Internet Engineering Task Force
INTERNET-DRAFT
draft-ietf-rmt-bb-webrc-00.ps

RMT WG
M. Luby/Digital Fountain
V. Goyal/Digital Fountain
S. Skaria/UC Irvine & DF
18 October 2001
Expires: April 2002

Wave and Equation Based Rate Control: A massively scalable receiver driven congestion control protocol

Status of this Document

This document is a product of the IETF RMT WG. Comments should be addressed to the authors, or the WG's mailing list at rmt@lbl.gov.

Abstract

This document describes Wave and Equation Based Rate Control, a massively scalable congestion control protocol, hereafter referred to as WEBRC. WEBRC can be used for multi-rate congestion control for multicast sessions that scale to an unlimited number of receivers, as well as for unicast sessions that require minimal support from the sender per receiver.

The sender sends packets to channels that act like rolling waves: At equally spaced intervals of time a channel starts up at a high rate that continually and gradually decreases over a long interval of time. Each WEBRC receiver independently uses a TFRC-like [2] equation-based approach to determine its current target reception rate, and then joins each wave channel either earlier or later in the wave descent in order to increase or decrease the receiver's reception rate.

WEBRC is designed to be fair to TCP flows and other WEBRC flows, while requiring minimal performance from the network or the sender in terms of processing join and leave messages.

Table of Contents

1. Introduction

This document specifies Wave and Equation Based Rate Control (WEBRC). WEBRC is a congestion control mechanism designed to be massively scalable for both multicast and unicast flows operating in a Internet environment and competing with TCP traffic. Instead of specifying a complete protocol, this document simply specifies a congestion control mechanism that could be used in a protocol instantiation such as ALC [3] to provide end-to-end congestion control. This document does not discuss reliability or other implementation-related issues that may be associated with a complete protocol instantiation.

WEBRC is a receiver-driven congestion control protocol in the spirit of [8] and [1]. This means that all measurements and decisions to raise or lower the reception rate are made by each individual receiver, and these decisions are acted upon by sending join and leave messages for channels to the network in the case of multicast and by sending join and leave messages for channels directly to the sender in the case of unicast.

Receivers using WEBRC adjust their reception rate independently of other concerns such as reliability. This is different from TCP, where the congestion control protocol and the reliability protocol are intricately interwoven with one another. Thus, WEBRC takes the same basic approach as TFRC [2]. Reliability can be added by independent means, such as by the use of FEC codes as described in [5] and specified in [6].

WEBRC uses many of the mechanisms of TFRC adjusted to work properly for WEBRC. In particular, each WEBRC receiver measures parameters that are plugged into a TCP-like equation to compute the receiver target reception rate, and adjusts its reception rate up and down to closely approximate the target reception rate. The sender sends packets to multiple channels called wave channels. Each wave channel follows the same pattern of packet rate transmission spread out over equally spaced intervals of time. The pattern of each wave is that it starts at a high rate that decreases gradually and continually over a long interval of time. (Picture an infinite sequence of rolling waves.) The receiver increases its reception rate by joining the next wave channel earlier than it joined the previous wave channel, and the receiver decreases its reception rate by joining the next wave channel later than it joined the previous wave channel.

WEBRC is designed to be reasonably fair when competing for bandwidth with TCP flows, where a flow is “reasonably fair” if its sending rate is generally within a factor of two of the sending rate of a TCP flow under the same conditions. However WEBRC has a much lower variation of throughput over time compared to TCP, which makes it more suitable for applications such as telephony or streaming media where a relatively smooth sending rate is of importance. Furthermore, WEBRC is designed to be massively scalable in the sense that the sender is insensitive to the number of receivers joined to a multicast session, and there is a minimal amount of work specific to WEBRC that the sender needs to do for each receiver joined to a unicast session.

The penalty of having smoother throughput than TCP while competing fairly for bandwidth is that WEBRC responds slower than TCP to changes in available bandwidth.

WEBRC is designed for applications that use a fixed packet size, and vary their packet reception rate in response to congestion.

The receiver measures and performs the calculation of congestion control parameters (e.g., the average loss probability) and makes decisions on how to increase or decrease its rate based on these parameters. This is well suited to an application where the sender is handling many concurrent connections, and the receiver has more memory and CPU cycles available for computation. In addition, a receiver-based mechanism is more suitable as a building block for multicast congestion control.

