Delay Jitter Bounds and Packet Scale Rate Guarantee for Expedited Forwarding

Jon C. R. Bennett, Kent Benson, Anna Charny, William F. Courtney, Jean-Yves Le Boudec

Abstract— **We consider the definition of the Expedited Forwarding Per-Hop Behaviour (EF PHB) as given in RFC 2598 [1], and its impact on worst case end-to-end delay jitter. On one hand, the definition in RFC 2598 can be used to predict extremely low end-to-end delay jitter, independent of the network scale. On the other hand, we find that the worst case delay jitter can be made arbitrarily large, if we allow networks to become arbitrarily large; this is in contradiction with the previous statement. We analyze where the contradiction originates, and find the explanation. It resides in the fact that the definition in RFC 2598 is not easily implementable in schedulers we know of, mainly because it is not formal enough, and also because it does not contain an error term. We propose a new definition for the EF PHB, called "Packet Scale Rate Guarantee", which preserves the spirit of RFC 2598, while allowing a number of reasonable implementations, and has very useful properties for per-node and end-to-end network engineering. We show that this definition is stronger than the rate-latency service curve guarantee. Then we propose some proven bounds on delay jitter for networks implementing this new definition, both in cases without loss and with loss.**

Keywords— **Differentiated Services; Expedited Forwarding; Delay Jitter**

I. INTRODUCTION

We consider the Expedited Forwarding (EF) service, defined by the IETF in the context of differentiated services. The aim of EF is to provide low delay and virtually no loss to some flows, without per flow queuing. The underlying principle of EF is to ensure that at each hop the aggregate of traffic requiring EF treatment receives a service rate exceeding the total bandwidth requirements of all flows in the aggregate at this hop. Recently, many practical implementations of EF PHB have been suggested, where all EF traffic is shaped and policed at the backbone ingress, while in the core equipment all EF traffic shares a single priority FIFO or single high-weight queue in a Class-Based Fair Queuing scheduler. Since these implementations offer a very high degree of scalability at comparatively low price, they are naturally very attractive.

Author Affiliations: Jon Bennett <jcrb@riverdelta.com> River Delta Networks, 3 Highwood Drive, Tewksbury, MA 01876; Kent Benson <Kent.Benson@tellabs.com> Tellabs Research Center, 3740 Edison Lake Parkway #101, Mishawaka, IN 46545; Anna Charny <acharny@cisco.com>, Cisco, Chelmsford, MA 01776; William F. Courtney <Bill.Courtney@trw.com> TRW, Bldg 201/3702, One Space Park, Redondo Beach, CA 90278; Jean-Yves Le Boudec <leboudec@epfl.ch> EPFL-DSC, CH-1015 Lausanne, Switzerland

More precisely, RFC2598 [1] defines the Expedited Forwarding Per-Hop Behaviour (PHB) as follows

Definition I.1 (RFC2598) "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 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."

The intuitive content of this definition is fairly clear. On all time scales ranging down to very small time scales, the EF aggregate should be given at least its fair share of the output link bandwidth. Among other things, this allows EF to support applications that are delay- and jitter-sensitive.

Definition I.1 has been used in [2] to propose a service called "Virtual Wire" which aims to provide a very low end-to-end delay jitter to some flows. The delay jitter is defined as the variable part of delay; in [2] as well as in this paper, it is equated with queuing delay (thus ignoring delay variations due to route changes or to the nature of the physical layer on radio links). Definition I.1 is used in [2] as the basis for showing that, in an arbitrary network, the end-to-end delay jitter is bounded by αT, where T is the assumed packet inter-emission time for sources using the Virtual Wire service, and α is a bound on the utilization factor on very link. The utilization factor is defined as the ratio between the maximum sustainable rate for the aggregate of all flows using the Virtual Wire service, and the configured rate at this link.

In this paper we first analyze a network which exhibits a behaviour very different from what can be expected based on the previous paragraph. Indeed, in Section II, we construct a family of networks, with a unique service class and constant rate links, which violates the expected result. More precisely, for any arbitrary large D, we can exhibit one network in this family for which the worst case end-

to-end delay jitter is larger than D, while the packet inter-emission time T and the utilization factor α remain constant (with $\alpha < 1$). This example, while being of very artificial nature, reveals a contradiction.

In Section III, we find that the contradiction comes from Definition I.1 itself. We identify that the schedulers/configuration rates on which the definition can be implemented are quite limited. We show that these difficulties are not correctable with any simple fix, and that they prevent network engineers from configuring, advertising, or analyzing the EF service.

Fortunately, there is an alternative definition, the packet scale rate guarantee, that captures the intuitive content of Definition I.1 and is implementable using a number of existing schedulers. It also admits quantitative compliance testing. In Section IV, we define the packet scale rate guarantee, which we propose as alternative to Definition I.1. This new definition has two parameters, a rate R (in bits per second) and an error term E (in units of time). We establish some of its properties, in particular in terms of delay jitter. We show that it implies, but is stronger than, the well known guaranteed rate scheduler definition, and therefore it satisfies the rate-latency service curve property defined for example in [3], [4], [5], [6]. We also give a catalog of schedulers which satisfy our definition and give their corresponding error terms. Hierarchical schedulers have particularly low error terms.

