HAL open science

Improving RFC5865 Core Network Scheduling with a Burst Limiting Shaper

Anaïs Finzi, Emmanuel Lochin, Ahlem Mifdaoui, Fabrice Frances

To cite this version:

Anaïs Finzi, Emmanuel Lochin, Ahlem Mifdaoui, Fabrice Frances. Improving RFC5865 Core Network Scheduling with a Burst Limiting Shaper. 2017 IEEE Global Communications Conference (GLOBECOM 2017), Dec 2017, Singapore, France. pp.1-6, 10.1109/GLOCOM.2017.8254449 . hal-02540324

HAL Id: hal-02540324

https://hal.science/hal-02540324v1

Submitted on 10 Apr 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Open Archive Toulouse Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of some Toulouse researchers and makes it freely available over the web where possible.

This is an author's version published in: https://oatao.univ-toulouse.fr/18448

To cite this version :

Finzi, Anaïs and Lochin, Emmanuel and Mifdaoui, Ahlem and Frances, Fabrice Improving RFC5865 Core Network Scheduling with a Burst Limiting Shaper. In: IEEE GLOBECOM 2017, 4 December 2017 - 8 December 2017 (Singapore, Singapore).

Any correspondence concerning this service should be sent to the repository administrator:

tech-oatao@listes-diff.inp-toulouse.fr

Improving RFC5865 Core Network Scheduling with a Burst Limiting Shaper

Anaïs Finzi, Emmanuel Lochin, Ahlem Mifdaoui, Fabrice Frances

Université de Toulouse ISAE-SUPAERO, Toulouse, France
firstname.lastname@isae-supaero.fr

***Abstract*—We define a novel core network router scheduling architecture to carry and isolate time constrained and elastic traffic flows from best-effort traffic. To date, one possible solution has been to implement a core DiffServ network with standard fair queuing and scheduling mechanisms as proposed in the well-known "A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic" from RFC5865. This architecture is one of the most selected solutions by internet service provider for access networks (e.g. Customer-Premises Equipment or satellite PEP). In this study, we argue that the proposed standard implementation does not allow to efficiently quantify the reserved capacity for the AF class. By using a novel credit based shaper mechanism called Burst Limiting Shaper, we show that we can provide the same isolation for the time constrained EF class while better quantifying the part allocated to the AF class.**

I. Introduction

The DiffServ architecture [1], [4] proposes a scalable mean to deliver IP quality of service (QoS) based on handling traffic aggregates. This architecture follows the philosophy that complexity should be delegated to the network edges while simple functionalities should be located in the core network. Thus, core devices only perform differentiated aggregate treatments based on the marking set by edge devices. Keeping aside policing mechanisms that might enable edge devices in this architecture, a DiffServ stateless core network is often used to differentiate time-constrained UDP traffic (e.g. VoIP or VoD) and TCP bulk data transfer from all the remaining best-effort (BE) traffic called default traffic (DE). Following the core router architecture defined in [2] and illustrated in Fig. 1, this kind of router is widely implemented inside Customer-Premises Equipment (CPE) [12] or satellite Performance Enhanced Proxy (PEP) [5], [9] to differentiate flows with different Quality of Service (QoS) requirements. In this study, the Expedited Forwarding (EF) class is used to carry UDP traffic coming from time-constrained applications (VoIP, Command/Control, ...); the Assured Forwarding (AF) class deals with elastic traffic as defined in [1] (data transfer, updating process, ...) while all other remaining traffic is classified inside the default (DE) best-effort class.

This core router implementation provides the first and best service to EF as the priority scheduler attributes the highest priority to this class. The second service is called assured service and is built on top of the AF class where elastic traffic such as TCP traffic, is intended to achieve a minimum level of throughput [7]. Usually, the minimum assured throughput

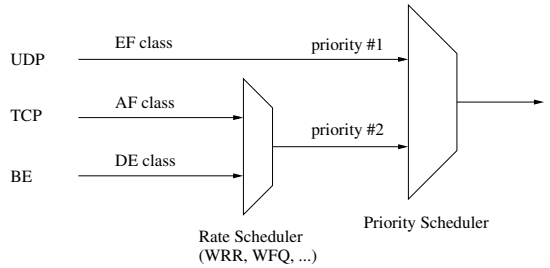


Fig. 1. DiffServ core router architecture following [2].

is given according to a negotiated profile with the client. The throughput increases as long as there are available resources and decreases when congestion occurs. As a matter of fact, a simple priority scheduler is insufficient to implement the AF service. Due to its opportunistic nature of fetching the full remaining capacity, TCP traffic increases until reaching the capacity of the bottleneck. In particular, this behaviour could lead to starve the DE class. To prevent this, the core router architecture proposed in [2] uses a rate scheduler between AF and DE classes to share the residual capacity left by the EF class. Nevertheless, one drawback of using a rate scheduler is the high impact of EF traffic on AF. Indeed, the residual capacity shared by AF and DE classes is directly impacted by the EF traffic variation. As a consequence, the AF class service is difficult to predict in terms of available capacity and latency.

