Network Working Group
Internet Draft
Intended Status: Informational
Expires: March 16, 2015

X. Zhu, R. Pan
M. A. Ramalho, S. M. de la Cruz
C. Ganzhorn, P. E. Jones
Cisco Systems
S. DAronco
Ecole Polytechnique Federale de Lausanne

September 12, 2014

NADA: A Unified Congestion Control Scheme for Real-Time Media
draft-zhu-rmcat-nada-04

Abstract

This document describes a scheme named network-assisted dynamic adaptation (NADA), a novel congestion control approach 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.

We present here the overall system architecture, recommended behaviors at the sender and the receiver, as well as expected network node operations. Results from extensive simulation studies of the proposed scheme are available upon request.

Status of this Memo

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 fast 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 losses, queuing delay, and explicit congestion notification (ECN) markings.

Based on the above considerations, we present a scheme named network-assisted dynamic adaptation (NADA). The proposed design benefits from explicit congestion control signals (e.g., ECN markings) from the network, and remains compatible in the presence of implicit signals (delay or loss) only. In addition, it supports weighted bandwidth sharing among competing video flows.

This documentation describes the overall system architecture, recommended designs at the sender and receiver, as well as expected network nodes operations. The signaling mechanism consists of standard RTP timestamp [RFC3550] and standard RTCP feedback reports.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. System Model

The system consists of the following elements:

* Incoming media stream, in the form of consecutive raw video frames and audio samples;

* Media encoder with rate control capabilities. It takes the incoming media stream and encodes it to an RTP stream at a target bit rate R_v. Note that the actual output rate from the encoder R_o may fluctuate randomly around the target R_v. Also, the encoder can only change its rate at rather coarse time intervals, e.g., once every 0.5 seconds.

* RTP sender, responsible for calculating the target bit rate R_n based on network congestion signals (delay or ECN marking reports from the receiver), and for regulating the actual sending rate R_s accordingly. A rate shaping buffer is employed

to absorb the instantaneous difference between video encoder output rate R_v and sending rate R_s. The buffer size L_s, together with R_n, influences the calculation of actual sending rate R_s and video encoder target rate R_v. The RTP sender also generates RTP timestamp in outgoing packets.

* RTP receiver, responsible for measuring and estimating end-to-end delay based on sender RTP timestamp. In the presence of packet losses and ECN markings, it also keeps track of packet loss and ECN marking ratios. It calculates the equivalent delay x_n that accounts for queuing delay, ECN marking, and packet losses, as well as the derivative (i.e., slope of change) of this congestion signal as x'_n. The receiver feeds both information (x_n and x'_n) back to the sender via periodic RTCP reports.

* 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 RED-based ECN marking, and PCN marking using a token bucket algorithm.

In the following, we will elaborate on the respective operations at the network node, the receiver, and the sender.

4. Network Node Operations

We consider three variations of queue management behavior at the network node, leading to either implicit or explicit congestion signals.

4.1 Default behavior of drop tail

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

4.2 ECN marking

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

* upon packet arrival, update smoothed queue size q_avg as:

$$q\_avg = alpha*q + (1-alpha)*q\_avg.$$

The smoothing parameter alpha is a value between 0 and 1. A value of alpha=1 corresponds to performing no smoothing at all.

* calculate marking probability p as:

$$p = 0, \text{ if } q < q\_lo;$$

$$p = p\_max*\frac{q\_avg - q\_lo}{q\_hi - q\_lo}, \text{ if } q\_lo <= q < q\_hi;$$

$$p = 1, \text{ if } q >= q\_hi.$$

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

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

4.3 PCN marking

As a more advanced feature, we also envision network nodes which support PCN marking based on virtual queues. In such a case, the marking probability of the ECN bit in the IP packet header is calculated as follows:

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

* update token level b_tk;

* calculate the marking probability as:

$$p = 0, \text{ if } b-b\_tk < b\_lo;$$

$$p = p\_max* \frac{b-b\_tk-b\_lo}{b\_hi-b\_lo}, \text{ if } b\_lo<= b-b\_tk <b\_hi;$$

$$p = 1, \text{ if } b-b\_tk>=b\_hi.$$

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 as r=gamma*C, where gamma<1 is the target utilization ratio and C designates link capacity. The maximum marking probability is p_max.

