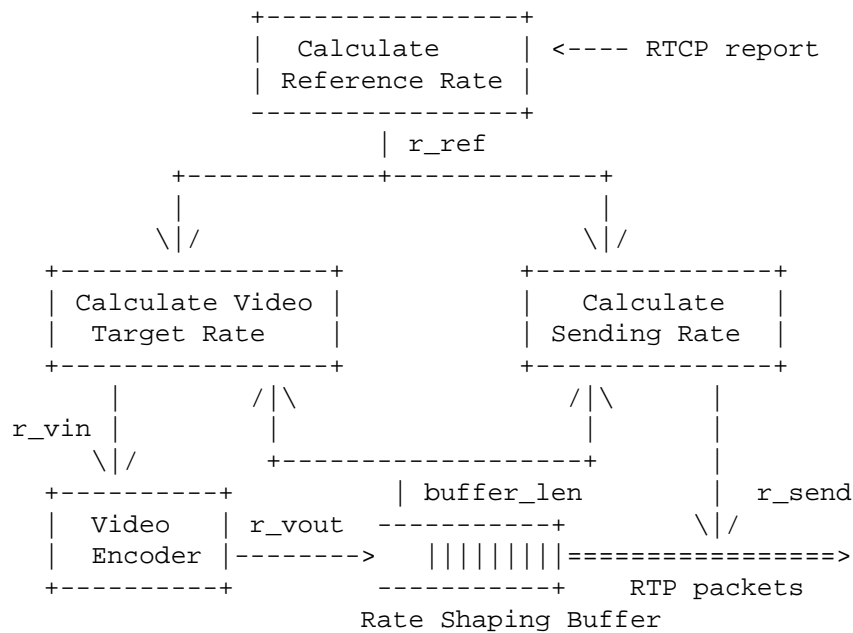shifts dynamically between light and heavy congestion levels.

```
                      +-----------------+
                      |   Calculate     | <---- RTCP report
                      | Reference Rate  |
                      +-----------------+
                             | r_ref
               +-----------+------------+
               |                        |
              \|/                      \|/
         +----------------+      +---------------+
         | 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:

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

   o 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

   If the level of occupancy in the rate shaping buffer is accessible at
   the sender, such information can be leveraged to further adjust the
   target rate of the live video encoder r_vin as well as the actual
   sending rate r_send. The purpose of such adjustments is to mitigate
   the additional latencies introduced by the rate shaping buffer. The
   amount of rate adjustment can be calculated as follows:

       r_diff_v = min(0.05*r_ref, BETA_V*8*buffer_len*FPS).   (11)
       r_diff_s = min(0.05*r_ref, BETA_S*8*buffer_len*FPS).   (12)
       r_vin = max(RMIN, r_ref - r_diff_v).   (13)
       r_send = min(RMAX, r_ref + r_diff_s).   (14)

   In (11) and (12), the amount of adjustment is calculated as
   proportional to the size of the rate shaping buffer but is bounded by
   5% of the reference rate r_ref calculated from network congestion
   feedback alone. This ensures that the adjustment introduced by the
   rate shaping buffer will not counteract with the core congestion
   control process. Equations (13) and (14) indicate 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_ref. The final video target rate
   (r_vin) and sending rate (r_send) are further bounded within the
   original range of [RMIN, RMAX].

   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
   reference 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 from the
   reference rate point. Empirical observations show that the rate
   shaping buffer for a responsive live video encoder typically stays
   empty and only occasionally holds a large frame (e.g., when an intra-
   frame is produced) in transit. Therefore, the rate adjustment
   introduced by this mechanism is expected to be minor. For instance,
   a rate shaping buffer of 2000 Bytes will lead to a rate adjustment of
   48Kbps given the recommended scaling parameters of BETA_V = 0.1 and
   BETA_S = 0.1 and reference frame rate of FPS = 30.

5.3. Feedback Message Requirements

   The following list of information is required for NADA congestion
   control to function properly:

    o Recommended rate adaptation mode (rmode): a 1-bit flag indicating
       whether the sender should operate in accelerated ramp-up mode
       (rmode=0) or gradual update mode (rmode=1).

    o Aggregated congestion signal (x_curr): the most recently updated
       value, calculated by the receiver according to Section 4.2. This
       information can be expressed with a unit of 100 microsecond (i.e.,
       1/10 of a millisecond) in 15 bits. This allows a maximum value of
       x_curr at approximately 3.27 second.