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 RFC2119.

1.1. Related Documents

As described in RFC3048, WEBRC is a building block that is intended to be used, in conjunction with other building blocks, to help specify a protocol instantiation. In particular, WEBRC is a suitable congestion control building block to be used by ALC [3]. The information required to be carried in each packet by WEBRC can be carried in the Congestion Control Information field in the LCT header [4].

1.2. Environmental Requirements and Considerations

WEBRC is intended to be a congestion control scheme that can be used in a complete protocol instantiation that delivers objects and streams (both reliable content delivery and streaming of multimedia information).

WEBRC is most applicable for sessions that deliver a substantial amount of data, i.e., in length from hundreds of kilobytes to many gigabytes, and whose duration is in the order of tens of seconds or more.

WEBRC can be used with both multicast and unicast delivery. WEBRC requires connectivity between a sender and receivers, but does not require connectivity from receivers to a sender for multicast delivery, and requires a small bandwidth connection from receivers to a sender for unicast delivery.

WEBRC inherently works with all types of networks, including LANs, WANs, Intranets, the Internet, asymmetric networks, wireless networks, and satellite networks. Thus, the inherent raw scalability of WEBRC is unlimited. However, in some network environments varying reception rates to receivers may not be advantageous, e.g., when the network is a satellite network with a fixed amount of bandwidth dedicated to a session.

2. General Architecture

A WEBRC session comprises a logically related set of channels originating from a single sender that are used for some period of time to carry data packets with a header carrying WEBRC Congestion Control Information.

At the network layer, a channel can be uniquely identified by a (sender IP address, group address) pair. For multicast, the group address is a multicast group address. For unicast, the group address is an identifier assigned by the sender so that no two group addresses associated with the sender are the same.

For each WEBRC session, the channels within the session are mapped uniquely to consecutive channel numbers. In each packet sent to a channel the channel number that corresponds to the channel is carried in the WEBRC Congestion Control Information. A WEBRC receiver uses the channel number to determine which channel within a session a packet is received from. When packets are received, they are first checked to see that they belong to the appropriate session before WEBRC is applied. For example, if LCT [4] is being used with the session, then the sender IP address together with the Transport Session Identifier supported by LCT would be used to determine which session received packets belong to. The particular details of how this filtering is performed it outside the scope of this document. In the remainder of this document, channels will be referred to by their channel number in the scope of the session.

At the sender, time is partitioned into time slots, each of duration TSD seconds. There are a fixed number T of time slot indices associated with a session. As time progresses, the time slot index increases by one modulo T each TSD seconds.

WEBRC congestion control is achieved by having the sender send packets associated with a given session to several different channels. Individual receivers dynamically join and leave these channels according to the network congestion as experienced by the receiver. The channels associated with a session consist of one base channel and T wave channels.

The packet rate for all channels varies over time. For the base channel, packets are sent to the channel at a low rate BCR_P at the beginning of a time slot and this rate gradually decreases to P*BCR_P at the end of the time slot, where P < 1 is a constant defined later. This pattern for the base channel repeats over each time slot.

For each wave channel i, packets are sent to channel i starting at a high rate MWCR_P and decreasing over time by a fixed fraction P per time slot until a rate of BCR_P is reached at the end of time slot i. Then, for a period of time called the quiescent period, no packets are sent to wave channel i, and thereafter the whole cycle repeats itself, where the duration of the cycle is T*TSD seconds. Thus, the wave channels are going through the same cyclic pattern of packet rate transmission spaced out evenly by TSD seconds.

The congestion control adjustments are performed at each receiver independently of all other receivers, and in the case of multicast without any impact on the sender. When a receiver initiates a session it first joins and remains joined to the base channel for the duration of its participation in the session. The packets in the base channel help the receiver orient itself in terms of what the current time slot index is.

Before joining a session, the receivers must obtain enough of the session description to start the session. This must include the relevant session parameters needed by a receiver to participate in the session and perform WEBRC congestion control. The session description is determined by the sender, and is typically communicated to the receivers out of band.