To overcome these limitations, we propose in this paper an alternative architecture based on a new shaper presented by the TSN Task group [6], called the Burst Limiting Shaper (BLS). The latter belongs to the credit-based shaper category and is simple to implement. The objective of the BLS is to reserve a given capacity for the shaped priority. As with a rate scheduler, this reservation sets the capacity allocated to the shaped priority in presence of DE traffic. However, contrary to the rate scheduler, the BLS is able to enforce the reserved capacity when the EF traffic dynamically evolves over the time.

Hence, the main aim of this paper is to assess whether the BLS would provide more benefit for the AF class, in terms of quantifiable rates, while avoiding any negative impact on the performances of the EF class, in comparison to a rate scheduler, e.g., Weighted Round Robin (WRR) or Weighted

Fair Queueing (WFQ). To tackle this problem, we first give the big picture of our idea by presenting our BLS router proposal. Then, we detail the BLS shaper and present experiments and results. In particular, we show that with a given EF traffic, a correspondence between the BLS parameters and the weights of a WRR can be simply established. Finally, we present the new service offered by the BLS when the EF traffic varies and compare it to the WRR service.

II. Specification of the TSN/BLS Router

We first present in this section the Burst Limiting Shaper and later show how this scheme is used to implement a core DiffServ router. The main notations we will use in this paper are presented in Table I.

TABLE I
NOTATIONS

C	capacity of a link
L_M	BLS maximum credit level
L_R	BLS resume credit level
I_{idle}	BLS idle slope
I_{send}	BLS sending slope
BW	BLS reserved capacity
L_j^{max}	maximum length of a packet of class $j \in \{EF, AF, DE\}$
L_j^{avg}	average length of a packet of class $j \in \{EF, AF, DE\}$
W_i	WRR weight of class $i \in \{AF, DE\}$
K_i	relative weight of class $i \in \{AF, DE\}$, defined in (2)
R_j	input rate of class $j \in \{EF, AF, DE\}$
R_j^{exp}	expected input rate of class $j \in \{EF, AF, DE\}$
$R_{k/j}^{*,th}$	theoretical output rate of class $j \in \{EF, AF, DE\}$ using $k \in \{WRR, BLS\}$
$R_{k/j}^{*,sim}$	simulated output rate of class $j \in \{EF, AF, DE\}$ using $k \in \{WRR, BLS\}$

A. Definition of the BLS

The Burst Limiting Shaper (BLS) belongs to the credit-based shapers class [3]. Presented in [6], the BLS is defined by an upper threshold: L_M, a lower threshold: L_R such as $0 \leqslant L_R < L_M$, and a reserved capacity: BW.

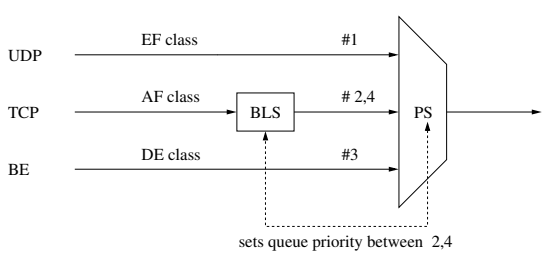


Fig. 2. Proposed output architecture

As shown in Fig. 2, the BLS must be coupled with a Priority Scheduler as it acts on the priority of the class it manages. The priority of a class j shaped by BLS denoted $p(j)$, is given by a set of priority values where the low value must be below the lowest priority of the unshaped traffic (e.g. below DE class from Fig. 2). For instance, as we have three classes represented in Fig. 2 and managed by a priority scheduler, the class shaped by the BLS will change its own priority value either to two or four, moving the AF class to a priority lower than the best-effort (i.e. the third class). Basically, the priority change depends on the credit counter of the burst limited flows as follows:

