              Supplemental Information for the New Definition
          of the EF PHB (Expedited Forwarding Per-Hop Behavior)

Status of this Memo

Copyright Notice

Abstract

   This document was written during the process of clarification of
   RFC2598 "An Expedited Forwarding PHB" that led to the publication of
   revised specification of EF "An Expedited Forwarding PHB". Its
   primary motivation is providing additional explanation to the revised
   EF definition and its properties. The document also provides
   additional implementation examples and gives some guidance for
   computation of the numerical parameters of the new definition for
   several well known schedulers and router architectures.

Table of Contents

1. Introduction

   The Expedited Forwarding (EF) Per-Hop Behavior (PHB) was designed to
   be used to build a low-loss, low-latency, low-jitter, assured
   bandwidth service. The potential benefits of this service, and
   therefore the EF PHB, are enormous. Because of the great value of
   this PHB, it is critical that the forwarding behavior required of and
   delivered by an EF-compliant node be specific, quantifiable, and
   unambiguous.

   Unfortunately, the definition of EF PHB in the original RFC2598 [10]
   was not sufficiently precise (see Appendix A and [4]). A more
   precise definition is given in [6]. This document is intended to aid
   in the understanding of the properties of the new definition and
   provide supplemental information not included in the text of [6] for
   sake of brevity.

   This document is outlined as follows. In section 2, we briefly
   restate the definition for EF PHB of [6]. We then provide some
   additional discussion of this definition and describe some of its
   properties. We discuss the issues associated with per-packet delay

and loss in sections 3 and 4. In section 5 we discuss the impact of
known scheduling architectures on the critical parameters of the new
definition. We also discuss the impact of deviation of real devices
from the ideal output-buffered model on the magnitude of the critical
parameters in the definition.

2. Definition of EF PHB

2.1. The formal definition

An intuitive explanation of the new EF definition is described in
[6]. Here we restate the formal definition from [6] verbatim.

A node that supports EF on an interface I at some configured rate R
MUST satisfy the following equations:

$$d_j <= f_j + E_a \text{ for all } j>0 \qquad\qquad (eq\_1)$$

where f_j is defined iteratively by

$$f_0 = 0, d_0 = 0$$

$$f_j = \max(a_j, \min(d_{j-1}, f_{j-1})) + l_j/R, \text{ for all } j > 0 \quad (eq\_2)$$

In this definition:

- d_j is the time that the last bit of the j-th EF packet to
  depart actually leaves the node from the interface I.

- f_j is the target departure time for the j-th EF packet to
  depart from I, the "ideal" time at or before which the last bit
  of that packet should leave the node.

- a_j is the time that the last bit of the j-th EF packet
  destined to the output I actually arrives at the node.

- l_j is the size (bits) of the j-th EF packet to depart from I.
  l_j is measured on the IP datagram (IP header plus payload) and
  does not include any lower layer (e.g. MAC layer) overhead.

- R is the EF configured rate at output I (in bits/second).

- E_a is the error term for the treatment of the EF aggregate.
  Note that E_a represents the worst case deviation between
  actual departure time of an EF packet and ideal departure time
  of the same packet, i.e. E_a provides an upper bound on $(d_j -
  f_j)$ for all j.

- d_0 and f_0 do not refer to a real packet departure but are
  used purely for the purposes of the recursion. The time origin
  should be chosen such that no EF packets are in the system at
  time 0.

- for the definitions of a_j and d_j, the "last bit" of the
  packet includes the layer 2 trailer if present, because a
  packet cannot generally be considered available for forwarding
  until such a trailer has been received.

An EF-compliant node MUST be able to be characterized by the range of
possible R values that it can support on each of its interfaces while
conforming to these equations, and the value of E_a that can be met
on each interface. R may be line rate or less. E_a MAY be specified
as a worst-case value for all possible R values or MAY be expressed
as a function of R.

Note also that, since a node may have multiple inputs and complex
internal scheduling, the j-th EF packet to arrive at the node
destined for a certain interface may not be the j-th EF packet to
depart from that interface. It is in this sense that eq_1 and eq_2
are unaware of packet identity.

In addition, a node that supports EF on an interface I at some
configured rate R MUST satisfy the following equations:

$$D_j <= F_j + E_p \text{ for all } j>0 \qquad\qquad (eq\_3)$$

where F_j is defined iteratively by

$$F_0 = 0, D_0 = 0$$

$$F_j = max(A_j, min(D_{j-1}, F_{j-1})) + L_j/R, \text{ for all } j > 0 \quad (eq\_4)$$

In this definition:

- D_j is the actual departure time of the individual EF packet
  that arrived at the node destined for interface I at time A_j,
  i.e., given a packet which was the j-th EF packet destined for
  I to arrive at the node via any input, D_j is the time at which
  the last bit of that individual packet actually leaves the node
  from the interface I.

