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 August 7, 2019.
Table of Contents

1. Introduction ........................................................................................................... 3
2. Terminology ........................................................................................................... 3
3. System Overview .................................................................................................. 3
4. Core Congestion Control Algorithm .................................................................... 5
   4.1. Mathematical Notations .................................................................................. 5
   4.2. Receiver-Side Algorithm ................................................................................ 8
   4.3. Sender-Side Algorithm .................................................................................. 10
5. Practical Implementation of NADA ........................................................................ 13
   5.1. Receiver-Side Operation ................................................................................ 13
   5.1.1. Estimation of one-way delay and queuing delay ........................................... 13
   5.1.2. Estimation of packet loss/marking ratio ....................................................... 13
   5.1.3. Estimation of receiving rate ......................................................................... 14
   5.2. Sender-Side Operation ................................................................................... 14
   5.2.1. Rate shaping buffer ..................................................................................... 15
   5.2.2. Adjusting video target rate and sending rate .............................................. 16
   5.3. Feedback Message Requirements ................................................................... 16
6. Discussions and Further Investigations ................................................................. 17
   6.1. Choice of delay metrics .................................................................................. 17
   6.2. Method for delay, loss, and marking ratio estimation ....................................... 18
   6.3. Impact of parameter values ............................................................................ 18
   6.4. Sender-based vs. receiver-based calculation ................................................... 19
   6.5. Incremental deployment ................................................................................ 19
7. Implementation Status ........................................................................................... 20
8. Suggested Experiments ......................................................................................... 20
9. IANA Considerations ............................................................................................ 21
10. Security Considerations ...................................................................................... 21
11. Acknowledgments ............................................................................................... 21
12. References .......................................................................................................... 21
   12.1. Normative References .................................................................................. 21
   12.2. Informative References ............................................................................... 22
Appendix A. Network Node Operations ................................................................. 24
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. 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 standardized RTCP feedback message requires future work 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 abrevity.

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

Figure 1: System Overview

- Media encoder with rate control capabilities. It encodes raw
  media (audio and video) frames into compressed bitstream which is
  later packetized into RTP packets. As discussed in
  [I-D.ietf-rmcat-video-traffic-model], 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.

- 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.

- 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.

- 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 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:

- **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.

- **Gradual rate update:** in the presence of non-zero aggregate congestion signal, the sending rate is adjusted in reaction to both its value \( x_{\text{curr}} \) and its change in value \( x_{\text{diff}} \).

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. In the experimental phase of this draft, additional studies in real-world settings will gather further learnings on how to choose and adapt these parameter values in practical deployment.
<table>
<thead>
<tr>
<th>Notation</th>
<th>Variable Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>t_curr</td>
<td>Current timestamp</td>
</tr>
<tr>
<td>t_last</td>
<td>Last time sending/receiving a feedback</td>
</tr>
<tr>
<td>delta</td>
<td>Observed interval between current and previous feedback reports: delta = t_curr-t_last</td>
</tr>
<tr>
<td>r_ref</td>
<td>Reference rate based on network congestion</td>
</tr>
<tr>
<td>r_send</td>
<td>Sending rate</td>
</tr>
<tr>
<td>r_recv</td>
<td>Receiving rate</td>
</tr>
<tr>
<td>r_vin</td>
<td>Target rate for video encoder</td>
</tr>
<tr>
<td>r_vout</td>
<td>Output rate from video encoder</td>
</tr>
<tr>
<td>d_base</td>
<td>Estimated baseline delay</td>
</tr>
<tr>
<td>d_fwd</td>
<td>Measured and filtered one-way delay</td>
</tr>
<tr>
<td>d_queue</td>
<td>Estimated queuing delay</td>
</tr>
<tr>
<td>d_tilde</td>
<td>Equivalent delay after non-linear warping</td>
</tr>
<tr>
<td>p_mark</td>
<td>Estimated packet ECN marking ratio</td>
</tr>
<tr>
<td>p_loss</td>
<td>Estimated packet loss ratio</td>
</tr>
<tr>
<td>x_curr</td>
<td>Aggregate congestion signal</td>
</tr>
<tr>
<td>x_prev</td>
<td>Previous value of aggregate congestion signal</td>
</tr>
<tr>
<td>x_diff</td>
<td>Change in aggregate congestion signal w.r.t. its previous value: x_diff = x_curr - x_prev</td>
</tr>
<tr>
<td>rmode</td>
<td>Rate update mode: (0 = accelerated ramp-up; 1 = gradual update)</td>
</tr>
<tr>
<td>gamma</td>
<td>Rate increase multiplier in accelerated ramp-up mode</td>
</tr>
<tr>
<td>loss_int</td>
<td>Measured average loss interval in packet count</td>
</tr>
<tr>
<td>loss_exp</td>
<td>Threshold value for setting the last observed packet loss to expiration</td>
</tr>
<tr>
<td>rtt</td>
<td>Estimated round-trip-time at sender</td>
</tr>
<tr>
<td>buffer_len</td>
<td>Rate shaping buffer occupancy measured in bytes</td>
</tr>
</tbody>
</table>