    o Receiving rate (r_recv): the most recently measured receiving rate
       according to Section 5.1.3. This information is expressed with a
       unit of bits per second (bps) in 32 bits (unsigned int). This
       allows a maximum rate of approximately 4.3Gbps, approximately 1000
       times of the streaming rate of a typical high-definition (HD)
       video conferencing session today. This field can be expanded
       further by a few more bytes, in case an even higher rate need to
       be specified.

    The above list of information can be accommodated by 48 bits, or 6
    bytes, in total. They can be either included in the feedback report
    from the receiver, or, in the case where all receiver-side
    calculations are moved to the sender, derived from per-packet
    information from the feedback message as defined in
    [I-D.ietf-avtcore-cc-feedback-message]. Choice of the feedback
    message interval DELTA is discussed in Section 6.3. A target
    feedback interval of DELTA=100ms is recommended.

6. Discussions and Further Investigations

    This section discussed the various design choices made by NADA,
    potential alternative variants of its implementation, and guidelines
    on how the key algorithm parameters can be chosen. Section 8
    recommends additional experimental setups to further explore these
    topics.

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

standing queue along the path, and sufficiently short so as to timely
reflect "true" changes in minimum OWD introduced by route changes and
other rare events and to mitigate the cumulative impact of clock rate
skew over time.

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 acknowledgments 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 applying minimum filter with a
window size of 15 samples 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 part of the experiments this document proposes.

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 --- approximately 1.6% overhead for a 1Mbps flow.
Furthermore, both simulation studies and frequency-domain analysis in
[IETF-95] have established that a feedback interval below 250ms
(i.e., more frequently than 4 feedback messages per second) 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 for determining whether to perform
the non-linear warping). Its value should be carefully tuned for
different operational environments (e.g., over wired vs. wireless
connections), so as to avoid the potential risk of prematurely
discounting the congestion signal level. It is possible to adapt its
value based on past observed patterns of queuing delay in the
presence of packet losses. It needs to be noted that the non-linear
warping mechanism may lead to multiple NADA streams stuck in loss-
based mode when competing against each other.

In calculating the aggregate congestion signal x_curr, 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, this document also
encourages further explorations of 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).

6.4. Sender-based vs. receiver-based calculation

   In the current design, the aggregated congestion signal x_curr is
   calculated at the receiver, keeping the sender operation completely
   independent of the form of actual network congestion indications
   (delay, loss, or marking) in use.

   Alternatively, one can shift receiver-side calculations to the
   sender, whereby the receiver simply reports on per-packet information
   via periodic feedback messages as defined in
   [I-D.ietf-avtcore-cc-feedback-message]. Such an approach enables
   interoperability amongst senders operating on different congestion
   control schemes, but requires slightly higher overhead in the
   feedback messages. See additional discussions in
   [I-D.ietf-avtcore-cc-feedback-message] regarding the desired format
   of the feedback messages and the recommended feedback intervals.

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.

   Initially, the proposed congestion control mechanism can be
   implemented without any explicit support from the network, and relies
   solely on observed relative 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. Reference Implementations

   The NADA scheme has been implemented in both [ns-2] and [ns-3]
   simulation platforms. The implementation in ns-2 hosts the
   calculations as described in Section 4.2 at the receiver side,
   whereas the implementation in ns-3 hosts these receiver-side
   calculations at the sender for the sake of interoperability.
   Extensive ns-2 simulation evaluations of an earlier version of the
   draft are documented in [Zhu-PV13]. An open source implementation of
   NADA as part of a ns-3 module is available at [ns3-rmcat].

Evaluation results of the current draft based on ns-3 are presented in [IETF-90] and [IETF-91] for wired test cases as documented in [I-D.ietf-rmcat-eval-test]. Evaluation results of NADA over WiFi-based test cases as defined in [I-D.ietf-rmcat-wireless-tests] are presented in [IETF-93]. These simulation-based evaluations have shown that NADA flows can obtain their fair share of bandwidth when competing against each other. They typically adapt fast in reaction to the arrival and departure of other flows, and can sustain a reasonable throughput when competing against loss-based TCP flows.

[IETF-90] describes the implementation and evaluation of NADA in a lab setting. Preliminary evaluation results of NADA in single-flow and multi-flow test scenarios have been presented in [IETF-91].