In Section V we give some bounds on the worst-case end-to-end delay jitter that can be expected from a network of nodes that satisfy our definition and preserve packet sequence. First, we consider the lossless case. We find delay jitter bounds under some assumptions on the maximum utilization factor. These bounds exploit the fact that the packet scale rate guarantee implies the service rate guarantee. We apply network calculus [4], [5], [6] to obtain the bounds. Second, we consider a network with losses. We obtain a bound by using the relation between queue length and delay jitter shown in Section IV. The bounds in this Section are deterministic, thus we expect that it is possible to give better bounds that would be true only with some probability; this is left for further work.

Proofs of the Theorems in Section IV are in the appendix.

II. A FAMILY OF NETWORK EXAMPLES WITH ARBITRARY LARGE DELAYS

In this section we present a family of networks where the delay jitter is arbitrarily large.

We consider a family of networks with a single traffic class (the EF class) and constant rate links, all with same bit rate C. The network is assumed to be made of infinitely fast switches, with one output buffer per link. Assume that sources are all leaky bucket constrained, but are served in an aggregate manner, first in first out. Leaky bucket constraints are implemented at the network entry; after that point, all flows are aggregated. Without loss of generality, we also assume in this paper that propagation delays can be set to 0; this is because we focus only on queuing delays (see [4], [6] for a discussion). As a simplification, we also assume that all packets have a unit size. In Section V, we give the result that, for such networks, the end-to-end delay jitter is bounded by $\frac{hM}{1-(h-1)\alpha}$, where α is the utilization factor, h is the maximum hop count for any flow, and M is a constant which depends on the packet size and the ratio between leaky bucket depth and rate, under the assumption that $\alpha < \frac{1}{h-1}$. In this section, we consider a family of networks with $\frac{1}{h-1} < \alpha < 1$ (thus do not satisfy the previous inequality). We show that for any fixed, but arbitrary delay jitter budget D, we can build a network of that family where the worst case delay jitter is larger than D. This will show a contradiction with the result on delays in [2] mentioned in the introduction.

This family of networks was first introduced in [7]. We give here a slight variant. A network in our family is called $\mathcal{N}(h, \alpha, J)$ and has three parameters: h (maximum hop count for any flow), α (utilization factor) and J (recursion depth). We focus on the cases where $h \geq 3$ and $\frac{1}{h-1} < \alpha < 1$, which implies that we can always find some integer k such that

$$\alpha > \frac{1}{h-1}\frac{kh+1}{kh-1} \tag{1}$$

Network $\mathcal{N}(h, \alpha, J)$ is illustrated in Figures 1 and 2; it is a collection of identical building blocks, arranged in a tree structure of depth J. Every building block has one internal source of traffic (called "transit traffic"), $kh(h-1)$ inputs (called the "building block inputs"), $kh(h-1)$ data sinks, $h-1$ internal nodes, and one output. Each of the $h-1$ internal nodes receives traffic from kh building block inputs plus it receives transit traffic from the previous internal node, with the exception of the first one which is fed by the internal source. After traversing one internal node, traffic from the building block inputs dies in a data sink. In contrast, transit traffic is fed to the next internal node, except for the last one which feeds the building block output (Figure 1). Figure 2 illustrates that our network has the structure of a complete tree, with depth J. The building blocks are organized in levels $j = 1, ..., J$. Each of the inputs of a level j building block ($j \geq 2$) is fed by the output of one level $j-1$ building block. The inputs of level 1 building blocks are data sources. The output of one $j-1$ building block feeds exactly one level j building block input. At level J, there is exactly one building block, thus

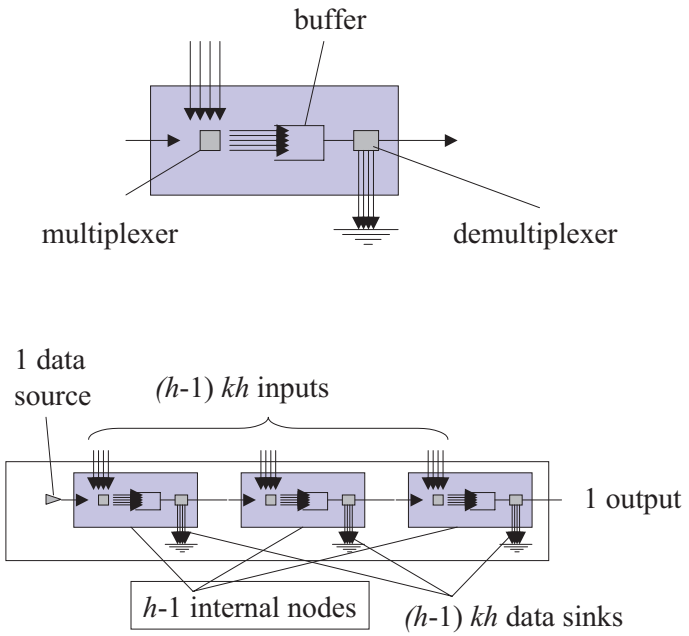

Fig. 1. The internal node (top) and the building block (bottom) used in our network example.

at level $J-1$ there are $kh(h-1)$ building blocks, and at level 1 there are $(kh(h-1))^{J-1}$ building blocks. All data