Once a receiver joins a session the receiver adjusts its rate upwards and downwards by joining wave channels in sequence, from the lowest rate wave channel and moving towards the higher rate wave channels. Since the relative ordering among the channels with respect to their rate depends on the time slot index, it is important that the receiver continually monitor the current time slot index contained in received packets. The reception rate at the receiver is determined by how early each wave channel is joined by the receiver: the earlier the receiver joins a channel with respect to when its wave started, the higher the reception rate. When the receiver wants to decrease its rate, it joins the next wave channel at a later time relative to when it joined the previous wave. When the receiver wants to increase its rate, it joins the next wave channel at an earlier time relative to when it joined the previous wave.

Once the receiver is joined to a wave channel, the receiver remains joined to the wave channel until the channel goes quiescent, at which point the receiver MUST leave the channel.

The way the receiver adjusts its reception rate is inspired by TFRC [2]. The receiver at all points in time maintains a target reception rate, and the receiver is allowed to join the next wave channel if after joining its reception rate would be at most its target reception rate. The target rate is continually updated based on a set of measured parameters. The primary parameters are the average loss probability (measured in much the same way as described in TFRC) and the average round-trip time (measured as described below).

3. Packet Congestion Control Information

Packets sent to a session using WEBRC must include Congestion Control Information. The default fields and their formats for this information, shown in Figure 1, are recommended for use by protocol instantiations to ensure a uniform format across different protocol instantiations. Other formats may be used by protocol instantiations, but if the default format is not used by a protocol insantiation, then the protocol instantiation must specify the lengths and positions of fields within the Congestion Control Information described in the default format.

Other building blocks may describe some of the same fields as described for the Congestion Control Information. It is recommended that protocol instantiations using multiple building blocks include shared fields at most once in each packet.

The position of the Congestion Control Information within a packet must be specified by any protocol instantiation that uses WEBRC.

The default length of the Congestion Control Information for WEBRC is 32-bits. All integer fields are carried in "big-endian" or "network order" format, that is, most significant byte (octet) first. All constants, unless otherwise specified, are expressed in base ten.

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |Time slot index| Channel number|     Packet sequence number    |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1 - Default Congestion Control Information format for WEBRC

The function of each field in the Congestion Control Information is the following.

Time slot index (TSI): 8 bits

TSI indicates the index of the current time slot. This must be sent in each packet within the session. The time slot index increases by one modulo T each TSD seconds at the sender, where T is the number of time slots associated with the session and TSD is the time slot duration.

Channel number (CN): 8 bits

CN is the channel number that this packet belongs to. CN for the base channel is T, and the CNs for the wave channels are 0 through T-1.

Packet sequence number (PSN): 16 bits

The PSN of each packet is scoped by its CN value. The sequence numbers of consecutive packets sent to the base channel are numbered consecutively decreasing modulo 2^16. The same sequence of PSNs are used for each wave channel in each cycle. The sequence numbers of consecutive packets sent to a wave channel are numbered consecutively decreasing modulo 2^16 within each cycle, ending with the last packet sent to the channel before the channel goes quiescent with PSN = 0.

4. Sender Operation

The sender operation is much simpler by design than the receiver operation.

4.1. Sender inputs and initialization

The primary input to the sender for the session is MSR_b. MSR_b is the maximum sender transmission rate in bits per second at any point in time in aggregate to all channels. This is also the maximum rate in bits per second that any receiver could receive data from the session at any point in time.

The secondary inputs to the sender are listed below. These are secondary inputs because in general the values for these inputs will be fixed to default values that won't change, or because they are set based on non-WEBRC considerations.

- o LENP_B is the length of packets in bytes sent to the session. The value of LENP_B depends on the complete protocol, but in general this should be set to as high a value as possible without exceeding the MTU size for the network that would cause fragmentation.
- o BCR_b is the transmission rate in bits per second to the base channel. The default value for BCR_b is 8 Kbps.
- o TSD is the time slot duration measured in seconds. The default value for TSD is 10 seconds.
- o QD is the quiescent period duration measured in seconds. The default value for QD is 30 seconds.
- o P is the drop in the rate over the duration of a time slot per channel. The default value for P is 0.75.

From these inputs the following fixed sender parameters can be derived as follows.

- o MSR_P = MSR_b/(8*LENP_B) is the maximum sender transmission rate in packets per second.
- o BCR_P = BCR_b/(8*LENP_B) is the rate of the base channel in packets per second at the beginning of a time slot.
- o N = ceil{log_{1/P}(((1-P)/P)*(MSR_P/BCR_P)+1)}-1 is the number of active time slots for a wave channel. A wave channel is active in any time slot if it is not quiescent. N is also the number of wave channels active in every time slot.