- initially, the credit counter starts at 0 and burst limited flows get its high priority (in the example Fig.2: priority #2);
- the main feature of the BLS is the change of priority $p(j)$ of the shaped queue j, which occurs in two cases:
  1) if $p(j)$ is high and the credit reaches L_M;
  2) if $p(j)$ is low and credit reaches L_R;
- when a packet is transmitted, the credit increases (is consumed) with a rate I_{send}, else the credit decreases (is gained) with a rate I_{idle};
- when the credit reaches L_M, it remains at this level until the end of the transmission of the current packet (if any);
- when the credit reaches L_R, and the transmission of the current frame finished, in the absence of BLS packets, it keeps decreasing at the rate I_{idle} until it reaches 0. The credit remains at 0 until a new BLS packet is transmitted.

The behavior of the BLS is illustrated in Fig. 3.

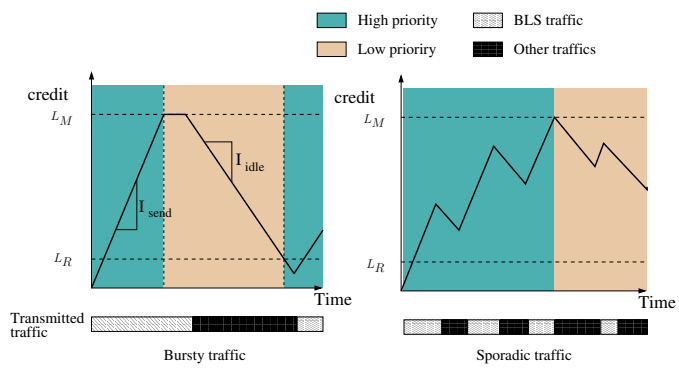


Fig. 3. BLS credit evolution

Finally, the different slopes of the BLS credit are defined as follows: the decreasing rate is $I_{idle} = BW \cdot C$, where C is the link speed and BW is the percentage of capacity reserved for BLS packets; the increasing rate is $I_{send} = (1 - BW) \cdot C$.

B. Algorithm for the TSN/BLS

As previously illustrated, the algorithm allowing to implement the BLS corresponds to a modification of the priority scheduler. The new dequeuing algorithm is presented in Algorithm 1. The credit denoted $credit$ and the dequeuing timer denoted $timerDQ$ are initialized to zero. The initial priority is set to the high value. First, in line 1, δ_{time}, the difference between the current time and the time stored in $timerDQ$, is computed. The duration δ_{time} represents the time elapsed since the last credit update, during which no shaped packet was sent, we call this the idle time. Then, if $\delta_{time} > 0$, the credit is updated by removing the credit gained during the idle time that just occurred (lines 2 and 3). Next, the $timerDQ$ is set to the current time to keep track of the time the credit is

last updated (line 4). If the credit reaches L_R, the priority changes to its high value (lines 5 and 6). Then, with the updated priorities, the priority scheduler performs as usual: each queue is checked for dequeuing (lines 9 and 10). When the BLS queue is selected, the credit expected to be consumed is added to the $credit$ variable (line 12). The time taken for the packet to be dequeued is added to the variable $timerDQ$ (lines 12 and 13) so the transmission time of the packet will not be taken into account in the idle time δ_{time} (line 1). If the credit reaches L_M, the priority changes to its low value (lines 14 and 15). Finally, the packet is dequeued (line 18).

Algorithm 1 BLS algorithm: dequeuing process

Input: $credit$; $timerDQ$;C L_M;L_R;BW;$prio_{low}$; $prio_{high}$;

1: $\delta_{time} = getcurrentTime() - timerDQ$
2: **if** $\delta_{time} > 0$ **then**
3: $credit = \max(credit - \delta_{time} \cdot BW \cdot C, 0)$
4: $timerDQ = getcurrentTime()$
5: **if** $credit \leqslant L_R$ and $prio(q_{BLS}) = prio_{low}$ **then**
6: $prio(q_{BLS}) = prio_{high}$
7: **end if**
8: **end if**
9: **for** each priority level, highest first **do**
10: **if** length(queue(p))>0 **then**
11: **if** queue(p)=q_{BLS} **then**
12: $credit$=min(L_M, $credit$+size(head(queue(p)) $\cdot(1 - BW)$)
13: $timerDQ$=$getcurrentTime()$ +size(head(queue(p))/C
14: **if** $credit \geqslant L_M$ and $prio(q_{BLS})$ = $prio_{high}$ **then**
15: $prio(q_{BLS}) = prio_{low}$
16: **end if**
17: **end if**
18: dequeue(head(queue(p)))
19: **end if**
20: **end for**