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

4.4 Comments and Discussions

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

The sender behavior stays the same in the presence of all types of congestion signals: delay, loss, ECN marking due to either RED/ECN or PCN algorithms. This unified approach allows a graceful transition of the scheme as the level of congestion in the network shifts dynamically between different regimes.

5. Receiver Behavior

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

5.1 Monitoring per-packet statistics

Upon receipt of each packet, the receiver calculates one-way delay as the difference between sender and receiver timestamps:

$$x_n = t_{r,n} - t_{s,n}.$$

It also maintains its estimate of baseline delay, d_f, as the minimum value of previously observed x_n's over a relatively longer period. This assumes that that sending and receiving clocks are either well-synchronized, or have a relatively stable offset. In our implementation, the baseline delay estimation is updated once every 10 minutes.

Correspondingly, the queuing delay experienced by the packet n is estimated as:

$$d_n = x_n - d_f.$$

In addition, the receiver keeps track of both packet loss ratios as p_L via detection of gaps in the packet sequence numbers, and ECN marking ratios as p_M.

5.2 Aggregating congestion signals

The receiver aggregates all three forms of congestion signal in terms of an equivalent delay:

$$x_n = d_n + p_M d_M + p_L d_L, \qquad (1)$$

where d_M is a prescribed fictitious delay value associated with ECN markings (e.g., d_M = 200 ms), and d_L is a prescribed fictitious delay value associated with packet losses (e.g., d_L = 1 second). By introducing a large fictitious delay penalty for ECN marking and packet losses, the proposed scheme leads to low end-to-end actual delays in the presence of such events.

While the value of d_M and d_L are fixed and predetermined in our current design, we also plan to investigate 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).

It should also be noted that in the absence of loss and marking information, the value of x_n falls back to the observed queuing delay d_n for packet n. Our algorithm operates in purely delay-based mode.

5.3 Sending periodic feedback

Periodically, the receiver sends back the most recent value of x_n in RTCP messages, to aid the sender in its calculation of target rate. It also calculates and sends the derivative of x_n as part of the RTCP message:

$$x'_n = \frac{x_n - x_{(n-k)}}{delta}. \qquad (2)$$

Here, the interval between current and previous RTCP messages is denoted as delta, and the corresponding packet indices are n and (n-k), respectively. Typically, the interval between adjacent RTCP receiver reports is on the order of sub-seconds (e.g., 100ms).

The size of acknowledgement packets are typically on the order of tens of bytes, and are significantly smaller than average video packet sizes. Therefore, the bandwidth overhead of the receiver acknowledgement stream is sufficiently low.

5.4 Discussions on delay metrics

Our current design works with relative 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 longer time horizon and subtract that out from the observed absolute OWD value. Such an approach cancels out the fixed clock difference from the sender and receiver clocks, and has been widely adopted by other delay-based congestion control approaches such as LEDBAT [RFC6817]. As discussed in [RFC6817], the time horizon for tracking the minimum OWD needs to be chosen with care: long enough for an opportunity to observe the minimum OWD with zero queuing delay along the path, and sufficiently short so as to timely reflect "true" changes in minimum OWD introduced by route changes and other rare events.

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

Alternatively, one could move the per-packet statistical handling to the sender instead, and use RTT in lieu of OWD, assuming that per-packet ACKs are available. The main drawback of this latter approach, on the other hand, is that the scheme will be confused by congestion in the reverse direction.