- o Q = ceil(QD/TSD) is the number of quiescent time slots for a wave channel.
- o T = N + Q is the total number of time slots in a cycle. T is also the total number of wave channels.
- o For the base channel CN = T and for the wave channels CN = 0,...,T-1. The sender has the description of the channels assigned to the session and the mapping between the channels and the CNs.
- o C = TSD*T is the total duration in seconds of a cycle.

4.2. Sending packets to the session

The sender keeps track of the current time slot index TSI. The value of TSI is incremented by 1 modulo T each TSD seconds. The value of TSI is placed into each packet in the format described in Section 3. For each packet sent to the session, the sender places the channel number CN of the channel into the packets in the format described in Section 3. Recall that CN = T for the base channel and CN = 0,...,T-1 for the wave channels.

For each packet sent to the session, the sender calculates and places PSN into the packet. The value of PSN is scoped by CN, and the value of PSN is consecutively decreasing within each CN as time moves forward. Furthermore, for each wave channel, the last packet sent to a wave channel before it becomes quiescent must have PSN =0. This implies that the PSN values are 0, 1, 2, 3, ..., etc., for a wave channel going backward in time starting at the last packet sent to the channel just before the channel becomes quiescent. The format for the PSN within packets is described in Section 3.

Packets are sent to the base channel at a rate of BCR_P packets per second at the beginning of a time slot decreasing at a constant relative rate till the rate reaches P*BCR_P at the end of a time slot. To explain the packet rate for wave channels, suppose for simplicity that MSR_P/BCR_P = 1 + (1/P) + (1/P)^2 + ... + (1/P)^N = ((1/P)^{N+1}-1)*P/(1-P). For all i = 0,...,T-1, the rate for wave channel i behaves as follows. At the beginning of time slot i+Q+1 modulo T, packets are sent to wave channel i at the rate of BCR_P*(1/P)^N. The rate of packets sent to wave channel i steadily decreases geometrically by a factor of P during each of the next N time slots until the end of time slot i when the rate reaches rate BCR_P. During the Q time slots i+1 modulo T, i+2 modulo T,...,i+Q modulo T, wave channel i is quiescent, i.e., no packets are sent to the channel.

5. Receiver Operation

The bulk of the complexity in WEBRC is in the receiver operation. The sender transmission rate to the channels is designed so that the receiver reception rate behaves as follows. As each wave channel becomes quiescent the receiver leaves the channel. Suppose for ease of explanation that during the reception there is no packet loss and packets are arriving at exactly the rate they were sent.

When the receiver wants to increase its rate, it first joins the base channel and then consecutively joins wave channels from the lowest rate channel to the highest rate channel. When the receiver wants to maintain its current reception rate and it is already joined to NWC wave channels, if the receiver joins channel i+NWC-2 modulo T sometime during time slot i-1 then the receiver joins channel i+NWC-1 modulo T TSD seconds later in time slot i.

Suppose the receiver wants to decrease its rate till it is joined to just the base channel. Assume that a receiver is joined to the NWC wave channels i, i+1 modulo T,...i+NWC-1 modulo T at the beginning of time slot i. Then, the aggregate packet reception rate of the receiver over the next NWC time slots will behave as follows. At the beginning of time slot i the receiver reception rate is BCR_P*(1 + (1/P) + (1/P)^2 + ... + (1/P)^NWC). Then the receiver reception rate decreases by a factor of P over the duration of each time slot, and between the end of each time slot and the beginning of the next the reception rate decreases by an additive amount of P*BCR_P. At the end of time slot i+NWC-1 mod T, the receiver reception rate is BCR_P*(1+P), and at the beginning of time slot i+NWC mod T the receiver is joined only to the base channel and its reception rate is BCR_P.

5.1. Receiver inputs and initialization

Before joining a session the receiver should have the values of LENP_B, BCR_P, TSD, P, N, Q and T. Some of these values may be obtained or measured once the receiver has joined the session. For example, the receiver can obtain LENP_B and T from the first packet received from the base channel, and the receiver can measure BCR_P once it is joined to the base channel. The values of P, Q and TSD may be fixed to default values built into the receiver that don't change from session to session, and the value of N can be computed as T-Q. The receiver must know the mapping between the CNs and the channels before it joins the session.