Algorithm 1 also implements the following functions:

- $getcurrentTime()$ uses a timer to return the current time;
- $queue(p)$ returns the queue associated to the priority p;
- q_{BLS} is the queue shaped by the BLS;
- $prio_{low}$ is the low priority of the BLS queue q_{BLS};
- $prio_{high}$ is the high priority of the BLS queue q_{BLS};
- $head(q)$ returns the first packet in the queue q;
- $size(f)$ returns the size of the packet f;
- $dequeue(f)$ activates the dequeuing event of packet f.

The complexity of this algorithm is the same as a priority scheduler and is $O(1)$ (the number of queues is constant).

C. Defining BLS parameters from WRR weights

The architecture proposed in RFC5865 [2] is usually implemented by vendors with a Weighted Round Robin (WRR) scheduler[1]. The rationale is that compared to WFQ, WRR provides quite similar guarantees with an easier implementation [10]. Last but not least, WRR can be implemented both in hardware and software while WFQ can only be implemented in software due to the virtual clock algorithm [10]. As said previously, the BLS has three parameters: L_M, L_R, and BW. Initially, the BLS has been proposed as a potential solution for realtime Ethernet switching where a value of $L_R > 0$ might have an interest. Actually, in the formal BLS analysis presented in [11], L_R is set to 5% of L_M. However, if we consider the model given in their paper, L_R always appears in formulas as $L_M - L_R$. As a matter of fact, the difference between L_M and L_R has much more impact than the value of L_R itself. So, in our packet switched network context, we choose to set $L_R = 0$.

To be compliant with the standard architecture, we seek to set L_M and BW to get the same characteristic than a WRR in terms of offered capacity. We call W_i the weight of class $i \in \{AF, DE\}$, which is a weight corresponding to the number of consecutive packets that can be sent. As WRR is upstream the priority scheduler, it shares the residual capacity left by the EF traffic between AF and DE. This capacity is equal to $C - R_{EF}$. As a result, the theoretical output rate of the AF class denoted $R^{*,th}_{WRR/AF}$ is:

$$R^{*,th}_{WRR/AF} = K_{AF} \cdot (C - R_{EF}) \tag{1}$$

with K_{AF} defined in (2), the relative weight of the AF class, which is the share given by WRR to AF relatively to the share given to DE, with L_i^{avg} the average length of a packet of class $i \in \{AF, DE\}$.

$$K_{AF} = \frac{W_{AF} \cdot L^{avg}_{AF}}{W_{AF} \cdot L^{avg}_{AF} + W_{DE} \cdot L^{avg}_{DE}} \tag{2}$$

We call sending window a period when AF traffic is continuously transmitting, and idle window a period between two sending windows. Our aim here is to share the residual bandwidth left by the EF traffic, in a core network router. So, we have several hypothesis: 1) the presence of EF traffic will be considered later on, 2) in a core router, the AF traffic is made of aggregated flows saturating the allocated bandwidth (the rate of a TCP flow increases until a bottleneck is reached), 3) the residual bandwidth available to DE flows is insufficient resulting in the fact that DE traffic is always available. From the definition of the BLS and our hypothesis, we deduce that in the presence of both AF and DE traffics, the minimal sending window is the time needed by the credit to increase from 0 to L_M: $\Delta^{min}_{send} = \frac{L_M}{I_{send}}$, and the minimal idle window is the time needed by the credit to decrease from L_M to 0: $\Delta^{min}_{idle} = \frac{L_M}{I_{idle}}$. Due to non-preemption, an additional packet can be sent if its

[1]Even for software based routers, WRR seems prefered. See *QoS Scheduling and Queueing on the CISCO Catalyst 3550 Switches.* configuration manual available online http://www.cisco.com/c/en/us/support/docs/lan-switching/lan-quality-of-service/24057-187.html

transmission started just before the credit reached L_M or 0. So the maximal sending window is given by:

$$\Delta_{send}^{max} = \frac{L_M}{I_{send}} + \frac{L_{AF}^{max}}{C},$$

and the maximal idle window by:

$$\Delta_{idle}^{max} = \frac{L_M}{I_{idle}} + \frac{L_{DE}^{max}}{C}.$$

From these above equations, we can deduce that if we define N such as at most $N-1$ packets of average size L_{AF}^{avg} can be fully sent during Δ_{send}^{min}, then, we have:
$(N-1) \cdot \frac{L_{AF}^{avg}}{C} \leqslant \Delta_{send}^{min} < N \cdot \frac{L_{AF}^{avg}}{C} \leqslant \Delta_{send}^{max}$.