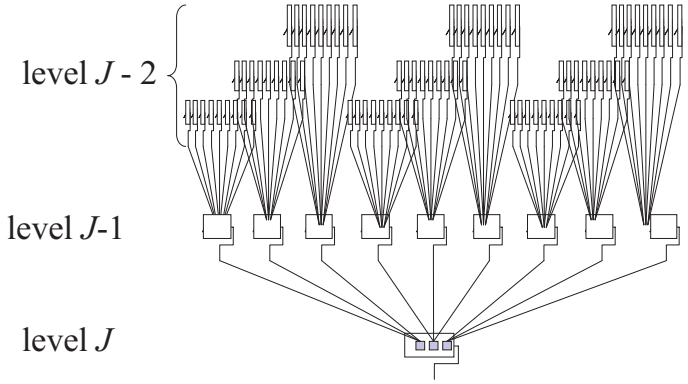

Fig. 2. The network made of building blocks from Figure 1

sources have the same rate $r = \frac{\alpha C}{kh+1}$ and burst tolerance $b = 1$ packet. In the rest of this section we take as time unit the transmission time for one packet, so that $C = 1$. Thus any source may transmit one packet every $\tau = \frac{kh+1}{\alpha}$ time units. Note that a source may refrain from sending packets, which is actually what causes the large delay jitter. The utilization factors on all links is α, and every flow uses 1 or h hops.

Now consider the following scenario. Consider some arbitrary level 1 building block. At time t_0, assume that a packet fully arrives at each of the inputs of all leftmost internal nodes inside building blocks in level 1, and at time t_0+1, let a packet fully arrive from each data source inside every level 1 building block (this is the first transit packet). The first transit packet is delayed by $hk-1$ time units in the first internal node. Just one time unit before this packet leaves the first queue, let a packet fully arrives at each input of the second internal node. Our first transit packet will be delayed again by $hk-1$ time units. If we repeat the scenario along all internal nodes inside the building block, we see that the first transit packet is delayed by $(h-1)(hk-1)$ time units. Now from (1), $\tau < (h-1)(hk-1)$, so it is possible for the data source to send a second transit packet at time $(h-1)(hk-1)$. Let all sources mentioned so far be idle, except for the emissions already described. The second transit packet will catch up the first one, so the output of any level 1 building block is a burst of two back-to-back packets. We can choose t_0 arbitrarily, so we have a mechanism for generating bursts of 2 packets.

Now we can iterate the scenario and use the same construction at level 2. The level-2 data source sends exactly three packets, spaced by τ. Since the internal node receives hk bursts of two packets originating from level 1, a judicious choice of the level 1 starting time lets the first level 2 transit packet find a queue of $2hk-1$ packets in the first internal node. With the same construction as in level 1, we end up with a total queuing delay of $(h-1)(2hk-1) > 2(h-1)(hk-1) > 2\tau$ for that packet. Now this delay is more than 2τ, and the first three level-2 transit packets are delayed by the same set of non-transit packets; as a result, the second and third level-2 transit packets will eventually catch up the first one and the output of a level 2 block is a burst of three packets. This procedure easily generalizes to all levels up to J. In particular, the first transit packet at level J has an end-to-end delay of at least $J\tau$. Since all sources become idle after some time, we can easily create a last level J transit packet which finds an empty network and thus a zero queuing delay.

Thus there are two packets in network $\mathcal{N}(h,\alpha,J)$, with one packet having a delay larger than $J\tau$, while the other packet has zero delay. This establishes that a bound on delay jitter in network $\mathcal{N}(h,\alpha,J)$ has to be at least as large as $J\tau$. This example contradicts the application of Definition I.1 made in [2], which would in this case predict a delay jitter bounded by $\tau\alpha$. Since J can be arbitrarily large, and τ does not depend on J, we have a contradiction. As explained in Section III-A.5, the contradiction comes from the fact that, contrary to the expected intuition, Definition I.1 does not apply to this example. In Section IV-C.1 we show that the new definition proposed in this paper does not have this problem.

III. Analysis of Definition I.1

We now show that the contradiction comes from Definition I.1, which is not satisfied by the nodes in our network, nor by most reasonable implementations of the EF "intuition". We first start with an analysis of some difficulties caused by Definition I.1 then argue that incremental fixes are not possible.

A. A Short List of Difficulties with Definition I.1

A literal interpretation of Definition I.1 would consider the following three behaviors non-compliant.

A.1 Perfectly-Clocked Forwarding:

Consider the following stream forwarded from a node with 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 $3\text{MTU}/C$, which is greater than $\text{MTU}/(C/2)$, and so is subject to the EF PHB definition. During this interval, $3\text{MTU}/2$ bits of the EF aggregate should be forwarded, but only MTU bits are forwarded.

A.2 No EF Packets to Forward

Consider a node configured as in the previous example, and suppose that no EF traffic is offered to the node. Naturally, it produces the output

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

As before, the interval between the vertical bars is $3\text{MTU}/C$ and is subject to the EF PHB definition, and again an insufficient number of EF bits are forwarded.

Outrageous as this last example may be, it indicates that a change to the definition is warranted. A related, slightly more complicated, example illustrates that the exposed difficulty is not as trivial as one might hope.