- F_j is the target departure time for the individual EF packet
  that arrived at the node destined for interface I at time A_j.

- A_j is the time that the last bit of the j-th EF packet
  destined to the output I to arrive actually arrives at the
  node.

- L_j is the size (bits) of the j-th EF packet to arrive at the
  node that is destined to output I. L_j is measured on the IP
  datagram (IP header plus payload) and does not include any
  lower layer (e.g. MAC layer) overhead.

- R is the EF configured rate at output I (in bits/second).

- E_p is the error term for the treatment of individual EF
  packets. Note that E_p represents the worst case deviation
  between the actual departure time of an EF packet and the ideal
  departure time of the same packet, i.e. E_p provides an upper
  bound on (D_j - F_j) for all j.

- D_0 and F_0 do not refer to a real packet departure but are
  used purely for the purposes of the recursion. The time origin
  should be chosen such that no EF packets are in the system at
  time 0.

- for the definitions of A_j and D_j, the "last bit" of the
  packet includes the layer 2 trailer if present, because a
  packet cannot generally be considered available for forwarding
  until such a trailer has been received.

It is the fact that D_j and F_j refer to departure times for the j-th
packet to arrive that makes eq_3 and eq_4 aware of packet identity.
This is the critical distinction between the last two equations and
the first two.

An EF-compliant node SHOULD be able to be characterized by the range
of possible R values that it can support on each of its interfaces
while conforming to these equations, and the value of E_p that can be
met on each interface. E_p MAY be specified as a worst-case value
for all possible R values or MAY be expressed as a function of R. An
E_p value of "undefined" MAY be specified.

Finally, there is an additional recommendation in [6] that an EF
compliant node SHOULD NOT reorder packets within a micorflow.

The definitions described in this section are referred to as
aggregate and packet-identity-aware packet scale rate guarantee
[4],[2]. An alternative mathematical characterization of packet
scale rate guarantee is given in Appendix B.

2.2. Relation to Packet Scale Rate Guarantee

   Consider the case of an ideal output-buffered device with an EF FIFO
   at the output. For such a device, the i-th packet to arrive to the
   device is also the i-th packet to depart from the device. Therefore,
   in this ideal model the aggregate behavior and packet-identity-aware
   characteristics are identical, and E_a = E_p. In this section we
   therefore omit the subscript and refer to the latency term simply as
   E.

   It could be shown that for such an ideal device the definition of
   section 2.1 is stronger than the well-known rate-latency curve [2] in
   the sense that if a scheduler satisfies the EF definition it also
   satisfies the rate-latency curve. As a result, all the properties
   known for the rate-latency curve also apply to the modified EF
   definition. However, we argue below that the definition of section
   2.1 is more suitable to reflect the intent of EF PHB than the rate-
   latency curve.

   It is shown in [2] that the rate-latency curve is equivalent to the
   following definition:

   Definition of Rate Latency Curve (RLC):

      $D(j) <= F'(j) + E$ (eq_5)

   where

      $F'(0)=0, F'(j)=\max(a(j), F'(j-1))+ L(j)/R$ for all $j>0$ (eq_6)

   It can be easily verified that the EF definition of section 2.1 is
   stronger than RLC by noticing that for all j, $F'(j) >= F(j)$.

   It is easy to see that $F'(j)$ in the definition of RLC corresponds to
   the time the j-th departure should have occurred should the EF
   aggregate be constantly served exactly at its configured rate R.
   Following the common convention, we refer to $F'(j)$ as the "fluid
   finish time" of the j-th packet to depart.

   The intuitive meaning of the rate-latency curve of RLC is that any
   packet is served at most time E later than this packet would finish
   service in the fluid model.

   For RLC (and hence for the stronger EF definition) it holds that in
   any interval (0,t) the EF aggregate gets close to the desired service
   rate R (as long as there is enough traffic to sustain this rate).
   The discrepancy between the ideal and the actual service in this
   interval depends on the latency term E, which in turn depends on the

scheduling implementation. The smaller E, the smaller the difference
between the configured rate and the actual rate achieved by the
scheduler.

While RLC guarantees the desired rate to the EF aggregate in all
intervals (0,t) to within a specified error, it may nevertheless
result in large gaps in service. For example, suppose that (a large
number) N of identical EF packets of length L arrived from different
interfaces to the EF queue in the absence of any non-EF traffic.
Then any work-conserving scheduler will serve all N packets at link
speed. When the last packet is sent at time NL/C, where C is the
capacity of output link, F'(N) will be equal to NL/R. That is, the
scheduler is running ahead of ideal, since NL/C < NL/R for R < C.
Suppose now that at time NL/C a large number of non-EF packets
arrive, followed by a single EF packet. Then the scheduler can
legitimately delay starting to send the EF packet until time
F'(N+1)=(N+1)L/R + E - L/C. This means that the EF aggregate will
have no service at all in the interval (NL/C, (N+1)L/R + E - L/C).
This interval can be quite large if R is substantially smaller than
C. In essence, the EF aggregate can be "punished" by a gap in
service for receiving faster service than its configured rate at the
beginning.