Due to non-preemption, N packets of average size will be sent during an average sending window. The same remains true for the idle window.

During the transmission of the non-preempted packets, the credit saturates either at L_M or $L_R = 0$, and no credit is gained or lost during this time. Thus the credit is only aware of traffic sent during minimal sending and idle windows. Since the credit is responsible for maintaining the used capacity at BW, we compute BW considering only $N-1$ packets when we want to send N packets. To obtain a BLS service equivalent to WRR, we need to send W_{AF} packets during an average sending window, and W_{DE} packets during an average idle window. To achieve this, we take into account the EF traffic to compute the BLS parameter BW as follows:

$$BW = B \cdot (C - R_{EF}), \tag{3}$$

with:

$$B = \frac{(W_{AF}-1) \cdot L_{AF}^{avg}}{(W_{AF}-1) \cdot L_{AF}^{avg} + (W_{DE}-1) \cdot L_{DE}^{avg}} \tag{4}$$

Finally, since we seek to send W_{AF} packets of average size during an average sending window, we set L_M so that $W_{AF}-1$ packets can be sent during the minimum sending windows. Thus, W_{AF} packets will be sent in an average sending window:

$$\Delta_{send}^{min} = \frac{L_M}{I_{send}} = \frac{L_{AF}^{avg}}{C} \cdot (W_{AF}-1)$$
$$L_M = L_{AF}^{avg} \cdot (W_{AF}-1) \cdot (1-BW) \tag{5}$$

We have presented the BLS algorithm and showed how to compute BLS parameters following those obtained by WRR parameters. We now propose to simulate and verify the consistency of the BLS parameters in the presence of a constant bit rate (CBR) UDP traffic. Finally, we will illustrate the advantage of the BLS in the presence of dynamic UDP traffic.

III. Results and Analysis

In this part, we first explain the simulations characteristics, then we show that the BLS and WRR can have the same behavior when considering a constant UDP flow within the EF class. Finally, we show that when UDP varies, the BLS enforces a better isolation of the AF flows than WRR.

A. Simulation characteristics

We implement our proposal described in Fig. 2 in $ns2$ and simulate EF, AF and DE traffic within this core node. Each simulation lasts one minute. We consider three types of flows: UDP/EF, TCP/AF and BE/DE. For UDP/EF, we consider ITU-G.711 VoIP flows with a packet size L_{EF} of 200 Bytes and a rate of $83\,Kb/s$. In the experiment, the number of flows is varying the aggregated rate of the UDP/EF traffic, call R_{EF}. For TCP/AF, we consider a packet size L_{AF} of 1500 Bytes. We generate 80 TCP flows inside the AF class, with both CUBIC and TCP Newreno. Since we obtained identical results, we present here the results with CUBIC. Finally, BE traffic is simulated with a constant bit rate (CBR) UDP traffic with a packet size of L_{DE} of 1500 Bytes and a rate to operate at 100% load.

B. Equivalence between BLS and WRR when EF traffic is known

The objective of this section is to show that under the same known and constant UDP/EF load, both BLS and WRR have the same behavior. To do this, we study the service obtained in terms of output rate. Knowing the EF traffic, we set the parameters as explained in II-C with the residual service $C - R_{EF}$. In this experiment, we vary two parameters: the percentage of capacity allocated by the WRR to the AF class: K_{AF}, and the number of EF flows, which modifies the EF input rate R_{EF}.

1) Isolation of EF traffic: first, we study the isolation of UDP/EF traffic when varying the WRR weights and the number of EF flows. The input rate R_{EF} gives the number of UDP flows crossing the EF class (knowing that ITU-G.711 codec generates at $83\,Kb/s$). In Fig. 4(a), we can see that the EF output rate remains steadily equal to the input rate. In Fig. 4(b), we can see that the latency also remains steady on average for each UDP VoIP flows. Additionally, we can also note that BLS and WRR latency is the same.

We have shown that the EF class gets the same service whether BLS or WRR is used. We now study in the following the service offered to the AF and DE classes.