A.3 Server Internal Delay

Consider a node configured as in the previous examples, but with an internal delay of $3T$, with $T = \text{MTU}/C$, between the time that a packet arrives at the node and the time that it is first eligible for forwarding. Such things as header processing, route look-up, and delay in switching through a multi-layer fabric could cause this delay. Now imagine that EF traffic arrives regularly at a rate of $\frac{2}{3}R = \frac{C}{3}$. The node will perform as shown below.

EF Packet Number	1	2	3	4	5
Arrival(at node)	0	3T	6T	9T	12T
Arrival(at scheduler)	3T	6T	9T	12T	15T
Departure	4T	7T	10T	13T	16T

Again, the output does not satisfy Definition I.1, even though the node is always backlogged. As with the previous example, the node cannot forward EF traffic faster than it arrives. This example is important because the obvious simple fixes discussed in section IV do not make this example compliant.

A.4 Maximum Configurable Rate and Provisioning Efficiency

With any non-preemptive scheduler, the maximum compliant configurable rate for a EF aggregate is $C/2$ [2]. 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 $[t, t + 2\text{MTU}/C]$ is MTU. But if $R > C/2$, then this interval would be of length greater than MTU/R, and more than MTU EF bits would have to be served during it. Thus, R must be no greater than $C/2$. Without significant over-provisioning, even this rate can be configured only with Priority Queueing (PQ) where the EF aggregate is assigned the highest priority. The behavior given in the perfectly-clocked-forwarding example could be obtained with a time-division multiplexed (TDM) circuit, but the EF-configured rate could be R/3 at most. In other words, the TDM circuit in this example would have to be 50% over-provisioned just to satisfy a strict interpretation of the definition. This is by no means the worst-case over-provisioning requirement.

A.5 A FIFO single class network

This is the example in Section II. Consider level j and the output link used by the transit flow (point 1 on Figure 1). There are j idle packet transmission times, followed by one packet transmission, thus the configured rate is at most $\frac{C}{j+1}$, which contradicts the EF intuition (the network is dedicated to EF traffic, thus the rate should be C), and explains the contradiction in Section II.

B. The Non-trivial Nature of the Difficulties

Upon the discovery of the examples given above and others like them, it was the authors' hope that simple fixes to Definition I.1 would eliminate them. For instance, we hoped that the first example could be corrected by a different definition of the intervals to which the definition applied or by averaging over intervals. However, it soon

became clear that any such redefinition of applicable intervals leads to considerable implementation difficulties. In essence, assuring that interval start and finish times are properly aligned with epochs of the forwarded stream is fraught with the risk of misinterpretation and mistake in practice. Averaging of intervals leads to long-term guarantees, but annihilates the short-term guarantees that are the essence of EF PHB.

The authors also explored two further 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 node (i.e., when an EF packet is in the node). The second is the addition of a latency term that could serve as a figure-of-merit in the advertising of EF services. That term could be expressed as

- In any interval of time $[t_1, t_2]$ in which EF traffic is continuously backlogged, at least $R(t_2 - t_1 - E)^+$ bits of EF traffic must be served, where R is the configured rate for the EF aggregate and E is an implementation-specific error (or latency) term.

The "continuously backlogged" condition eliminates the no-packets-to-forward difficulty, while the addition of a latency term of size MTU$/C$ resolves the perfectly-clocked-forwarding example in Section III-A.1. However, neither fix (nor the two of them together) resolves the third example. The EF aggregate is continuously backlogged and no finite latency term will suffice to bring the example into conformance.

IV. Packet Scale Rate Guarantee, a new definition for EF PHB

In this section we introduce our proposed definition, study its properties, and show how some schedulers satisfy it.

A. Packet Scale Rate Guarantee, a formal definition

The intent of EF PHB is to provide the configured service rate to the EF aggregate at as small a timescale as possible. In order to express this notion rigorously, we introduce the definition of "packet scale rate guarantee".

We first need some notation.

Let $P_{in}(j)$ and $P_{out}(j)$ denote the j-th packet of the EF aggregate arriving to and departing from a network node respectively. In the case when all EF traffic shares a single FIFO queue, $P_{in}(j)$ and $P_{out}(j)$ refer to the same packet, but in general the j-th arrival and the j-th departure may correspond to different packets.

Let $a(j)$ denote the time of arrival of the last bit of $P_{in}(j)$ to a network node. Let $d(j)$ denote the time of departure of the last bit of $P_{out}(j)$ from the network node. Let $L(j)$ denote the length of $P_{out}(j)$.

We require that the indexing is chosen in such a way that the packet $P_{in}(1)$ arriving at time $a(1)$ sees no other packet of the EF aggregate in the node upon arrival, and $d(1) \geq a(1)$.

Definition IV.1 (Packet Scale Rate Guarantee) We say that a node offers to the EF aggregate a "packet scale rate guarantee R with latency E" if the j-th departure time satisfies the following condition for all $j \geq 0$:

$$d(j) \leq F(j) + E \tag{2}$$

where $F(j)$ is defined iteratively by

$$\begin{aligned} &F(0) = 0, d(0) = 0 \\ &\text{for all } j > 0 : \\ &F(j) = \max\left[a(j), \min\left(d(j-1), F(j-1)\right)\right] + \tfrac{L(j)}{R} \end{aligned} \tag{3}$$

Note that the choice of indexes does not restrict when in the actual packet stream we start the observation of the arrival and departure process. The only restriction that is being imposed is that the observation starts when there are no EF packets in the node.