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

6. Sender Behavior

Figure 1 provides a more detailed view of the NADA sender. Upon receipt of an RTCP report from the receiver, the NADA sender updates its calculation of the reference rate R_n. It further adjusts both the target rate for the live video encoder R_v and the sending rate R_s over the network based on the updated value of R_n, as well as the size of the rate shaping buffer.

```
                    --------------------
                    |                  |
                    |  Reference Rate  | <---- RTCP report
                    |  Calculator      |
                    |                  |
                    --------------------
                            |
                            | R_n
                            |
                ---------------------------
               |                          |
               |                          |
              \ /                        \ /
    --------------------           -----------------
    |                  |           |               |
    |  Video Target    |           | Sending Rate  |
    |  Rate Calculator |           | Calculator    |
    |                  |           |               |
    --------------------           -----------------
       |        /|\                   /|\      |
    R_v|         |                     |       |
       |         -----------------------       |
       |                     |                 | R_s
    ------------             |L_s              |
    |          |             |                 |
    |          |  R_o    --------------       \|/
    |  Encoder |---------->   | | | | | --------------->
    |          |              | | | | |     video packets
    ------------         --------------
                         Rate Shaping Buffer
```

Figure 1 NADA Sender Structure

The following sections describe these modules in further details, and explain how they interact with each other.

6.1 Video encoder rate control

The video encoder rate control procedure has the following characteristics:

* Rate changes can happen only at large intervals, on the order of seconds.

* Given a target rate R_o, the encoder output rate may randomly fluctuate around it.

* The encoder output rate is further constrained by video content complexity. The range of the final rate output is [R_min, R_max]. Note that it’s content-dependent, and may change over time.

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

6.2 Rate shaping buffer

A rate shaping buffer is employed to absorb any instantaneous mismatch between encoder rate output R_o and regulated sending rate R_s. The size of the buffer evolves from time t-tau to time t as:

$$L_s(t) = \max [0, L_s(t-\tau)+(R_o-R_s)*\tau].$$

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 constrain the size of the shaping buffer. It can either deplete it faster by increasing the sending rate R_s, or limit its growth by reducing the target rate for the video encoder rate control R_v.

6.3 Reference rate calculator

The sender initializes the reference rate R_n as R_min. Upon receipt of a new receiver RTCP reports containing values of x_n and x’_n, it updates the rate as:

$$R_n \longleftarrow R_n + \frac{\kappa * \delta_s}{\tau_o^2} * (\theta-(R_n-R_{min})*\hat{x}) \quad (3)$$

where

$$\theta = w*(R_{max} - R_{min})*x_{ref}. \quad (4)$$

$$\hat{x} = x_n + \eta*\tau_o* x'_n \quad (5)$$

In (3), delta_s refers to the time interval between current and previous rate updates. Note that delta_s is the same as the RTCP report interval at the receiver (see delta from (2)) when the backward path is un-congested.

In (4), R_min and R_max denote the content-dependent rate range the encoder can produce. The weight of priority level is w. The reference congestion signal x_ref is chosen so that the maximum rate of R_max can be achieved when x_hat = w*x_ref.

Proper choice of the scaling parameters eta and kappa in (3) and (5) can ensure system stability so long as the RTT falls below the upper bound of tau_o. In our design, tau_o is chosen as 500ms.

The final target rate R_n is clipped within the range of [R_min, R_max].

Note that the sender does not need any explicit knowledge of the management scheme inside the network. Rather, it reacts to the aggregation of all forms of congestion indications (delay, loss, and marking) via the composite congestion signals x_n and x'_n from the receiver in a coherent manner.

6.4 Video target rate and sending rate calculator

The target rate for the live video encoder is updated based on both the reference rate R_n and the rate shaping buffer size L_s, as follows:

```
                              L_s
        R_v = R_n - beta_v * -------.       (6)
                             tau_v
```

Similarly, the outgoing rate is regulated based on both the reference rate R_n and the rate shaping buffer size L_s, such that:

```
                              L_s
        R_s = R_n + beta_s * -------.       (7)
                             tau_v
```

In (6) and (7), the first term indicates the rate calculated from network congestion feedback alone. The second term indicates the influence of the rate shaping buffer. A large rate shaping buffer nudges the encoder target rate slightly below -- and the sending rate slightly above -- the reference rate R_n.