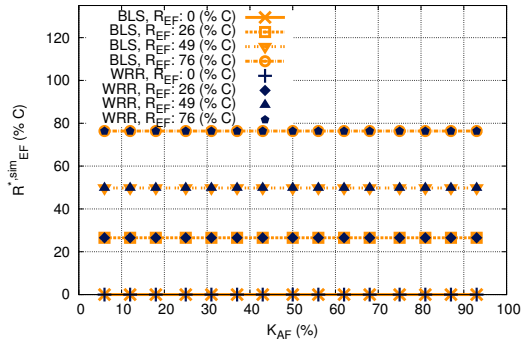
2) Performance of AF flows: as explained in II-C, the WRR shares between AF and DE classes the residual capacity left by the EF traffic. The capacity allocated to AF is defined in (4). For a tuple (W_{AF}, W_{DE}), when R_{EF} increases, the capacity allocated to AF decreases. Fig. 5 confirms this theoretical behavior. As BLS curves fit WRR ones, we can deduce that both schemes have the same behavior. It follows that:

$$R_{BLS/AF}^{*,sim} = R_{WRR/AF}^{*,sim} = R_{WRR/AF}^{*,th}.$$

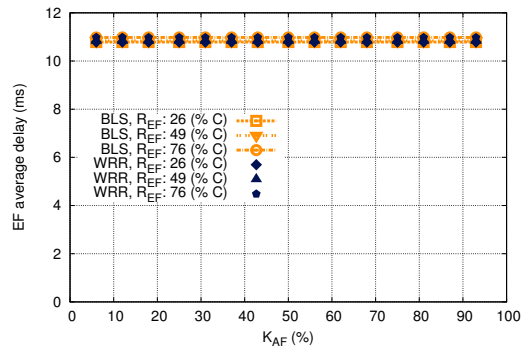
3) Performance of DE flows: these flows share the remaining capacity allocated by the WRR:

$$R_{WRR/DE}^{*,th} = (1-K_{AF}) \cdot (C - R_{EF}), \tag{6}$$

with K_{AF} already defined in (2). In Fig. 6, we can see the expected behavior again confirmed.



(a) EF output rate



(b) EF average delay

Fig. 4. EF output rate and average delay

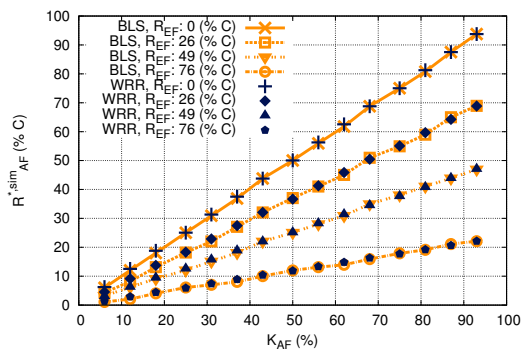


Fig. 5. AF output rate

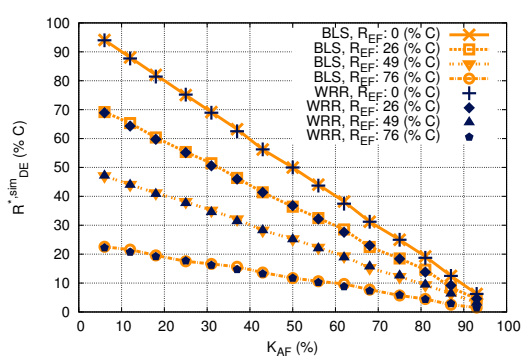


Fig. 6. DE output rate

This part shows that with a simple translation of the WRR parameters and when EF traffic is known, the BLS gets similar performance for both the EF and AF traffics. We now have to assess whether the BLS achieves a better isolation of the AF flows than the WRR when EF traffic is unknown.

C. Benefit of using BLS when EF traffic is not known

In the previous section, we showed by setting BLS parameters following (3), that BLS and WRR get a similar behavior when EF traffic is known. We now consider that the BLS parameters are set for a certain amount of expected EF traffic denoted R_{EF}^{exp}. So, $BW = B \cdot (C - R_{EF}^{exp})$, and L_M can be computed with (5). Thus, theoretical BLS output rate for the AF class is:

$$R_{BLS/AF}^{*,th} = \min\left[K_{AF} \cdot (C - R_{EF}^{exp}), C - R_{EF}\right] \quad (7)$$

The min is done between the residual capacity and the theoretical AF output rate as given by (1). This is linked to the network stability condition, i.e., the total input traffic rate has to be lower than the transmission capacity of the scheduler, of the AF traffic.