We now define the EF PHB as a forwarding treatment for a particular diffserv aggregate where the node offers to the aggregate a packet scale rate guarantee R with latency E, where R is a configurable rate and E is a tolerance which depends on the particular node characteristics.

B. Properties of Definition IV.1

B.1 Delay as a function of queue length

It is first important to note that just as Definition I.1, the new definition does not in itself guarantee per-packet delay for the EF aggregate. While the definition implies that the aggregate service is within a certain error from the desired service at the configured rate, the definition says nothing about per-packet delay. In particular, for non-FIFO service order for packets within the EF aggregate it is possible in principle that a scheduler satisfying the EF definition (both new and old) delays a given packet an infinite amount of time.

However, if the node serves packets in order of arrival, then a simple relation exists between queue length and maximum delay. The proof is given in the appendix.

Theorem 1: If a scheduler conforms to Definition IV.1, and the EF packets are served in FIFO order, then all EF packets that are in the system at time t will leave the system no later than at time $t + Q/R + E$, where Q is the total EF backlog at time t.

B.2 Service Curve Property

We compare our definition with the rate-latency service curve property [3], [4], [5], [6] used in the Integrated Services context. It can be shown [4] that the rate-latency curve is equivalent to the following conditions (we use the same notation as with Definition IV.1):

$$d(j) \leq F'(j) + E \tag{4}$$

where $F'(j)$ is defined iteratively by

$$\begin{aligned} &F'(0) = 0 \\ &\text{for all } j > 0 : \\ &F'(j) = \max[a(j), F'(j-1)] + \tfrac{L(j)}{R} \end{aligned} \tag{5}$$

It can be easily verified that Definition IV.1 is stronger than the rate-latency service curve, namely if a scheduler satisfies Definition IV.1 it also satisfies the rate-latency curve. For this, simply notice that for all j, $F'(j) \geq F(j)$. As a result, all the properties known for the rate-latency curve also apply to our definition (see Section V for an application).

Our interest in the service curve guarantee concept is that we know good end-to-end delay jitter bounds for it. However, we now argue why Definition IV.1 is more suitable to reflect the intent of EF PHB than the rate-latency curve. Theorem 1 and Section V-B also illustrate a bound which is not obtainable with the rate-latency service curve.

It is easy to see that $F'(j)$ 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.

While (5) guarantees the desired rate to the EF aggregate in all intervals $[0, t]$ within a specified error term, 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. 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 $(N+1)L/R + E - L/C$. 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 or N is large. In essence, the EF aggregate can be "punished" by a gap in service for receiving faster service than its configured rate at the beginning.

Definition IV.1 alleviates this problem by introducing the term $\min(d(j-1), F(j-1))$ in the recursion. In particular, this allows Theorem 1 to hold, while it is easy to see that it does not for the rate-latency service curve property.

C. Satisfiability of the definition

We show in this section that a wide variety of schedulers satisfy our definition.

C.1 Priority Scheduler (PQ)

Theorem 2: A strict priority scheduler in which all EF packets share a single FIFO queue with total output rate C, which is served at strict non-preemptive priority over other queues satisfies Definition IV.1, with rate $R = C$ and error term $E = \frac{L_{max}}{C}$. In the formula, L_{max} is the maximum packet size of non-EF packets.
The proof is given in the appendix. This shows that the network in Section II satisfies the packet scale rate guarantee with rate C and error term $E = 0$.

C.2 Packet Based Implementations of WFQ

A wide family of schedulers can be thought of as derived from Weighted Fair Queueing (WFQ) [8]. More precisely, we say that a scheduler is derived from WFQ if we can compare its accuracy with respect to the reference fluid scheduler when both are subject to the same arrival patterns, in the following sense. Call $d(i)$ the time of the i-th departure under scheduler S, and $G(i)$ the i-th departure in the reference GPS scheduler with rate R allocated to the flow. The accuracy of S with respect to GPS is determined by two error terms E_1 and E_2 such that for all i

$$G(i) - E_1 \leq d(i) \leq G(i) + E_2 \tag{6}$$

The term E_2 determines the maximum per-hop delay bound, whereas E_1 has an effect on the jitter at the output of the scheduler. For example, it is shown in [9] that W2FQ satisfies $E_1(\text{W2FQ}) = L_{max}/R$, $E_2(\text{W2FQ}) = L_{max}/C$, where R is the rate allocated to a flow (here an EF aggregate) and C is the total output rate. In contrast, for PGPS [8] $E_2(\text{PGPS}) = E_2(\text{W2FQ})$, while $E_1(\text{PGPS})$ is linear in the number of queues in the scheduler. This illustrates that, while WF2Q and PGPS have the same delay bounds, PGPS may result in substantially burstier departure patterns. A systematic collection of error terms E_1 and E_2 for all known schedulers is work in progress.

Theorem 3: If a scheduler satisfies (6), then it satisfies Definition IV.1 with rate R and error term $E = E_1 + E_2$.
The proof is given in appendix.

C.3 The new definition avoids the difficulties in Section III-A

Definition IV.1 avoids the difficulties mentioned in Sections III-A.1 and III-A.2 because it does not attempt to qualify the output rate, but rather considers an input-output relationship.