The new EF definition alleviates this problem by introducing the term
min(D(j-1), F(j-1)) in the recursion. Essentially, this means that
the fluid finishing time is "reset" if that packet is sent before its
"ideal" departure time. As a consequence of that, for the case where
the EF aggregate is served in the FIFO order, suppose a packet
arrives at time t to a server satisfying the EF definition. The
packet will be transmitted no later than time t + Q(t)/R + E, where
Q(t) is the EF queue size at time t (including the packet under
discussion)[4].

2.3. The need for dual characterization of EF PHB

In a more general case, where either the output scheduler does not
serve the EF packets in a FIFO order, or the variable internal delay
in the device reorders packets while delivering them to the output
(or both), the i-th packet destined to a given output interface to
arrive to the device may no longer be the i-th packet to depart from
that interface. In that case the packet-identity-aware and the
aggregate definitions are no longer identical.

The aggregate behavior definition can be viewed as a truly aggregate
characteristic of the service provided to EF packets. For an
analogy, consider a dark reservoir to which all arriving packets are
placed. A scheduler is allowed to pick a packet from the reservoir
in a random order, without any knowledge of the order of packet

arrivals. The aggregate part of the definition measures the accuracy
of the output rate provided to the EF aggregate as a whole. The
smaller E_a, the more accurate is the assurance that the reservoir is
drained at least at the configured rate.

Note that in this reservoir analogy packets of EF aggregate may be
arbitrarily reordered. However, the definition of EF PHB given in
[6] explicitly requires that no packet reordering occur within a
microflow. This requirement restricts the scheduling
implementations, or, in the reservoir analogy, the order of pulling
packets out of the reservoir to make sure that packets within a
microflow are not reordered, but it still allows reordering at the
aggregate level.

Note that reordering within the aggregate, as long as there is no
flow-level reordering, does not necessarily reflect a "bad" service.
Consider for example a scheduler that arbitrates among 10 different
EF "flows" with diverse rates. A scheduler that is aware of the rate
requirements may choose to send a packet of the faster flow before a
packet of the slower flow to maintain lower jitter at the flow level.
In particular, an ideal "flow"-aware WFQ scheduler will cause
reordering within the aggregate, while maintaining packet ordering
and small jitter at the flow level.

It is intuitively clear that for such a scheduler, as well as for a
simpler FIFO scheduler, the "accuracy" of the service rate is crucial
for minimizing "flow"-level jitter. The packet-identity-aware
definition quantifies this accuracy of the service rate.

However, the small value of E_a does not give any assurances about
the absolute value of per-packet delay. In fact, if the input rate
exceeds the configured rate, the aggregate behavior definition may
result in arbitrarily large delay of a subset of packets. This is
the primary motivation for the packet-identity-aware definition.

The primary goal of the packet-aware characterization of the EF
implementation is that, unlike the aggregate behavior
characterization, it provides a way to find a per-packet delay bound
as a function of input traffic parameters.

While the aggregate behavior definition characterizes the accuracy of
the service rate of the entire EF aggregate, the packet-identity-
aware part of the definition characterizes the deviation of the
device from an ideal server that serves the EF aggregate in FIFO
order at least at the configured rate.

The value of E_p in the packet-identity-aware definition is therefore
affected by two factors: the accuracy of the aggregate rate service

and the degree of packet reordering within the EF aggregate (under
the constraint that packets within the same microflow are not
reordered). Therefore, a sub-aggregate aware device that provides an
ideal service rate to the aggregate, and also provides an ideal rate
service for each of the sub-aggregates, may nevertheless have a very
large value of E_p (in this case E_p must be at least equal to the
ratio of the maximum packet size divided by the smallest rate of any
sub aggregate). As a result, a large value of E_p does not
necessarily mean that the service provided to EF aggregate is bad -
rather it may be an indication that the service is good, but non-
FIFO. On the other hand, a large value of E_p may also mean that the
aggregate service is very inaccurate (bursty), and hence in this case
the large value of E_p reflects a poor quality of implementation.

As a result, a large number of E_p does not necessarily provide any
guidance on the quality of the EF implementation. However, a small
value of E_p does indicate a high quality FIFO implementation.

Since E_p and E_a relate to different aspects of the EF
implementation, they should be considered together to determine the
quality of the implementation.

3. Per Packet delay

   The primary motivation for the packet-identity-aware definition is
   that it allows quantification of the per-packet delay bound. This
   section discusses the issues with computing per-packet delay.