Intuitively, the amount of extra rate offset needed to completely drain the rate shaping buffer within the same time frame of encoder rate adaptation tau_v is given by L_s/tau_v. The scaling parameters beta_v and beta_s can be tuned to balance between the competing goals of maintaining a small rate shaping buffer and deviating the system from the reference rate point.

6.5 Start-up behavior

The rate adaptation algorithm specified by (3)--(5) naturally leads to a linear rate increase at start-up, when queuing delay stays at zero in the beginning:

$$R_n \leftarrow R_n + \frac{\text{kappa}*\text{delta\_s}}{\text{tau\_o}^2} * \text{theta} \qquad (8)$$

Given that theta = w*(R_max - R_min)*x_ref, the speed of increase scales with the value of kappa, weight of priority w, and dynamic range of the flow (R_max - R_min).

In practice, one may desire a more aggressive ramp-up behavior during the start-up period, e.g., by doubling the rate upon the receipt of each new RTCP message which reports on near-zero values of x_n and x’_n.

We note here that design of the start-up behavior can be kept orthogonal to the design of the steady-state rate adaptation behavior. This topic is worthy of further investigation separately.

7. Incremental Deployment

One nice property of proposed design is the consistent video end point behavior irrespective of network node variations. This facilitates gradual, incremental adoption of the scheme.

To start off with, the proposed encoder congestion control mechanism can be implemented without any explicit support from the network, and rely 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.

8. Implementation Status

The proposed NADA scheme has been implemented in the ns-2 simulation platform [ns2]. Extensive simulation evaluations of an earlier version of the draft are documented in [Zhu-PV13]. Evaluation results of current draft over several test cases in [I-D.draft-sarker-rmcat-eval-test] have been presented at the recent IETF meeting [IETF-90].

The scheme has also been implemented in Linux and has been evaluated in a lab setting also described in [IETF-90]. Evaluation results of NADA in single-flow and multi-flow scenarios from this testbed will be disclosed soon.

9. IANA Considerations

There are no actions for IANA.

10. References

10.1 Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

10.2 Informative References

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC2309] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, April 1998.

[RFC6187] S. Shalunov, G. Hazel, J. Iyengar, and M. Kuehlewind, "Low Extra Delay Background Transport (LEDBAT)", RFC 6817, December 2012

[ns2] "The Network Simulator - ns-2", http://www.isi.edu/nsnam/ns/

[Zhu-PV13] Zhu, X. and Pan, R., "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.

[I-D.draft-sarker-rmcat-eval-test] Sarker, Z., Singh, V., Zhu, X., and Ramalho, M., "Test Cases for Evaluating RMCAT Proposals", draft-sarker-rmcat-eval-test-01 (work in progress), June 2014.

[IETF-90] Zhu, X. et al., "NADA Update: Algorithm, Implementa6on, and Test Case Evalua6on Results", presented at IETF 90, https://tools.ietf.org/agenda/90/slides/slides-90-rmcat-6.pdf

Authors' Addresses

Xiaoqing Zhu
Cisco Systems,
12515 Research Blvd.,
Austin, TX 78759, USA
Email: xiaoqzhu@cisco.com

Rong Pan
Cisco Systems
510 McCarthy Blvd,
Milpitas, CA 95134, USA
Email: ropan@cisco.com

Michael A. Ramalho
6310 Watercrest Way Unit 203
Lakewood Ranch, FL, 34202, USA
Email: mramalho@cisco.com

Sergio Mena de la Cruz
EPFL, Quartier de l'Innovation
Batiment E
Ecublens, Vaud 1015, Switzerland
Email: semena@cisco.com

Charles Ganzhorn
7900 International Drive
International Plaza, Suite 400
Bloomington, MN 55425, USA
Email: cganzhor@cisco.com

Paul E. Jones
7025 Kit Creek Rd.
Research Triangle Park, NC 27709, USA
Email: paulej@packetizer.com

Stefano D’Aronco
EPFL STI IEL LTS4
ELD 220 (Batiment ELD), Station 11
CH-1015 Lausanne, Switzerland
Email: stefano.daronco@epfl.ch