When a receiver first joins a session, it must first join just the base channel and start receiving packets to determine the current time slot index. If during the course of the session the receiver continually loses a high fraction of the packets from the base channel even when the receiver is only joined to the base channel, the receiver SHOULD leave the session.

The receiver may also have other individually set parameters that may be used to determine its behavior. Two such parameters are:

- o MRR_b is the maximum receiver reception rate in bits per second. This may be used to determine the maximum reception rate this receiver is willing to reach. Thus, the maximum reception rate that the client can possibly achieve in the session is the minimum of MSR_b and MRR_b. A recommended value of MRR_b for a receiver is the bandwidth capacity of the last link to the receiver. MRR_P is the maximum receiver reception rate in packets per second, i.e., MRR_P = MRR_b/(8*LENP_B).
- o CONNEC is the number of virtual connections that the receiver is making to the sender to receive data from the session. This affects how competitive the receiver will be compared to other receivers (both TCP and WEBRC receivers) in competing for bandwidth over bottleneck links. The default standard value for CONNEC is 1.

5.2. Receiver measurements and calculations

As outlined in the introduction, the way a receiver adjusts its reception rate is inspired by TFRC [2]. The receiver at all points in time maintains a target reception rate, and the receiver is allowed to join the next wave channel if joining would increase its reception rate to at most its target reception rate. The target rate is continually updated based on a set of measured parameters. The primary parameters are the average loss probability LOSSP and the average round-trip time AMRTT.

5.2.1. Average loss probability

The average loss probability LOSSP of the receiver is maintained in a manner very similar to that described in TFRC [2] with the following difference:

- o In TFRC, it is recommended that the loss probability be averaged over the previous 8 loss events.
- o In WEBRC, it is recommended that the loss probability LOSSP be averaged over the previous channel periods that include 8 channel periods where there was at least one loss event. A channel period is defined as the time between when a wave channel is joined till the time the next wave channel is joined. Thus, since a channel period is generally around TSD seconds, this generally means that the loss probability will be averaged over at least 8*TSD seconds. The calculation of LOSSP is exactly as described in TFRC, with the same definition of loss events.

5.2.2. Average round-trip time

The receiver maintains an average round-trip time, ARTT, as the exponentially weighted average of an RTT value. Each time the receiver joins a channel (either the base channel at the beginning or wave channels continually), it makes a measurement of the round-trip time RTT as follows. When the receiver sends the join for the channel it records the current time X and sets a boolean variable JOINING to true. When the first packet is received from the channel the receiver records the current time Y and resets the value of JOINING to false. When the receiver receives a second packet from the channel it records the current time Z. Then, the value of RTT is set to 3*Y/2 - Z/2 - X. (Note that this value can be negative.)

The value of ARTT is updated to max{ARRT/2, (1-alpha)*ARTT + alpha*RTT}. The recommended value for alpha is 0.1.

5.2.3. Rate Equation

The receiver calculates the expected reception rate REQN based on the TCP-equation as follows: REQN = CONNEC/(ARTT*sqrt{LOSSP}(0.816 + 7.35*LOSSP*(1+32*LOSSP^2))). This equation comes from TFRC [2].

5.2.4. Epochs

The receiver makes decisions on whether or not to join another wave channel at equally spaced units of time called epochs. The duration of an epoch, EL, is set to be a small fraction of TSD, so that decisions to join a channel can be made at a much finer granularity than TSD. A standard setting is EL = TSD/20. Thus, if TSD = 10 seconds, then EL = 0.5 seconds.

5.2.5. Average reception rate

There are two averaged versions of reception rate maintained by the receiver, TRR_P, the true reception rate, and ARR_P, the anticipated reception rate, that are used for different purposes. Thus, these two averaged versions are calculated quite differently.

The true reception rate TRR_P is used to ensure that the receiver does not increase its reception rate too quickly above its current reception rate. TRR_P is calculated based on the measurement of RR_P, where RR_P is the receiver reception rate in packets per second measured at the beginning of an epoch averaged over the epoch that just ended. TRR_P is initialized to BCR_P. TRR_P is updated to (1-beta)*TRR_P + beta*RR_P at the beginning of each epoch after RR_P is measured for the previous epoch. The recommended value for beta is 2.0*EL/TSD.