3.1. Single hop delay bound

   If the total traffic arriving to an output port I from all inputs is
   constrained by a leaky bucket with parameters (R, B), where R is the
   configured rate at I, and B is the bucket depth (burst), then the
   delay of any packet departing from I is bounded by D_p, given by

      $$D_p = B/R + E_p \qquad\qquad\qquad\qquad\qquad\qquad (eq\_7)$$

   (see appendix B).

   Because the delay bound depends on the configured rate R and the
   input burstiness B, it is desirable for both of these parameters to
   be visible to a user of the device. A PDB desiring a particular
   delay bound may need to limit the range of configured rates and
   allowed burstiness that it can support in order to deliver such
   bound. Equation (eq_7) provides a means for determining an
   acceptable operating region for the device with a given E_p. It may
   also be useful to limit the total offered load to a given output to
   some rate $R_1 < R$ (e.g. to obtain end-to-end delay bounds [5]). It

   is important to realize that, while R_1 may also be a configurable
   parameter of the device, the delay bound in (eq_7) does not depend on
   it. It may be possible to get better bounds explicitly using the
   bound on the input rate, but the bound (eq_7) does not take advantage
   of this information.

3.2. Multi-hop worst case delay

   Although the PHB defines inherently local behavior, in this section
   we briefly discuss the issue of per-packet delay as the packet
   traverses several hops implementing EF PHB. Given a delay bound
   (eq_7) at a single hop, it is tempting to conclude that per-packet
   bound across h hops is simply h times the bound (eq_7). However,
   this is not necessarily the case, unless B represents the worst case
   input burstiness across all nodes in the network.

   Unfortunately, obtaining such a worst case value of B is not trivial.
   If EF PHB is implemented using aggregate class-based scheduling where
   all EF packets share a single FIFO, the effect of jitter accumulation
   may result in an increase in burstiness from hop to hop. In
   particular, it can be shown that unless severe restrictions on EF
   utilization are imposed, even if all EF flows are ideally shaped at
   the ingress, then for any value of delay D it is possible to
   construct a network where EF utilization on any link is bounded not
   to exceed a given factor, no flow traverses more than a specified
   number of hops, but there exists a packet that experiences a delay
   more than D [5]. This result implies that the ability to limit the
   worst case burstiness and the resulting end-to-end delay across
   several hops may require not only limiting EF utilization on all
   links, but also constraining the global network topology. Such
   topology constraints would need to be specified in the definition of
   any PDB built on top of EF PHB, if such PDB requires a strict worst
   case delay bound.

4. Packet loss

   Any device with finite buffering may need to drop packets if the
   input burstiness becomes sufficiently high. To meet the low loss
   objective of EF, a node may be characterized by the operating region
   in which loss of EF due to congestion will not occur. This may be
   specified as a token bucket of rate r <= R and burst size B that can
   be offered from all inputs to a given output interface without loss.

   However, as discussed in the previous section, the phenomenon of
   jitter accumulation makes it generally difficult to guarantee that
   the input burstiness never exceeds the specified operating region for
   nodes internal to the DiffServ domain. A no-loss guarantee across
   multiple hops may require specification of constraints on network

topology which are outside the scope of inherently local definition
of a PHB. Thus, it must be possible to establish whether a device
conforms to the EF definition even when some packets are lost.

This can be done by performing an "off-line" test of conformance to
equations (eq_1)- (eq_4). After observing a sequence of packets
entering and leaving the node, the packets which did not leave are
assumed lost and are notionally removed from the input stream. The
remaining packets now constitute the arrival stream and the packets
which left the node constitute the departure stream. Conformance to
the equations can thus be verified by considering only those packets
that successfully passed through the node.

Note that specification of which packets are lost in the case when
loss does occur is beyond the scope of the definition of EF PHB.
However, those packets that were not lost must conform to the
equations definition of EF PHB in section 2.1.

5. Implementation considerations

A packet passing through a router will experience delay for a number
of reasons. Two familiar components of this delay are the time the
packet spends in a buffer at an outgoing link waiting for the
scheduler to select it and the time it takes to actually transmit the
packet on the outgoing line.

There may be other components of a packet's delay through a router,
however. A router might have to do some amount of header processing
before the packet can be given to the correct output scheduler, for
example. In another case a router may have a FIFO buffer (called a
transmission queue in [7]) where the packet sits after being selected
by the output scheduler but before it is transmitted. In cases such
as these, the extra delay a packet may experience can be accounted
for by absorbing it into the latency terms E_a and E_p.

Implementing EF on a router with a multi-stage switch fabric requires
special attention. A packet may experience additional delays due to
the fact that it must compete with other traffic for forwarding
resources at multiple contention points in the switch core. The
delay an EF packet may experience before it even reaches the output-
link scheduler should be included in the latency term. Input-
buffered and input/output-buffered routers based on crossbar design
may also require modification of their latency terms. The factors
such as the speedup factor and the choice of crossbar arbitration
algorithms may affect the latency terms substantially.