Concerning DE traffic, it uses the remaining capacity left by the EF and AF traffics. With the necessary stability condition due to AF priority being lower than EF, this gives the theoretical BLS output rate for DE class is:

$$\begin{aligned} R_{BLS/DE}^{*,th} &= C - R_{EF} - R_{BLS/AF}^{*,th} \quad (8) \\ &= \max\left[C - R_{EF} - K_{AF} \cdot (C - R_{EF}^{exp}), 0\right] \\ &= \max[(1 - K_{AF}) \cdot (C - R_{EF}) \\ &\quad + K_{AF} \cdot (R_{EF}^{exp} - R_{EF}), 0] \\ &= \max[R_{WRR/DE}^{*,th} + K_{AF} \cdot (R_{EF}^{exp} - R_{EF}), 0] \end{aligned}$$

We now have theoretical output rates for AF and DE with BLS in (7) and (8) resp. with WRR in (1) and (6). We will use these results to interpret the simulations.

Following a recent study published in 2015 [8], the part of the Internet real-time traffic is ranging from 35 to 65%. So we consider an expected rate of R_{EF}^{exp} at 49% of the capacity C. We vary the EF traffic to estimate the performance of the BLS. The WRR weights do not depend on the EF traffic so the results of our simulations with the WRR are identical to Fig. 5. The results in Fig. 7 confirm the theoretical output rate calculations:

$$R_{BLS/j}^{*,sim} = R_{BLS/j}^{*,th}, j \in \{AF, DE\}.$$

We observe while WRR rates are strongly impacted by the EF traffic, with the BLS, as long as the stability limit is not reached, the AF class has the same output rate when EF varies. Of course if the AF traffic is not impacted, then this means the impact is fully on the DE class, as defined in (8). This is visible in Fig. 8. The results in Fig. 7 and Fig. 8 can be linked to theory as follows:

1) when $R_{EF} = 0.76 \cdot C$, we have $R_{EF} > R_{EF}^{exp}$. The AF rate is lower with WRR than with the BLS

because WRR shares the reduced capacity (24% instead of 51%) between the AF and EF classes. The complete comparison between the theoretical output rates and the simulations is given in Appendix. In particular, it shows that $R^{*,th}_{WRR/AF} \leqslant R^{*,th}_{BLS/AF}$;

2) when $R_{EF} = 0.49 \cdot C$, we are in the case where $R_{EF} = R^{exp}_{EF}$. So the WRR and the BLS have the same output rates as they were designed to, when R^{exp}_{EF} was set to 49% of C;
3) when $R_{EF} = 0.26 \cdot C$, $R_{EF} < R^{exp}_{EF}$. The AF rate is higher with WRR than with the BLS because WRR shares the benefit of the additional capacity (74% instead of 51%) between AF and EF classes. The same analysis as we did for $R_{EF} = 0.76 \cdot C$ can be done in this case. It shows that $R^{*,th}_{WRR/DE} \leqslant R^{*,th}_{BLS/DE}$ and $R^{*,th}_{WRR/AF} \geqslant R^{*,th}_{BLS/AF}$.

To sum up, we have shown through simulations backed up with theory that the BLS limits the impact of UDP/EF variations on the TCP/AF traffic. This holds as long as the stability condition is fulfilled. The full impact of the increase of UDP/EF traffic is limited to the DE class flows.