Node internal delays are accounted for by adding the delay to the error term E. Consider for example the case in Section III-A.3 and assume that the scheduler (which is left unspecified in the example) offers a packet scale rate guarantee with rate R and error term E (for example we may assume it is a scheduler derived from WFQ, see Section IV-C.2). Then the complete node offers a packet scale rate guarantee with rate R and error term $E + 3T$.

The problem mentioned in Section III-A.4 does not exist anymore, as can be seen by examining the results in Sections IV-C.1 and IV-C.2. Finally, the problem mentioned in Section III-A.5 is solved as mentioned in Section IV-C.1.

V. Some Bounds on Delays achievable with Packet Scale Rate Guarantee

In this section we consider a network of arbitrary topology, offering to the EF class a service according to Definition IV.1. We first find a bound on delay jitter for the case where network buffers are large enough to avoid any loss, then we consider the case with losses.

A. Lossless case

There is a widespread belief, based largely on intuition, that as long as the utilization on any link is kept small enough (such as less than 50%), the worst case delay through the network will be very small. Unfortunately, this intuition leads to erroneous conclusions. The question of delay bounds for a network with aggregate scheduling was raised in [10]. In general, the set of utilization factors for a session oriented network with aggregate scheduling that keeps the network stable[1] is not known. A partial result is in [11], which shows that a uni-directional ring is stable for any utilization factor less than 1, assuming that there is only type of traffic and links are dedicated. The proof does not hold, even for a ring, for a GPS scheduler, therefore it does not hold for the definition we are proposing for EF. In [12], the author exhibits an unstable, session-oriented network, with utilization factor 1 (a "critical" network), and claims that this also holds for some sub-critical networks, but the proof is not conclusive. In addition, a startling fact referred to as a "non-monotone property" of FIFO networks is described in [12], where it is shown that a network that is *stable* with a set of sessions with given rates may become *unstable* if the rates of some of these sessions are *reduced* for a period of time.

On the other side of the spectrum, [13], [14] demonstrate that the ability to provide good delay bounds may depend on complex global conditions. In particular, this work assumes that individual flows are shaped at the network entry in such a way that the spacing between packets is at least equal to the so-called route interference number (RIN)[2]. It is shown that in this case the end-to-end result is bounded by the time to transmit a number of packets equal to the RIN.

Considering a network where the EF aggregate is served at all nodes in accordance to Definition IV.1, we can apply Section IV-B.2 and exploit the fact that a service curve property holds. Indeed, delay bounds for such a case are given in [7]. We recall them here.

The assumptions for the bound are as follows.

- Each end to end EF flow f is shaped to conform to a leaky bucket with parameters (ρ_f, σ_f) when it arrives at the ingress edge. Note that the flow can itself consist of a number of microflows sharing the same ingress-egress edge pair, but no assumption is made on how those microflows are shaped.
- The node serving link l offers to the EF aggregate a packet scale rate guarantee R_l with error term E_l. Let E be a bound on all E_l (namely, $E \geq \max_l E_l$).
- Let $S(l)$ denote the set of all priority EF flows constituting the EF aggregate on link l. It is assumed that the amount of EF traffic on any link does not exceed a certain ratio $\alpha < 1$ of the configured rate on any link. More specifically it is required that for any link l in the network $\sum_{f \in S(l)} \rho_f \leq \alpha R_l$.
- For any link l let $\tau_l = \frac{1}{R_l} \sum_{f \in S(l)} \sigma_f$, and let τ be a bound on all τ_l.
- The route of any flow in the network traverses at most h nodes (also referred to as hops)
- Let P_l denote a bound on the peak rate of all incoming EF traffic at link l. If we have no information about this peak rate, then $P_l = +\infty$. For a router with large internal speed and buffering only at the output, P_l is the sum of the bit rates of all incoming links. The delay bound is better for a smaller P_l.
- Let $u_l = \frac{P_l - R_l}{P_l - \alpha R_l}$. Note that $0 < u_l \leq 1$, u_l increases with P_l, and if $P_l = +\infty$, then $u_l = 1$. Call $u = \max_l u_l$.

Then we have:

Theorem 4 ([7]) If $\alpha < \min_l \frac{P_l}{(P_l - R_l)(h-1) + R_l}$ then a

[1] I.e. that ensure that queues in the network are bounded

[2] RIN is defined as the number of occurrences of a flow joining the path of some other flow.

bound on the end-to-end delay jitter for EF traffic is

$$D = \frac{h}{1-(h-1)u\alpha}\left(E + u\tau\right)$$

If we have no information about the peak incoming rate P_l, then we set $P_l = +\infty$ and the theorem tells us that, for $\alpha < \frac{1}{h-1}$, a bound on delay is $\frac{h}{1-(h-1)\alpha}\left(E+\tau\right)$.

If the condition on the utilization factor α in Theorem 4 is not satisfied, then it is not clear whether finite bounds exist, and if so, what they would be. Section II is precisely about the case where Theorem 4 does not apply. It shows that the worst case delay jitter can be made arbitrarily large; thus if such bounds exist, then they must depend on the network topology or size, not only on the utilization factor and the number of hops as in Theorem 4.