Delay in the switch core comes from two sources, both of which must
be considered. The first part of this delay is the fixed delay a
packet experiences regardless of the other traffic. This component
of the delay includes the time it takes for things such as packet
segmentation and reassembly in cell based cores, enqueueing and
dequeuing at each stage, and transmission between stages. The second
part of the switch core delay is variable and depends on the type and
amount of other traffic traversing the core. This delay comes about
if the stages in the core mix traffic flowing between different
input/output port pairs. Thus, EF packets must compete against other
traffic for forwarding resources in the core. Some of this
competing traffic may even be traffic from other, non-EF aggregates.
This introduces extra delay, that can also be absorbed by the latency
term in the definition.

To capture these considerations, in this section we will consider two
simplified implementation examples. The first is an ideal output
buffered node where packets entering the device from an input
interface are immediately delivered to the output scheduler. In this
model the properties of the output scheduler fully define the values
of the parameters E_a and E_p. We will consider the case where the
output scheduler implements aggregate class-based queuing, so that
all EF packets share a single queue. We will discuss the values of
E_a and E_p for a variety of class-based schedulers widely
considered.

The second example will consider a router modeled as a black box with
a known bound on the variable delay a packet can experience from the
time it arrives to an input to the time it is delivered to its
destination output. The output scheduler in isolation is assumed to
be an aggregate scheduler where all EF packets share a single FIFO
queue, with a known value of $E_a(S)=E_p(S)=E(S)$. This model provides
a reasonable abstraction to a large class of router implementations.

5.1. The output buffered model with EF FIFO at the output.

As has been mentioned earlier, in this model $E_a = E_p$, so we shall
omit the subscript and refer to both terms as latency E. The
remainder of this subsection discusses E for a number of scheduling
implementations.

5.1.1. Strict Non-preemptive Priority Queue

A Strict Priority scheduler in which all EF packets share a single
FIFO queue which is served at strict non-preemptive priority over
other queues satisfies the EF definition with the latency term E =
MTU/C where MTU is the maximum packet size and C is the speed of the
output link.

5.1.2. WF2Q

   Another scheduler that satisfies the EF definition with a small
   latency term is WF2Q described in [1]. A class-based WF2Q scheduler,
   in which all EF traffic shares a single queue with the weight
   corresponding to the configured rate of the EF aggregate satisfies
   the EF definition with the latency term E = MTU/C+MTU/R.

5.1.3. Deficit Round Robin (DRR)

   For DRR [12], E can be shown to grow linearly with
   N*(r_max/r_min)*MTU, where r_min and r_max denote the smallest and
   the largest rate among the rate assignments of all queues in the
   scheduler, and N is the number of queues in the scheduler.

5.1.4. Start-Time Fair Queuing and Self-Clocked Fair Queuing

   For Start-Time Fair Queuing (SFQ) [9] and Self-Clocked Fair Queuing
   (SCFQ) [8] E can be shown to grow linearly with the number of queues
   in the scheduler.

5.2. Router with Internal Delay and EF FIFO at the output

   In this section we consider a router which is modeled as follows. A
   packet entering the router may experience a variable delay D_v with a
   known upper bound D. That is, $0 \le D_v \le D$. At the output all EF
   packets share a single class queue. Class queues are scheduled by a
   scheduler with a known value $E_p(S) = E(S)$ (where E(S) corresponds to
   the model where this scheduler is implemented in an ideal output
   buffered device).

   The computation of E_p is more complicated in this case. For such
   device, it can be shown that $E_p = E(S)+2D+2B/R$ (see [13]).

   Recall from the discussion of section 3 that bounding input
   burstiness B may not be easy in a general topology. In the absence
   of the knowledge of a bound on B one can bound E_p as $E_p = E(S) +
   D*C_{inp}/R$ (see [13]).

   Note also that the bounds in this section are derived using only the
   bound on the variable portion of the interval delay and the error
   bound of the output scheduler. If more details about the
   architecture of a device are available, it may be possible to compute
   better bounds.

6. Security Considerations

   This informational document provides additional information to aid in
   understanding of the EF PHB described in [6]. It adds no new
   functions to it. As a result, it adds no security issues to those
   described in that specification.

7. References

   [1]   J.C.R. Bennett and H. Zhang, "WF2Q: Worst-case Fair Weighted
          Fair Queuing", INFOCOM'96, March 1996.

   [2]   J.-Y. Le Boudec, P. Thiran, "Network Calculus", Springer Verlag
          Lecture Notes in Computer Science volume 2050, June 2001
          (available online at http://lcawww.epfl.ch).

   [3]   Bradner, S., "Key Words for Use in RFCs to Indicate Requirement
          Levels", BCP 14, RFC 2119, March 1997.