The anticipated reception rate ARR_P is used to compare against the target rate to decide whether or not the receiver can increase its reception rate by joining the next channel. ARR_P is calculated based on two other measurements, IRR_P and NWC. IRR_P is what the ideal receiver reception rate should have been in packets per second measured at the beginning of the epoch averaged over the previous epoch , i.e., IRR_P includes both received and lost packets. NWC is the number of wave channels that the receiver is joined to at any point in time. ARR_P is initialized to BCR_P and NWC is initialized to 0. ARR_P, IRR_P and NWC are updated as follows.

- o At the beginning of each epoch, IRR_P is measured over the previous epoch and then ARR_P is updated to pow{P, EL/TSD}*(1-gamma)*ARR_P + gamma*IRR_P.
- o At the beginning of an epoch when a join is made to a wave channel, NWC is updated to NWC+1 and then ARR_P is further updated to ARR_P*((1/P)^{NWC+1}-1)/((1/P)^NWC-1).
- o When the next time slot index is detected and a leave is sent for the wave channel that just went quiescent, NWC is updated to NWC-1, and ARR_P is updated to ARR_P - P*BCR_P.

The intuition for this is that ARR_P is compared against the target rate to determine if another channel should be joined. When a join is sent for a channel the value of ARR_P is raised to what the reception rate would be if the join occurred immediately. This ensures that ARR_P will immediately spike up to the anticipated reception rate after the join is processed, and this will inhibit a join to another channel at the beginning of the subsequent epoch. Lowering ARR_P by a relative amount pow{P,EL/TSD} at the beginning of each epoch and by an additive amount P*BCR_P at the beginning of each time slot ensures that the averaged ARR_P value is tracking the decreasing reception rate faithfully, and is needed to ensure that the average value of ARR_P is the average ideal reception rate, i.e., the average reception rate counting both received and lost packets.

The value of gamma is different when the value of ARR_P is less than SSR_P (see below, the slow start threshold in packets per second). When ARR_P is less than SSR_P the value of gamma is set to P^{1/2}(1-P)/(1-P^{3/2}). When ARR_P is greater than SSR_P the recommended value of gamma is 2.0*EL/TSD.

5.2.6. Slow start

WEBRC uses a slow start mechanism to quickly ramp up its rate at both the beginning of the session and in the middle of a session when the rate drops precipitously. To enact this, the receiver maintains the following parameters:

- o SSMINR_P is the minimum allowed slow start threshold rate in packets per second. SSMINR_P is set to BCR_P/P^2.
- o SSR_P is the slow start threshold rate in packets per second. SSR_P is initialized to MRR_P*P^2.

5.2.7. Target rate

The target rate TRATE is computed as TRATE = min{max{SSR_P, REQN}, TRR_P/P^2, MRR_P}.

5.3. Receiver events

There are various receiver events, some of which are triggered by the passing of time on the receiver, and others by events such as packet loss, reception of a first packet from a channel, and exceptional time-outs.

5.3.1. Epoch change

This is an event that is driven by the passage of time on the receiver, which occurs each EL seconds. When this happens, TRR_P and ARR_P are computed as described in Section 5.2.5. Immediately after these updates, a decision is made about whether to join a wave channel as described in Section 5.3.3.

5.3.2. Time slot change

This is an event that is driven by the reception of a packet with a new TSI value. When a packet with a new TSI = i is received, a leave is sent for wave channel i, NWC is updated to NWC-1 and ARR_P is updated to ARR_P - P*BCR_P as described in Section 5.2.5.

5.3.3. Join a wave channel

At the beginning of each epoch, after updating the values of ARR_P and TRR_P as described in Section 5.2.5, the receiver decides whether or not to join a wave channel as follows:

- o If there is a loss event in progress (LOSS_EVENT = true) or there is a join of a channel in progress (JOINING = true), then no join of a channel is attempted.
- o The receiver calculates REQN as described in Section 5.2.3.
- o The receiver calculates TRATE as described in Section 5.2.7.
- o The receiver sees if TRATE > ARR_P*((1/P)^{NWC+2}-1)/((1/P)^{NWC+1}-1). Suppose TSI = i and NWC = J. If the inequality is true then the receiver joins the next wave channel with CN = i+J modulo T, updates NWC to NWC+1 and then ARTT is updated to ARR_P*((1/P)^{NWC+1}-1)/((1/P)^NWC-1) as described in Section 5.2.5.

