Network Working Group
Internet Draft
Intended Status: Informational
Expires: September 27, 2015

X. Zhu, R. Pan
M. A. Ramalho, S. Mena
C. Ganzhorn, P. E. Jones
Cisco Systems
S. De Aronco
Ecole Polytechnique Federale de Lausanne

March 26, 2015

NADA: A Unified Congestion Control Scheme for Real-Time Media
draft-zhu-rmcat-nada-06

Abstract

Network-Assisted Dynamic Adaptation (NADA) is a novel congestion control scheme for interactive real-time media applications, such as video conferencing. In NADA, the sender regulates its sending rate based on either implicit or explicit congestion signaling in a consistent manner. As one example of explicit signaling, NADA can benefit from explicit congestion notification (ECN) markings from network nodes. It also maintains consistent sender behavior in the absence of explicit signaling by reacting to queuing delay and packet loss.

This document describes the overall system architecture for NADA, as well as recommended behavior at the sender and the receiver.

Copyright and License 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.

Based on the above considerations, this document describes a scheme called network-assisted dynamic adaptation (NADA). The NADA design benefits from explicit congestion control signals (e.g., ECN markings) from the network, yet also operates when only implicit congestion indicators (delay and/or loss) are available. In addition, it supports weighted bandwidth sharing among competing video flows.

This documentation describes the overall system architecture, recommended designs at the sender and receiver, as well as expected network node 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 overall system consists of the following elements:

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

* Media encoder with rate control capabilities. It takes the source 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 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 indicators (delay, loss, or ECN marking reports from the receiver), for updating the video encoder with a new target rate R_v, 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 loss and ECN markings, it 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 loss, as well as the derivative (i.e., rate of change) of this congestion signal as x’_n. The receiver feeds both pieces of 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. Note that network node operation is out of scope for the design of NADA.

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

4. NADA Receiver Behavior

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.

4.1 Estimation of one-way delay and queuing delay

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

In mathematical terms, for packet n arriving at the receiver, one-way delay is calculated as:

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

where t_s,n and t_r,n are sender and receiver timestamps, respectively. A real-world implementation should also properly handle practical issues such as wrap-around in the value of x_n, which are omitted from the above simple expression for brevity.

The base delay, d_f, is estimated as the minimum value of previously observed x_n’s over a relatively long period. This assumes that the drift between sending and receiving clocks remains bounded by a small value.

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

$$d_n = x_n - d_f.$$

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. We denote the resulting queuing delay value as d_hat_n.

Our current implementation employs a simple 5-point median filter over per-packet queuing delay estimates, followed by an exponential smoothing filter. We have found such relatively simple treatment to suffice 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.

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.

4.2 Estimation of packet loss/marking ratio

The receiver detects packet losses via gaps in the RTP sequence numbers of received packets. It then calculates instantaneous packet loss ratio as the ratio between the number of missing packets over the number of total transmitted packets in the given time window (e.g., during the most recent 500ms). This instantaneous value is passed over an exponential smoothing filter, and the filtered result is reported back to the sender as the observed packet loss ratio p_L.

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

Estimation of packet marking ratio p_M, when ECN is enabled at bottleneck network nodes along the path, will follow the same procedure as above. Here it is assumed that ECN marking information at the IP header are somehow passed along to the transport layer by the receiving endpoint.

4.3 Non-linear warping of delay

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

Inspired by the delay-adaptive congestion window backoff policy in [Budzisz-TON11] -- the work by itself is a window-based congestion control scheme with fair coexistence with TCP -- we devise the following non-linear warping of estimated queuing delay value:

$$\tilde{d}_n = (\hat{d}_n), \quad \text{if } \hat{d}_n < d_{th};$$

$$\tilde{d}_n = d_{th} \frac{(d_{max} - \hat{d}_n)^4}{(d_{max} - d_{th})^4}, \quad \text{if } d_{th} < \hat{d}_n < d_{max};$$

$$\tilde{d}_n = 0, \quad \text{if } \hat{d}_n > d_{max}.$$

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

When queuing delay is in the range (0, d_th), NADA operates in pure delay-based mode if no losses/markings are present. When queuing delay exceeds d_max, NADA reacts to loss/marking only. In between d_th and d_max, the sending rate will converge and stabilize at an operating point with a fairly high queuing delay and non-zero packet loss ratio.