   [4]   J.C.R. Bennett, K. Benson, A. Charny, W. Courtney, J.Y. Le
          Boudec, "Delay Jitter Bounds and Packet Scale Rate Guarantee
          for Expedited Forwarding", Proc. Infocom 2001, April 2001.

   [5]   A. Charny, J.-Y. Le Boudec "Delay Bounds in a Network with
          Aggregate Scheduling". Proc. of QoFIS'2000, September 25-26,
          2000, Berlin, Germany.

   [6]   Davie, B., Charny, A., Baker, F., Bennett, J.C.R., Benson, K.,
          Boudec, J., Chiu, A., Courtney, W., Davari, S., Firoiu, V.,
          Kalmanek, C., Ramakrishnan, K.K. and D. Stiliadis, "An
          Expedited Forwarding PHB (Per-Hop Behavior)", RFC 3246, March
          2002.

   [7]   T. Ferrari and P. F. Chimento, "A Measurement-Based Analysis of
          Expedited Forwarding PHB Mechanisms," Eighth International
          Workshop on Quality of Service, Pittsburgh, PA, June 2000.

   [8]   S.J. Golestani. "A Self-clocked Fair Queuing Scheme for Broad-
          band Applications". In Proceedings of IEEE INFOCOM'94, pages
          636-646, Toronto, CA, April 1994.

   [9]   P. Goyal, H.M. Vin, and H. Chen. "Start-time Fair Queuing: A
          Scheduling Algorithm for Integrated Services". In Proceedings
          of the ACM-SIGCOMM 96, pages 157-168, Palo Alto, CA, August
          1996.

   [10]   Jacobson, V., Nichols, K. and K. Poduri, "An Expedited
          Forwarding PHB", RFC 2598, June 1999.

   [11] Jacobson, V., Nichols, K. and K. Poduri, "The 'Virtual Wire'
         Behavior Aggregate", Work in Progress.

   [12] M. Shreedhar and G. Varghese. "Efficient Fair Queuing Using
         Deficit Round Robin". In Proceedings of SIGCOMM'95, pages
         231-243, Boston, MA, September 1995.

   [13] Le Boudec, J.-Y., Charny, A. "Packet Scale Rate Guarantee for
         non-FIFO Nodes", Infocom 2002, New York, June 2002.

Appendix A. Difficulties with the RFC 2598 EF PHB Definition

   The definition of the EF PHB as given in [10] states:

   "The EF PHB is defined as a forwarding treatment for a particular
   diffserv aggregate where the departure rate of the aggregate's
   packets from any diffserv node must equal or exceed a configurable
   rate. The EF traffic SHOULD receive this rate independent of the
   intensity of any other traffic attempting to transit the node. It
   [the EF PHB departure rate] SHOULD average at least the configured
   rate when measured over any time interval equal to or longer than the
   time it takes to send an output link MTU sized packet at the
   configured rate."

   A literal interpretation of the definition would consider the
   behaviors given in the next two subsections as non-compliant. The
   definition also unnecessarily constrains the maximum configurable
   rate of an EF aggregate.

A.1 Perfectly-Clocked Forwarding

   Consider the following stream forwarded from a router with EF-
   configured rate $R=C/2$, where C is the output line rate. In the
   illustration, E is an MTU-sized EF packet while x is a non-EF packet
   or unused capacity, also of size MTU.

```
   E x E x E x E x E x E x...
    |-----|
```

   The interval between the vertical bars is $3*MTU/C$, which is greater
   than $MTU/(C/2)$, and so is subject to the EF PHB definition. During
   this interval, $3*MTU/2$ bits of the EF aggregate should be forwarded,
   but only MTU bits are forwarded. Therefore, while this forwarding
   pattern should be considered compliant under any reasonable
   interpretation of the EF PHB, it actually does not formally comply
   with the definition of RFC 2598.

   Note that this forwarding pattern can occur in any work-conserving
   scheduler in an ideal output-buffered architecture where EF packets
   arrive in a perfectly clocked manner according to the above pattern
   and are forwarded according to exactly the same pattern in the
   absence of any non-EF traffic.

   Trivial as this example may be, it reveals the lack of mathematical
   precision in the formal definition. The fact that no work-conserving
   scheduler can formally comply with the definition is unfortunate, and
   appears to warrant some changes to the definition that would correct
   this problem.

The underlying reason for the problem described here is quite simple
- one can only expect that the EF aggregate is served at configured
rate in some interval where there is enough backlog of EF packets to
sustain that rate. In the example above the packets come in exactly
at the rate at which they are served, and so there is no persistent
backlog. Certainly, if the input rate is even smaller than the
configured rate of the EF aggregate, there will be no backlog as
well, and a similar formal difficulty will occur.

A seemingly simple solution to this difficulty might be to require
that the EF aggregate is served at its configured rate only when the
queue is backlogged. However, as we show in the remainder of this
section, this solution does not suffice.