Figure 2: List of variables.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Parameter Name</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRIO</td>
<td>Weight of priority of the flow</td>
<td>1.0</td>
</tr>
<tr>
<td>RMIN</td>
<td>Minimum rate of application supported by media encoder</td>
<td>150 Kbps</td>
</tr>
<tr>
<td>RMAX</td>
<td>Maximum rate of application supported by media encoder</td>
<td>1.5 Mbps</td>
</tr>
<tr>
<td>XREF</td>
<td>Reference congestion level</td>
<td>10ms</td>
</tr>
<tr>
<td>KAPPA</td>
<td>Scaling parameter for gradual rate update calculation</td>
<td>0.5</td>
</tr>
<tr>
<td>ETA</td>
<td>Scaling parameter for gradual</td>
<td>2.0</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Default Value</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>TAU</td>
<td>Upper bound of RTT in gradual rate update calculation</td>
<td>500ms</td>
</tr>
<tr>
<td>DELTA</td>
<td>Target feedback interval</td>
<td>100ms</td>
</tr>
<tr>
<td>LOGWIN</td>
<td>Observation window in time for calculating packet summary</td>
<td>500ms</td>
</tr>
<tr>
<td>QEPS</td>
<td>Threshold for determining queuing delay build up at receiver</td>
<td>10ms</td>
</tr>
<tr>
<td>DFILT</td>
<td>Bound on filtering delay</td>
<td>120ms</td>
</tr>
<tr>
<td>GAMMA_MAX</td>
<td>Upper bound on rate increase ratio for accelerated ramp-up</td>
<td>0.5</td>
</tr>
<tr>
<td>QBOUND</td>
<td>Upper bound on self-inflicted queuing delay during ramp up</td>
<td>50ms</td>
</tr>
<tr>
<td>MULTILOSS</td>
<td>Multiplier for self-scaling the expiration threshold of the last observed loss (loss_exp) based on measured average loss interval (loss_int)</td>
<td>7.0</td>
</tr>
<tr>
<td>QTH</td>
<td>Delay threshold for invoking non-linear warping</td>
<td>50ms</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>Scaling parameter in the exponent of non-linear warping</td>
<td>0.5</td>
</tr>
<tr>
<td>PLRREF</td>
<td>Reference packet loss ratio</td>
<td>0.01</td>
</tr>
<tr>
<td>PMRREF</td>
<td>Reference packet marking ratio</td>
<td>0.01</td>
</tr>
<tr>
<td>DLOSS</td>
<td>Reference delay penalty for loss when packet loss ratio is at PLRREF</td>
<td>10ms</td>
</tr>
<tr>
<td>DMARK</td>
<td>Reference delay penalty for ECN marking when packet marking is at PMRREF</td>
<td>2ms</td>
</tr>
<tr>
<td>FPS</td>
<td>Frame rate of incoming video</td>
<td>30</td>
</tr>
<tr>
<td>BETA_S</td>
<td>Scaling parameter for modulating outgoing sending rate</td>
<td>0.1</td>
</tr>
<tr>
<td>BETA_V</td>
<td>Scaling parameter for modulating video encoder target rate</td>
<td>0.1</td>
</tr>
<tr>
<td>ALPHA</td>
<td>Smoothing factor in exponential smoothing of packet loss and marking ratios</td>
<td>0.1</td>
</tr>
</tbody>
</table>