A reference implementation of NADA has been carried out by modifying the WebRTC module embedded in the Mozilla open source browser. Presentations from [IETF-103] and [IETF-105] document real-world evaluations of the modified browser driven by NADA. The experimental setting involve remote connections with endpoints over either home or enterprise wireless networks. These evaluations validate the effectiveness of NADA flows in recovering quickly from throughput drops caused by intermittent delay spikes over the last-hop wireless connections.

8. Suggested Experiments

NADA has been extensively evaluated under various test scenarios, including the collection of test cases specified by [I-D.ietf-rmcat-eval-test] and the subset of WiFi-based test cases in [I-D.ietf-rmcat-wireless-tests]. Additional evaluations have been carried out to characterize how NADA interacts with various active queue management (AQM) schemes such as RED, CoDel, and PIE. Most of these evaluations have been carried out in simulators. A few key test cases have been evaluated in lab environments with implementations embedded in video conferencing clients. It is strongly recommended to carry out implementation and experimentation of NADA in real-world settings. Such exercise will provide insights on how to choose or automatically adapt the values of the key algorithm parameters (see list in Figure 3) as discussed in Section 6.

Additional experiments are suggested for the following scenarios and preferably over real-world networks:

o Experiments reflecting the setup of a typical WAN connection.

o Experiments with ECN marking capability turned on at the network for existing test cases.

o Experiments with multiple NADA streams bearing different user-
   specified priorities.

o Experiments with additional access technologies, especially over
   cellular networks such as 3G/LTE.

o Experiments with various media source contents, including audio
   only, audio and video, and application content sharing (e.g.,
   slide shows).

9. IANA Considerations

   This document makes no request of IANA.

10. Security Considerations

   The rate adaptation mechanism in NADA relies on feedback from the
   receiver. As such, it is vulnerable to attacks where feedback
   messages are hijacked, replaced, or intentionally injected with
   misleading information resulting in denial of service, similar to
   those that can affect TCP. It is therefore RECOMMENDED that the RTCP
   feedback message is at least integrity checked. In addition,
   [I-D.ietf-avtcore-cc-feedback-message] discusses the potential risk
   of a receiver providing misleading congestion feedback information
   and the mechanisms for mitigating such risks.

   The modification of sending rate based on send-side rate shaping
   buffer may lead to temporary excessive congestion over the network in
   the presence of a unresponsive video encoder. However, this effect
   can be mitigated by limiting the amount of rate modification
   introduced by the rate shaping buffer, bounding the size of the rate
   shaping buffer at the sender, and maintaining a maximum allowed
   sending rate by NADA.

11. Acknowledgments

   The authors would like to thank Randell Jesup, Luca De Cicco, Piers
   O'Hanlon, Ingemar Johansson, Stefan Holmer, Cesar Ilharco Magalhaes,
   Safiqul Islam, Michael Welzl, Mirja Kuhlewind, Karen Elisabeth Egede
   Nielsen, Julius Flohr, Roland Bless, Andreas Smas, and Martin
   Stiemerling for their valuable review comments and helpful input to
   this specification.

12. Contributors

   The following individuals have contributed to the implementation and
   evaluation of the proposed scheme, and therefore have helped to
   validate and substantially improve this specification.

Paul E. Jones <paulej@packetizer.com> of Cisco Systems
implemented an early version of the NADA congestion control scheme
and helped with its lab-based testbed evaluations.

Jiantao Fu <jianfu@cisco.com> of Cisco Systems helped with the
implementation and extensive evaluation of NADA both in Mozilla
web browsers and in earlier simulation-based evaluation efforts.

Stefano D'Aronco <stefano.daronco@geod.baug.ethz.ch> of ETH Zurich
(previously at Ecole Polytechnique Federale de Lausanne when
contributing to this work) helped with implementation and
evaluation of an early version of NADA in [ns-3].

Charles Ganzhorn <charles.ganzhorn@gmail.com> contributed to the
testbed-based evaluation of NADA during an early stage of its
development.

13. References

13.1. 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>.

[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,
           <https://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, <https://www.rfc-editor.org/info/rfc3550>.

[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,
           <https://www.rfc-editor.org/info/rfc5348>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P.,
           and K. Carlberg, "Explicit Congestion Notification (ECN)
           for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August
           2012, <https://www.rfc-editor.org/info/rfc6679>.

   [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>.

13.2. Informative References

   [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.

   [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.