A.2 Router Internal Delay

We now argue that the example considered in the previous section is
not as trivial as it may seem at first glance.

Consider a router with EF configured rate $R = C/2$ as in the previous
example, but with an internal delay of $3T$ (where T = MTU/C) between
the time that a packet arrives at the router and the time that it is
first eligible for forwarding at the output link. Such things as
header processing, route look-up, and delay in switching through a
multi-layer fabric could cause this delay. Now suppose that EF
traffic arrives regularly at a rate of $(2/3)R = C/3$. The router will
perform as shown below.

    EF Packet Number 1 2 3 4 5 6 ...

    Arrival (at router) 0 3T 6T 9T 12T 15T ...

    Arrival (at scheduler) 3T 6T 9T 12T 15T 18T ...

    Departure 4T 7T 10T 13T 16T 19T ...

Again, the output does not satisfy the RFC 2598 definition of EF PHB.
As in the previous example, the underlying reason for this problem is
that the scheduler cannot forward EF traffic faster than it arrives.
However, it can be easily seen that the existence of internal delay
causes one packet to be inside the router at all times. An external
observer will rightfully conclude that the number of EF packets that
arrived to the router is always at least one greater than the number
of EF packets that left the router, and therefore the EF aggregate is
constantly backlogged. However, while the EF aggregate is
continuously backlogged, the observed output rate is nevertheless
strictly less that the configured rate.

This example indicates that the simple addition of the condition that
EF aggregate must receive its configured rate only when the EF
aggregate is backlogged does not suffice in this case.

Yet, the problem described here is of fundamental importance in
practice. Most routers have a certain amount of internal delay. A
vendor declaring EF compliance is not expected to simultaneously
declare the details of the internals of the router. Therefore, the
existence of internal delay may cause a perfectly reasonable EF
implementation to display seemingly non-conformant behavior, which is
clearly undesirable.

A.3 Maximum Configurable Rate and Provisioning Efficiency

It is well understood that with any non-preemptive scheduler, the
RFC-2598-compliant configurable rate for an EF aggregate cannot
exceed C/2 [11]. This is because an MTU-sized EF packet may arrive
to an empty queue at time t just as an MTU-sized non-EF packet begins
service. The maximum number of EF bits that could be forwarded
during the interval [t, t + 2*MTU/C] is MTU. But if configured rate
R > C/2, then this interval would be of length greater than MTU/R,
and more than MTU EF bits would have to be served during this
interval for the router to be compliant. Thus, R must be no greater
than C/2.

It can be shown that for schedulers other than PQ, such as various
implementations of WFQ, the maximum compliant configured rate may be
much smaller than 50%. For example, for SCFQ [8] the maximum
configured rate cannot exceed C/N, where N is the number of queues in
the scheduler. For WRR, mentioned as compliant in section 2.2 of RFC
2598, this limitation is even more severe. This is because in these
schedulers a packet arriving to an empty EF queue may be forced to
wait until one packet from each other queue (in the case of SCFQ) or
until several packets from each other queue (in the case of WRR) are
served before it will finally be forwarded.

While it is frequently assumed that the configured rate of EF traffic
will be substantially smaller than the link bandwidth, the
requirement that this rate should never exceed 50% of the link
bandwidth appears unnecessarily limiting. For example, in a fully
connected mesh network, where any flow traverses a single link on its
way from source to its destination there seems no compelling reason
to limit the amount of EF traffic to 50% (or an even smaller
percentage for some schedulers) of the link bandwidth.

Another, perhaps even more striking example is the fact that even a
TDM circuit with dedicated slots cannot be configured to forward EF
packets at more than 50% of the link speed without violating RFC 2598

(unless the entire link is configured for EF). If the configured
rate of EF traffic is greater than 50% (but less than the link
speed), there will always exist an interval longer than MTU/R in
which less than the configured rate is achieved. For example,
suppose the configured rate of the EF aggregate is 2C/3. Then the
forwarding pattern of the TDM circuit might be

```
E E x E E x E E x ...
   |---|
```

where only one packet is served in the marked interval of length 2T =
2MTU/C. But at least 4/3 MTU would have to be served during this
interval by a router in compliance with the definition in RFC 2598.
The fact that even a TDM line cannot be booked over 50% by EF traffic
indicates that the restriction is artificial and unnecessary.

A.4 The Non-trivial Nature of the Difficulties

One possibility to correct the problems discussed in the previous
sections might be to attempt to clarify the definition of the
intervals to which the definition applied or by averaging over
multiple intervals. However, an attempt to do so meets with
considerable analytical and implementation difficulties. For
example, attempting to align interval start times with some epochs of
the forwarded stream appears to require a certain degree of global
clock synchronization and is fraught with the risk of
misinterpretation and mistake in practice.