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 \( d_{\text{base}} = +\infinity \)
- set \( p_{\text{loss}} = 0 \)
- set \( p_{\text{mark}} = 0 \)
- set \( r_{\text{recv}} = 0 \)
- set both \( t_{\text{last}} \) and \( t_{\text{curr}} \) as current time in milliseconds

On receiving a media packet:

- obtain current timestamp \( t_{\text{curr}} \) from system clock
- obtain from packet header sending time stamp \( t_{\text{sent}} \)
- obtain one-way delay measurement: \( d_{\text{fwd}} = t_{\text{curr}} - t_{\text{sent}} \)
- update baseline delay: \( d_{\text{base}} = \min(d_{\text{base}}, d_{\text{fwd}}) \)
- update queuing delay: \( d_{\text{queue}} = d_{\text{fwd}} - d_{\text{base}} \)
- update packet loss ratio estimate \( p_{\text{loss}} \)
- update packet marking ratio estimate \( p_{\text{mark}} \)
- update measurement of receiving rate \( r_{\text{recv}} \)

On time to send a new feedback report (\( t_{\text{curr}} - t_{\text{last}} > \text{DELTA} \)):

- calculate non-linear warping of delay \( d_{\tilde{t}} \) if packet loss exists
- calculate current aggregate congestion signal \( x_{\text{curr}} \)
- determine mode of rate adaptation for sender: \( r_{\text{mode}} \)
- send feedback containing values of: \( r_{\text{mode}}, x_{\text{curr}}, \) and \( r_{\text{recv}} \)
- update \( t_{\text{last}} = t_{\text{curr}} \)

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 \( \text{loss}_{\text{exp}} \) (measured in terms of packet counts), the estimated queuing delay follows a non-linear warping:
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 \( QTH \) may need to be tuned for different operational environments. 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 \( \text{loss}_{\text{exp}} \) is configured to self-scale with the average packet loss interval \( \text{loss}_{\text{int}} \) with a multiplier \( \text{MULTILOSS} \):

\[
\text{loss}_{\text{exp}} = \text{MULTILOSS} \times \text{loss}_{\text{int}}.
\]

Estimation of the average loss interval \( \text{loss}_{\text{int}} \), 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 \( \text{d}_{\text{tilde}} \) and non-warped \( \text{d}_{\text{queue}} \) values of the queuing delay during the transitional period lasting for the duration of \( \text{loss}_{\text{int}} \).

The aggregate congestion signal is:

\[
x_{\text{curr}} = \text{d}_{\text{tilde}} + \text{DMARK} \times \frac{\text{p}_{\text{mark}}}{\text{PMRREF}} \times \text{DMARK} + \text{DLOSS} \times \frac{\text{p}_{\text{loss}}}{\text{PLRREF}} \times \text{DLOSS}.
\]
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 \( x_{\text{curr}} \) reduces to the observed queuing delay \( d_{\text{queue}} \). 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:

- No recent packet losses within the observation window LOGWIN; and
- No build-up of queuing delay: \( d_{\text{fwd}} - d_{\text{base}} < \text{QEPS} \) 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:

\[
\text{gamma} = \min\left(\frac{\text{QBOUND}}{\rho + \text{RTT} + \text{DFILT}}\right) \quad (3)
\]
\[
r_{\text{ref}} = \max\left(r_{\text{ref}}, (1 + \text{gamma}) \times r_{\text{recv}}\right) \quad (4)
\]

The rate increase multiplier \(\text{gamma}\) is calculated as a function of:
- upper bound of self-inflicted queuing delay (QBOUND),
- round-trip-time (rtt),
- target feedback interval (DELTA)
- 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}  \\
\tau_{ref} \equiv r_{ref} - KAPPA*-------------*r_{ref}  \\
\tau_{tau} \tau_{tau}

\delta x_{diff}  \\
\tau_{ref} \equiv r_{ref} - KAPPA*ETA*------------*r_{ref}  \\
\tau_{tau} \tau_{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, as specified in [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 2Mbps for RMAX.

As mentioned in the sender-side algorithm, the final rate is 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]. Instead of relying on RTP timestamps, the NADA sender generates its own timestamp based on local system clock and embeds 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. All delay estimations are based on sender timestamps with a recommended granularity of 100 microseconds or higher.