B. Delay Jitter Bounds for lossy systems

Consider the same assumptions as in Section V-A with one exception. Assume that the buffer size B_l dedicated to the EF aggregate at link l is limited, and the buffer may thus overflow. A direct application of Theorem 1 gives a bound hD^* on delay jitter, where D^* is a bound on delay jitter at one node:

$$D^* = \max_l \left(\frac{B_l}{R_l} + E_l\right)$$

In this case, another quantity of interest is the loss ratio. It is proposed in [15] to upper bound the loss ratio at a node l by

$$\lambda_l = 1 - \frac{B_l}{B_l'} \tag{7}$$

where B_l' is the buffer which would be required for a loss free operation. B_l' can be estimated by the network calculus method used in [7], as follows. A total arrival curve for EF traffic arriving at node l has as an upper bound the arrival curve $a(t) = \min(P_l t, \alpha R_l t + b_l)$, with $b_l = R_l(\tau + (h-1)\alpha D^*)$. It follows that

$$\begin{aligned} B_l' = \max \quad \{\min[\quad & P_l E_l, R_l(\tau + \alpha(h-1)D^* + E_l], \\ \min[\quad & P_l \tfrac{u-1}{\alpha-1}(\tau + \alpha(h-1)D^*), \\ & R_l(u(\tau + \alpha(h-1)D^*) + E_l)]\} \end{aligned} \tag{8}$$

Combining (7) and (8) would give an upper bound of the loss ratio at one node. The accuracy of this estimation is work in progress. Alternative methods based on exponentially bounded burstiness [16] will also be examined.

VI. Conclusion

While the intuitive content of the current definition of EF is fairly clear, we have argued that it is not readily operational. In order for vendors, operators, service providers, and users to exploit EF, they must be able to build nodes that provide quantifiable EF service, to configure those nodes, to advertise accurately the capabilities and capacities of the offered EF service, and to determine the per-node and end-to-end service characteristics. None of this is possible unless the definition of EF admits quantitative compliance testing.

As it stands, the RFC 2598 definition does not admit such testing. Fortunately, there is an alternative definition, the packet scale rate guarantee, that captures the intuitive content of the RFC 2598 definition and also admits quantitative compliance testing. We have introduced this new definition and shown that there are well established scheduling policies to which it applies. We have also given some explicit, deterministic bounds on delay jitter, in cases with or without loss. The sharpness of the bounds is a topic of its own which is not fully addressed in this paper. However we have indicated that better bounds must depend on finer descriptions of the network, which may be beyond the intention of differentiated services.

Appendix

I. Proof of Theorem 1

Consider time t. Let $a(1), ..., a(n)$ denote the arrival times of all EF packets that are in the system at time t. For all of these packets $a(i) \leq t$. Let $d(0) \leq t$ denote the last departure time before t. Let $d(1), ..., d(n)$ denote the first n departures after time t, let $L(1), ...L(n)$ denote the packet lengths corresponding to these departures and let $F(1), ..., F(n)$ denote the corresponding "finish times" in Definition IV.1. We now prove by induction that $F(i) \leq t + \frac{L(1)+...L(i)}{R}$ for all $1 \leq i \leq n$.

Base case: Since $a(1) < t$, it is easy to see that

$$\begin{aligned} F(1) &= \max[a(1), \min(d(0), F(0))] + \tfrac{L(1)}{R} \\ &\leq \max[a(1), d(0)] + \tfrac{L(1)}{R} \\ &\leq t + \tfrac{L(1)}{R} \end{aligned}$$

Inductive step: Suppose $F(i) \leq t + \frac{L(1)+...L(i)}{R}$ for all $i \leq j < n$. Then, recalling that for all $1 \leq j \leq n$, $a(j) < t$, we have

$$\begin{aligned} F(j+1) &= \max[a(j+1), \min(d(j), F(j))] + \tfrac{L(j+1)}{R} \\ &\leq \max[a(j+1), F(j)] + \tfrac{L(j+1)}{R} \\ &\leq t + \tfrac{L(1)+...L(j+1)}{R} \end{aligned}$$

Therefore, the first n departures must occur no later than $F(n) + E = t + \frac{L(1)+....L(n)}{R} + E$. QED.

II. Proof of Theorem 2

Consider any busy period of the EF queue. Let $k = 1$ correspond to the first packet in that busy period. Define $F(j)$ for all $j \geq 0$ by (3) with the value of R set to C.

We prove by induction that for all $k \geq 1$ in this busy period

$$d(k) \leq F(k) + \frac{L_{max}}{C} \tag{9}$$

This would immediately imply the theorem.

Base case: For $k = 1$,

$$\begin{array}{rl} F(1) & = \max[a(1), \min(d(0), F(0)] + \frac{L(1)}{C} \\ & \geq a(1) + \frac{L(1)}{C} \end{array} \tag{10}$$

because the first priority packet in the queue may wait at most for one largest packet transmission before its own transmission begins. It follows that

$$\begin{array}{rl} d(1) & \leq a(1) + \frac{L_{max}}{C} + \frac{L(1)}{C} \\ & \leq F(1) + \frac{L_{max}}{C} \end{array}$$

Inductive step: Note that for a packet $k > 1$ in the busy period of the EF queue

$$d(k) = d(k-1) + \frac{L(k)}{C} \tag{11}$$

Now from the induction hypothesis

$$F(k-1) \geq d(k-1) - \frac{L_{max}}{C}$$

since EF has the highest priority. Plugging this into the definition (3) of $F(k)$ gives

$$\begin{array}{l} F(k) \geq \\ \quad \max[a(k), \min(d(k-1), d(k-1) - \frac{L_{max}}{C})] + \frac{L(k)}{C} \\ = \max[a(k), d(k-1) - \frac{L_{max}}{C}] + \frac{L(k)}{C} \end{array} \tag{12}$$

It follows immediately from (12) that

$$F(k) \geq d(k-1) - \frac{L_{max}}{C} + \frac{L(k)}{C}$$

Combining with (11) shows (9) and completes the inductive step.