Another approach might be to allow averaging of the rates over some
larger time scale. However, it is unclear exactly what finite time
scale would suffice in all reasonable cases. Furthermore, this
approach would compromise the notion of very short-term time scale
guarantees that are the essence of EF PHB.

We also explored a combination of two simple fixes. The first is the
addition of the condition that the only intervals subject to the
definition are those that fall inside a period during which the EF
aggregate is continuously backlogged in the router (i.e., when an EF
packet is in the router). The second is the addition of an error
(latency) term that could serve as a figure-of-merit in the
advertising of EF services.

With the addition of these two changes the candidate definition
becomes as follows:

In any interval of time (t1, t2) in which EF traffic is continuously
backlogged, at least R(t2 - t1 - E) bits of EF traffic must be
served, where R is the configured rate for the EF aggregate and E is
an implementation-specific latency term.

The "continuously backlogged" condition eliminates the insufficient-
packets-to-forward difficulty while the addition of the latency term
of size MTU/C resolves the perfectly-clocked forwarding example
(section A.1), and also removes the limitation on EF configured rate.

However, neither fix (nor the two of them together) resolves the
example of section A.2. To see this, recall that in the example of
section A.2 the EF aggregate is continuously backlogged, but the
service rate of the EF aggregate is consistently smaller than the
configured rate, and therefore no finite latency term will suffice to
bring the example into conformance.

Appendix B. Alternative Characterization of Packet Scale Rate Guarantee

The proofs of several bounds in this document can be found in [13].
These proofs use an algebraic characterization of the aggregate
definition given by (eq_1), (eq_2), and packet identity aware
definition given by (eq_3), (eq_4). Since this characterization is
of interest on its own, we present it in this section.

Theorem B1. Characterization of the aggregate definition without
             f_n.

Consider a system where packets are numbered 1, 2, ... in order of
arrival. As in the aggregate definition, call a_n the n-th arrival
time, d_n - the n-th departure time, and l_n the size of the n-th
packet to depart. Define by convention $d_0=0$. The aggregate
definition with rate R and latency E_a is equivalent to saying that
for all n and all $0<=j<= n-1$:

    d_n <= E_a + d_j + (l_(j+1) + ... + l_n)/R   (eq_b1)

or

there exists some j+1 <= k <= n such that

    d_n <= E_a + a_k + (l_k + ... + l_n)/R   (eq_b2)

Theorem B2.   Characterization of packet-identity-aware definition
              without F_n.

   Consider a system where packets are numbered 1, 2, ... in order of
   arrival. As in the packet-identity-aware definition, call A_n, D_n
   the arrival and departure times for the n-th packet, and L_n the size
   of this packet. Define by convention $D_0=0$. The packet identity
   aware definition with rate R and latency E_p is equivalent to saying
   that for all n and all $0<=j<= n-1$:

      $$D_n <= E_p + D_j + (L_{j+1} + ... + L_n)/R$$ (eq_b3)

   or

   there exists some $j+1 <= k <= n$ such that

      $$D_n <= E_p + A_k + (L_k + ... + L_n)/R$$ (eq_b4)

   For the proofs of both Theorems, see [13].

Acknowledgements

Authors' Addresses

   Anna Charny
   Cisco Systems
   300 Apollo Drive
   Chelmsford, MA 01824

   EMail: acharny@cisco.com

   Jon Bennett
   Motorola
   3 Highwood Drive East
   Tewksbury, MA 01876

   EMail: jcrb@motorola.com

   Kent Benson
   Tellabs Research Center
   3740 Edison Lake Parkway #101
   Mishawaka, IN 46545

   EMail: Kent.Benson@tellabs.com

   Jean-Yves Le Boudec
   ICA-EPFL, INN
   Ecublens, CH-1015
   Lausanne-EPFL, Switzerland

   EMail: jean-yves.leboudec@epfl.ch

   Angela Chiu
   Celion Networks
   1 Sheila Drive, Suite 2
   Tinton Falls, NJ 07724

   EMail: angela.chiu@celion.com

   Bill Courtney
   TRW
   Bldg. 201/3702
   One Space Park
   Redondo Beach, CA 90278

   EMail: bill.courtney@trw.com

   Shahram Davari
   PMC-Sierra Inc
   411 Legget Drive
   Ottawa, ON K2K 3C9, Canada

   EMail: shahram_davari@pmc-sierra.com

   Victor Firoiu
   Nortel Networks
   600 Tech Park
   Billerica, MA 01821

   EMail: vfiroiu@nortelnetworks.com

   Charles Kalmanek
   AT&T Labs-Research
   180 Park Avenue, Room A113,
   Florham Park NJ

   EMail: crk@research.att.com

   K.K. Ramakrishnan
   TeraOptic Networks, Inc.
   686 W. Maude Ave
   Sunnyvale, CA 94086

   EMail: kk@teraoptic.com

Full Copyright Statement

Acknowledgement