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, instead of simply calculating the value of base delay using \( d_{\text{base}} = \min(d_{\text{base}}, d_{\text{fwd}}) \), as expressed in Section 5.1, current implementation 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 \( p_{\text{inst}} \) 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 $p_{\text{loss}}$ is obtained after exponential smoothing:

$$p_{\text{loss}} = \text{ALPHA} \cdot p_{\text{inst}} + (1-\text{ALPHA}) \cdot p_{\text{loss}}.$$  (10)

The filtered result is reported back to the sender as the observed packet loss ratio $p_{\text{loss}}$.

Estimation of packet marking ratio $p_{\text{mark}}$ 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 $r_{\text{recv}}$. 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 ($r_{\text{recv}}$) 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 $r_{\text{ref}}$ as specified in Section 4.3. It further adjusts both the target rate for the live video encoder $r_{\text{vin}}$ and the sending rate $r_{\text{send}}$ over the network based on the updated value of $r_{\text{ref}}$ and rate shaping buffer occupancy $\text{buffer}_\text{len}$.

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.
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 \( r_{\text{ref}} \) based on the level of occupancy in the rate shaping buffer:

\[
r_{v\text{in}} = r_{\text{ref}} - B\text{ETA}_V \times 8 \times \text{buffer}\_\text{len} \times \text{FPS}.
\]  

(11)

The actual sending rate \( r_{\text{send}} \) is regulated in a similar fashion:

\[
r_{\text{send}} = r_{\text{ref}} + B\text{ETA}_S \times 8 \times \text{buffer}\_\text{len} \times \text{FPS}.
\]  

(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 \( r_{\text{ref}} \).

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 \( 8 \times \text{buffer}\_\text{len} \times \text{FPS} \), where FPS stands for the frame rate of the video. The scaling parameters \( B\text{ETA}_V \) and \( B\text{ETA}_S \) can be tuned to balance between the competing goals of maintaining a small rate shaping buffer and deviating from the reference rate point.

5.3. Feedback Message Requirements

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

- 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).

- Aggregated congestion signal (x_curr): the most recently updated value, calculated by the receiver according to Section 4.2. This information is 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.

- 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. 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.

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 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 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 --- approximately 1.6% overhead for a 1 Mbps flow. Furthermore, both simulation studies and frequency-domain analysis 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 may need to be tuned for different operational environments (e.g., over wired vs. wireless connections). 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, 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.

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). 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 ($d_{queue}$), packet loss ratio ($p_{loss}$), packet marking ratio ($p_{mark}$), receiving rate ($r_{recv}$), and recommended rate adaptation mode ($r_{mode}$).

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]. Evaluation results of the current draft over several test cases in [I-D.ietf-rmcat-wireless-tests] have been presented at [IETF-93]. An open source implementation of NADA as part of a ns-3 module is available at [ns3-rmcat].

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. 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 to 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:

- Experiments reflecting the set up of a typical WAN connection.
- Experiments with ECN marking capability turned on at the network for existing test cases.
Experiments with multiple RMCAT streams bearing different user-specified priorities.

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

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

TBD

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 various valuable review comments and feedback. Thanks to Charles Ganzhorn for contributing to the testbed-based evaluation of NADA during an early stage of its development.

12. References

12.1. Normative References


12.2. Informative References


[I-D.ietf-rmcat-wireless-tests]


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:
\[
\begin{align*}
    p &= 0, & \text{if } q < q_{lo}; \\
    \quad & q_{avg} - q_{lo}, & \text{if } q_{lo} \leq q < q_{hi}; \\
    \quad & q_{hi} - q_{lo}, & \text{if } q \geq q_{hi}.
\end{align*}
\]
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:

\[
p = \begin{cases} 
0, & \text{if } b-b_{tk} < b_{lo}; \\
\frac{b-b_{tk}-b_{lo}}{b_{hi}-b_{lo}} \times p_{max}, & \text{if } b_{lo} \leq b-b_{tk} < b_{hi}; \\
1, & \text{if } b-b_{tk} \geq b_{hi}.
\end{cases}
\]

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: 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