5.3.4. Loss event

Each time the receiver detects a lost packet (based on the sequence numbers in the packets scoped by the channel number), the receiver records the start of a new loss event, and sets a boolean variable LOSS_EVENT to true that will automatically reset to false after ARTT seconds. All subsequent packet loss for a period of ARTT seconds is ignored. When a start of a loss event is detected, the value of SSR_P is updated to max{SSMINR_P, TRR_P*P^2}.

5.3.5. Exceptional timeouts

These are timeouts when the packet reception behavior is far from what it should be and should trigger a drastic event by the client. Exception timeouts include events such as when no packets are received for a long time, there is no change in the time slot index for a long time, or no first or second packet is received after join to channel for a long time.

6. Security Considerations

WEBRC can be subject to denial-of-service attacks by attackers which try to confuse the congestion control mechanism, or send forged packets to the session which would prevent successful reconstruction of large portions of the objects.

The same exact problems are present in TCP, where an attacker can forge packets and either slow down or increase the throughput of the session, or replace parts of the data stream with forged data. If the stream is carrying compressed or otherwise coded data, even a single forged packet could also cause incorrect reconstruction of the rest of the data stream.

It is therefore RECOMMENDED that protocol instantiations that use WEBRC implement some form of packet authentication to protect themselves against such attacks, such as that described in [7].

7. IANA Considerations

No information in this specification is subject to IANA registration.

Building blocks used in conjunstion with WEBRC may introduce IANA considerations.

8. Intellectual Property Issues

Digital Fountain maintains all rights to the WEBRC technology, but will grant a royalty free worldwide license to the WEBRC technology once it has been standardized.

WEBRC may be used with other protocols which are proprietary, or have pending or granted patents.

9. Acknowledgments

Thanks to X for their comments on this draft.

10. References

[1] Byers, J.W., Frumin, M., Horn, G., Luby, M., Mitzenmacher, M., Roetter, A., and Shaver, W., "FLID-DL: Congestion Control for Layered Multicast", Proceedings of Second International Workshop on Networked Group Communications (NGC 2000), Palo Alto, CA, November 2000.

[2] Handley, M., Padhye, J., Floyd, S., Widmer, J., "TCP Friendly Rate Control (TFRC): Protocol Specification", Internet Draft draft-ietf-tsvwg-tfrc-03, July 2001, a work in progress.

[3] Luby, M., Gemmell, J., Vicisano, L., Rizzo, L., Crowcroft, J., "Asynchronous Layered Coding: A massively scalable reliable content delivery protocol instantiation", Internet Draft draft-ietf-rmt-pi-alc-03, October 2001, a work in progress.

[4] Luby, M., Gemmell, J., Vicisano, L., Rizzo, L., Handley, M., Crowcroft, J., "Layered Coding Transport: A massively scalable content delivery transport building block", Internet Draft draft-ietf-rmt-bb-lct-02.txt, October 2001, a work in progress.

[5] Luby, M., Gemmell, Vicisano, L., J., Rizzo, L., Handley, M., Crowcroft, J., "The use of Forward Error Correction in Reliable Multicast", Internet Draft draft-ietf-rmt-info-fec-00.txt, November 2000, a work in progress.

[6] Luby, M., Gemmell, J., Vicisano, L., Rizzo, L., Handley, M., Crowcroft, J., "RMT BB Forward Error Correction Codes", Internet Draft draft-ietf-rmt-bb-fec-03.txt, July 2001.

[7] Perrig, A., Canetti, R., Song, D., Tygar, J.D., "Efficient and Secure Source Authentication for Multicast", Network and Distributed System Security Symposium, NDSS 2001, pp. 35-46, February 2001.

[8] Vicisano, L., Rizzo, L., Crowcroft, J., "TCP-like Congestion Control for Layered Multicast Data Transfer", IEEE Infocom '98, San Francisco, CA, Mar.28-Apr.1 1998.

11. Authors' Addresses

Michael Luby
luby@digitalfountain.com
Digital Fountain
39141 Civic Center Drive
Suite 300
Fremont, CA, USA, 94538

Vivek Goyal
vivek@digitalfountain.com
Digital Fountain
39141 Civic Center Drive
Suite 300
Fremont, CA, USA, 94538

Simon Skaria
simonskaria@yahoo.com
Digital Fountain and UC Irvine

12. Full Copyright Statement