III. Proof of Theorem 3

We first prove that for all $i \geq 0$

$$F(i) \geq G(i) - E_1 \tag{13}$$

where $F(i)$ is the set of finish times recursively defined by (3). Indeed, if (13) holds, then from (6) and (13):

$$d(i) \leq G(i) + E_2 \leq F(i) + E_1 + E_2$$

which means that the scheduler satisfies Definition IV.1 with error term $E = E_1 + E_2$.

Proof of (13): First note that in the reference GPS system, packet i starts its service at time $\max[a(i), G(i-1)]$ and receives a service rate at least equal to R, thus

$$G(i) \leq \max[a(i), G(i-1)] + \frac{L(i)}{R} \tag{14}$$

Now the proof of (13) proceeds by induction.

Base case: $F(0) = 0$, $G(0) = 0$, so (13) trivially holds for $i = 0$.

Inductive step: Suppose (13) holds for all $j = 0, 1...i-1$, ($i \geq 1$). We have both

$$F(i-1) \geq G(i-1) - E_1$$

and from (6)

$$d(i-1) \geq G(i-1) - E_1$$

thus

$$\min[F(i-1), d(i-1)] \geq G(i-1) - E_1 \tag{15}$$

Combining this with (3), we obtain

$$F(i) \geq G(i-1) - E_1 + \frac{L(i)}{R} \tag{16}$$

Again from (3) we have

$$\begin{array}{rl} F(i) \geq & a(i) + \frac{L(i)}{R} \\ \geq & a(i) - E_1 + \frac{L(i)}{R} \end{array} \tag{17}$$

Combining (16), (17) and (14) gives

$$F(i) \geq G(i) - E_1$$

which completes the proof.

References

[1] V. Jacobson, K. Nichols, and K. Poduri, "An expedited forwarding phb," June 1999, RFC 2598, IETF.

[2] V. Jacobson, K. Nichols, and K. Poduri, ""the virtual wire" behavior aggregate," March 2000, Work in progress `www.ietf.org/internet-drafts/draft-ietf-diffserv-vw-00.txt`.

[3] Parekh A. K. and Gallager R. G., "A generalized processor sharing approach to flow control in integrated services networks: The single node case," *IEEE/ACM Trans. Networking, vol 1-3*, pp. 344–357, June 1993.

[4] J.-Y. Le Boudec, "Application of network calculus to guaranteed service networks," *IEEE Transactions on Information Theory*, vol. 44, pp. 1087–1096, May 1998.

[5] C.S. Chang, "On deterministic traffic regulation and service guarantee: A systematic approach by filtering," *IEEE Transactions on Information Theory*, vol. 44, pp. 1096–1107, August 1998.

[6] R. Agrawal, R. L. Cruz, C. Okino, and R. Rajan, "Performance bounds for flow control protocols," *IEEE/ACM Transactions on Networking (7) 3*, pp. 310–323, June 1999.

[7] Anna Charny and Jean-Yves Le Boudec, "Delay bounds in a network with aggregate scheduling," in *First International Workshop on Quality of future Internet Services*, Berlin, Germany, 2000.

[8] Parekh A. K. and Gallager R. G., "A generalized processor sharing approach to flow control in integrated services networks: The multiple node case," *IEEE/ACM Trans. Networking, vol 2-2*, pp. 137–150, April 1994.

[9] J.C.R. Bennett and H. Zhang, "Wf2q: Worst-case fair weighted fair queuing," in *Proceedings of Infocom*, Mar 1996.

[10] C. S. Chang, "Stability, queue length and delay, part i: Deterministic queuing networks," Tech. Rep. Technical Report RC 17708, IBM, 1992.

[11] L. Tassulias and L. Georgiadis, "Any work conserving policy stabilizes the ring with spatial reuse," *IEEE/ACM Transactions on Networking*, pp. 205–208, April 1996.

[12] M. Andrews, "Instability of fifo in session-oriented networks," in *Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2000)*, January 2000.

[13] J.-Y. Le Boudec and G. Hebuterne, "Comment on a deterministic approach to the end-to-end analysis of packet flows in connection oriented network," *IEEE/ACM Transactions on Networking*, February 2000.

[14] I. Chlamtac, A. Faragó, H. Zhang, and A. Fumagalli, "A deterministic approach to the end-to-end analysis of packet flows in connection oriented networks," *IEEE/ACM transactions on networking*, vol. (6)4, pp. 422–431, 08 1998.

[15] Tijani Chahed, Gerard Hebuterne, and Caroline Fayet, "Mapping of loss and delay between ip and atm using network calculus," in *Networking 2000*, Paris, France, May 2000.

[16] Z. Zhang, D. Towsley, and J.F. Kurose, "Statistical network performance guarantees with generalized processor sharing scheduling," *IEEEJournal on Selected Areas in Communications (JSAC)*, August 1995.