In our current implementation d_th is chosen as 50ms and d_max is chosen as 400ms. The impact of the choice of d_th and d_max will be investigated in future work.

4.4 Aggregating congestion signals

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

$$x_n = d\_tilde_n + p_M*d_M + p_L*d_L, \qquad (1)$$

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 loss, the proposed scheme leads to low end-to-end actual delay in the presence of such events.

While the value of d_M and d_L 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 being studied.

In the absence of ECN marking from the network, the value of x_n falls back to the observed queuing delay d_n for packet n when queuing delay is low and no packets are lost over a lightly congested path. In that case the algorithm operates in purely delay-based mode.

4.5 Estimating receiving rate

Estimation of receiving rate of the flow is fairly straightforward. NADA maintains a recent observation window of 500ms, and simply divides the total size of packets arriving during that window over the time span. The receiving rate is denoted as R_r.

4.6 Sending periodic feedback

Periodically, the receiver feeds back a tuple of the most recent values of <d_hat_n, x_n, x’_n, R_r> in RTCP feedback messages to aid the sender in its calculation of target rate. The queuing delay value d_hat_n is included along with the composite congestion signal x_n so that the sender can decide whether the network is truly underutilized (see Sec. 6.1.1 Accelerated ramp-up).

The value of x’_n corresponds to the derivative (i.e., rate of change) of the composite congestion signal:

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

where the interval between consecutive RTCP feedback messages is denoted as delta. The packet indices corresponding to the current and previous feedback are n and (n-k), respectively.

The choice of target feedback interval needs to strike the right balance between timely feedback and low RTCP feedback message counts. Through simulation studies and frequency-domain analysis, it was determined that a feedback interval below 250ms will not break up the feedback control loop of the NADA congestion control algorithm. Thus, it is recommended to use a target feed interval of 100ms. This will result in a feedback bandwidth of 16Kbps with 200 bytes per feedback message, less than 0.1% overhead for a 1Mbps flow.

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

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

Alternatively, one could move the per-packet statistical handling to the sender instead and use RTT in lieu of OWD, assuming that per-packet ACKs are available. The main drawback of this latter approach is that the scheme will be confused by congestion 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 at the sender. Therefore, comparing the pros and cons regarding which delay metric to adopt can be kept as an orthogonal direction of investigation.

5. NADA Sender Behavior

Figure 1 provides a 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.

In the following, we describe these modules in further detail, and explain how they interact with each other.

```
                    --------------------
                    |                  |
                    |  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

5.1 Reference rate calculation

The sender initializes the reference rate R_n as R-min by default, or to a value specified by the upper-layer application. [Editor’s note: should proper choice of starting rate value be within the scope of the CC solution?]

The reference rate R_n is calculated based on receiver feedback information regarding queuing delay d_tilde_n, composite congestion signal x_n, its derivative x’_n, as well as the receiving rate R_r. The sender switches between two modes of operation:

* Accelerated ramp up: if the reported queuing delay is close to zero and both values of x_n and x’_n are close to zero, indicating empty queues along the path of the flow and, consequently, underutilized network bandwidth; or

* Gradual rate update: in all other conditions, whereby the receiver reports on a standing or increasing/decreasing queue and/or composite congestion signal.

5.1.1 Accelerated ramp up

In the absence of a non-zero congestion signal to guide the sending rate calculation, the sender needs to ramp up its estimated bandwidth as quickly as possible without introducing excessive queuing delay. Ideally the flow should inflict no more than T_th milliseconds of queuing delay at the bottleneck during the ramp-up process. A typical value of T_th is 50ms.

Note that the sender will be aware of any queuing delay introduced by its rate increase after at least one round-trip time. In addition, the bottleneck bandwidth C is greater than or equal to the receive rate R_r reported from the most recent "no congestion" feedback message. The rate R_n is updated as follows:

```
                                   T_th
    gamma = min [gamma_0, ---------------]      (3)
                                RTT_0+delta_0


    R_n  =  (1+gamma) R_r                     (4)
```

In (3) and (4), the multiplier gamma for rate increase is upper-bounded by a fixed ratio gamma_0 (e.g., 20%), as well as a ratio which depends on T_th, base RTT as measured during the non-congested phase, and target ACK interval delta_0. The rationale behind this is that the rate increase multiplier should decrease with the delay in the feedback control loop, and that RTT_0 + delta_0 provides a worst-case estimate of feedback control delay when the network is not congested.

5.1.2. Gradual rate update

When the receiver reports indicate a standing congestion level, NADA operates in gradual update mode, and calculates its reference rate as:

```
                   kappa * delta_s
    R_n  <-- R_n + ---------------- * (theta-(R_n-R_min)*x_hat) (5)
                      tau_o^2
```

where

```
    theta = w*(R_max - R_min)*x_ref.    (6)

    x_hat = x_n + eta*tau_o* x'_n       (7)
```

In (5), 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 (6), R_min and R_max denote the content-dependent rate range the encoder can produce. The weighting factor reflecting a flow's priority 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 (5) and (7) can ensure system stability so long as the RTT falls below the upper bound of tau_o. The recommended default value of tau_o is chosen as 500ms.

For both modes of operations, the final reference rate R_n is clipped within the range of [R_min, R_max]. Note also 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 explicit markings) via the composite congestion signals x_n and x'_n from the receiver in a coherent manner.

5.2 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.

* The encoder output rate may fluctuate around the target rate R_v.

* 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 is content-dependent and may vary over time.

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.

5.3 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.

5.4 Adjusting video target rate and sending rate

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:

$$R\_v = R\_n - beta\_v * \frac{L\_s}{tau\_v}. \qquad (8)$$

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

$$R\_s = R\_n + beta\_s * \frac{L\_s}{tau\_v}. \qquad (9)$$

In (8) and (9), 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. 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 encoder 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 the ns-2 simulation platform [ns2]. Extensive 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.draft-sarker-rmcat-eval-test] have been presented at recent IETF meetings [IETF-90][IETF-91].

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

8. IANA Considerations

There are no actions for IANA.

9. References

9.1 Normative References

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

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

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

9.2 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.

[RFC6817] Shalunov, S., Hazel, G., Iyengar, J., and Kuehlewind, M., "Low Extra Delay Background Transport (LEDBAT)", RFC 6817, December 2012

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

[Budzisz-TON11] Budzisz, L. et al., "On the Fair Coexistence of Loss- and Delay-Based TCP", IEEE/ACM Transactions on Networking, vol. 19, no. 6, pp. 1811-1824, December 2011.

[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, Implementation, and Test Case Evalua6on Results", presented at IETF 90, https://tools.ietf.org/agenda/90/slides/slides-90-rmcat-6.pdf

[IETF-91] Zhu, X. et al., "NADA Algorithm Update and Test Case Evaluations", presented at IETF 91 Interium, https://datatracker.ietf.org/meeting/91/agenda/rmcat/

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

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

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

A.1 Default behavior of drop tail

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 equivalent delay x_n. No special action is required at network node.

A.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, if q < q_lo;

           q_avg - q_lo
p = p_max*--------------, if q_lo <= q < q_hi;
           q_hi - q_lo

p = 1, if q >= q_hi.
```

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

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

A.3 PCN marking

As a more advanced feature, we also envisage 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:

$$p = 0, \text{ if } b-b_{tk} < b_{lo};$$

$$p = p_{max} \cdot \frac{b-b_{tk}-b_{lo}}{b_{hi}-b_{lo}}, \text{ if } b_{lo} \le b-b_{tk} < b_{hi};$$

$$p = 1, \text{ if } b-b_{tk} \ge b_{hi}.$$

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 $r=\gamma \cdot C$, 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.

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
Cisco Systems
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: charles.ganzhorn@gmail.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