   [I-D.ietf-avtcore-cc-feedback-message]
              Sarker, Z., Perkins, C., Singh, V., and M. Ramalho, "RTP
              Control Protocol (RTCP) Feedback for Congestion Control",
              draft-ietf-avtcore-cc-feedback-message-04 (work in
              progress), July 2019.

   [I-D.ietf-rmcat-cc-codec-interactions]
              Zanaty, M., Singh, V., Nandakumar, S., and Z. Sarker,
              "Congestion Control and Codec interactions in RTP
              Applications", draft-ietf-rmcat-cc-codec-interactions-02
              (work in progress), March 2016.

   [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.

   [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., Johansson, I., Zhu, X., Fu, J., Tan, W., and
              M. Ramalho, "Evaluation Test Cases for Interactive Real-
              Time Media over Wireless Networks", draft-ietf-rmcat-
              wireless-tests-08 (work in progress), July 2019.

[IETF-103]
          Zhu, X., Pan, R., Ramalho, M., Mena, S., Jones, P., Fu,
          J., and S. D'Aronco, "NADA Implementation in Mozilla
          Browser", November 2018,
          <https://datatracker.ietf.org/meeting/103/materials/
          slides-103-rmcat-nada-implementation-in-mozilla-browser-
          00>.

[IETF-105]
          Zhu, X., Pan, R., Ramalho, M., Mena, S., Jones, P., Fu,
          J., and S. D'Aronco, "NADA Implementation in Mozilla
          Browser and Draft Update", July 2019,
          <https://datatracker.ietf.org/meeting/105/materials/
          slides-105-rmcat-nada-update-02.pdf>.

[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>.

[IETF-93] Zhu, X., Pan, R., Ramalho, M., Mena, S., Ganzhorn, C.,
          Jones, P., D'Aronco, S., and J. Fu, "Updates on NADA",
          July 2015, <https://www.ietf.org/proceedings/93/slides/
          slides-93-rmcat-0.pdf>.

[IETF-95] Zhu, X., Pan, R., Ramalho, M., Mena, S., Jones, P., Fu,
          J., D'Aronco, S., and C. Ganzhorn, "Updates on NADA:
          Stability Analysis and Impact of Feedback Intervals",
          April 2016, <https://www.ietf.org/proceedings/95/slides/
          slides-95-rmcat-5.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/>.

[ns3-rmcat]
          Fu, J., Mena, S., and X. Zhu, "NS3 open source module of
          IETF RMCAT congestion control protocols", November 2017,
          <https://github.com/cisco/ns3-rmcat>.

   [RFC5450] Singer, D. and H. Desineni, "Transmission Time Offsets in
              RTP Streams", RFC 5450, DOI 10.17487/RFC5450, March 2009,
              <https://www.rfc-editor.org/info/rfc5450>.

   [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,
              <https://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,
              <https://www.rfc-editor.org/info/rfc6817>.

   [RFC7567] Baker, F., Ed. and G. Fairhurst, Ed., "IETF
              Recommendations Regarding Active Queue Management",
              BCP 197, RFC 7567, DOI 10.17487/RFC7567, July 2015,
              <https://www.rfc-editor.org/info/rfc7567>.

   [RFC8033] Pan, R., Natarajan, P., Baker, F., and G. White,
              "Proportional Integral Controller Enhanced (PIE): A
              Lightweight Control Scheme to Address the Bufferbloat
              Problem", RFC 8033, DOI 10.17487/RFC8033, February 2017,
              <https://www.rfc-editor.org/info/rfc8033>.

   [RFC8290] Hoeiland-Joergensen, T., McKenney, P., Taht, D., Gettys,
              J., and E. Dumazet, "The Flow Queue CoDel Packet Scheduler
              and Active Queue Management Algorithm", RFC 8290,
              DOI 10.17487/RFC8290, January 2018,
              <https://www.rfc-editor.org/info/rfc8290>.

   [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>.

   [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.

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. It needs to be acknowledged that NADA has not
yet been tested with non-probabilistic ECN marking behaviors.

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_curr. 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
[RFC7567]. 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_curr 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.

   Calculation of the marking probability involves the following steps:

```
    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_curr 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 *
* Pending affiliation change.

Email: rong.pan@gmail.com

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

Phone: +1 919 476 2038
Email: mar42@cornell.edu

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

Email: semena@cisco.com