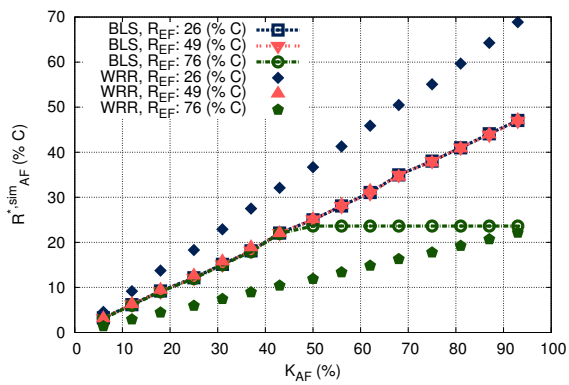


Fig. 7. AF output rate with $R^{exp}_{EF} = 0.49 \cdot C$

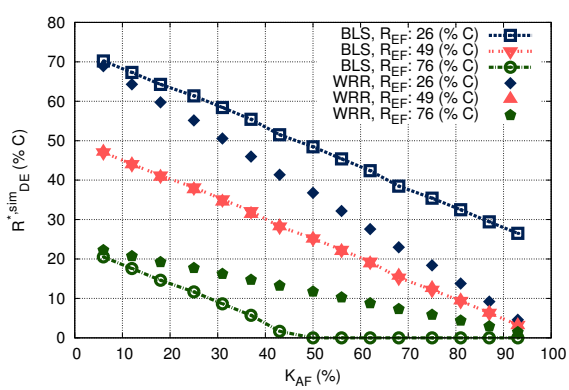


Fig. 8. DE output rate with $R^{exp}_{EF} = 0.49 \cdot C$

IV. Conclusion

We have presented in this paper the benefit of using the Burst Limiting Shaper which better quantifies the performance of the AF class of a core network scheduling. Basically, the BLS enforces the rate guarantee offered by the AF class whatever the traffic in the EF class. In particular, we have shown that the new service offered to the AF class is defined by the relation (7). Compared to the WRR, the AF output rate is less dependent on the EF traffic, which improves the quantification of the reserved capacity of AF, without impacting EF traffic. Finally, we have shown that a correspondence between WRR weights and BLS parameters can be achieved, making this proposal simple to deploy by network engineers. We wish to highlight that we did some experiments over a multihop network that confirm the good properties of BLS. We now implement this shaper inside the Linux Traffic Control API to test in real condition this proposal and expect to present these results to the AQM IETF working group.

References

[1] J. Babiarz, K. Chan, and F. Baker. Configuration Guidelines for DiffServ Service Classes. RFC 4594 (Informational), August 2006. Updated by RFC 5865.

[2] F. Baker, J. Polk, and M. Dolly. A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic. RFC 5865 (Proposed Standard), May 2010.

[3] B. Bensaou, K. T. Chan, and D. H. K. Tsang. Credit-based fair queueing (cbfq): a simple and feasible scheduling algorithm for packet networks. In *IEEE ATM Workshop 1997. Proceedings*, pages 589–594, May 1997.

[4] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. RFC 2475 (Informational), December 1998. Updated by RFC 3260.

[5] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. RFC 3135 (Informational), June 2001.

[6] Franz-Josef Gotz. Traffic Shaper for Control Data Traffic (CDT). *IEEE 802 AVB Meeting*, July 2012.

[7] Emmanuel Lochin and Pascal Anelli. TCP throughput guarantee in the DiffServ Assured Forwarding service: what about the results ? *Annals of Telecommunications*, vol. 6(3-4), April 2009.

[8] Sandvine. Global Internet Phenomena Report. Technical report, 2015.

[9] Pentti A. Leppnen Sastri L. Kota, Kaveh Pahlavan. In *Broadband Satellite Communications for Internet Access*. Springer-Verlag Berlin Heidelberg, Berlin, Allemagne, 2004.

[10] Chuck Semeria. Supporting Differentiated Service Classes: Queue Scheduling Disciplines, 2001. White Paper 200020-001 12/01 - Juniper Networks, Inc.

[11] Daniel Thiele and Rolf Ernst. Formal worst-case timing analysis of Ethernet TSN's burst-limiting shaper. In *Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016.

[12] Michel Diaz Jos Enrquez-Gabeiras Torsten Braun, Thomas Staub. In *End-to-End Quality of Service Over Heterogeneous Networks End-to-End Quality of Service Over Heterogeneous Networks*. Springer-Verlag Berlin Heidelberg, Berlin, Allemagne, 2008.

Appendix

When $R_{EF} = 0.76 \cdot C$, we have $R_{EF} > R^{exp}_{EF} = 0.49 \cdot C$. As long as $R^{*,sim}_{BLS/AF} \leqslant C - R_{EF}$, then $R^{*,sim}_{BLS/AF}$ increases as defined by the relation $K_{AF} \cdot (C - R^{exp}_{EF})$. Then, when $K_{AF} \cdot (C - R^{exp}_{EF}) \geqslant C - R_{EF}$, the output $R^{*,sim}_{BLS/AF}$ is limited by the priority scheduler due to the EF traffic. This leaves $R^{*,sim}_{BLS/AF} = C - R_{EF}$. So $R^{*,sim}_{BLS/AF} = R^{*,th}_{BLS/AF}$. When we compare $R^{*,sim}_{BLS/AF}$ to $R^{*,sim}_{WRR/AF}$ we find in Fig. 7 that $R^{*,sim}_{WRR/AF} \leqslant R^{*,sim}_{BLS/AF}$. This fits the theoretical output rates since we have $K_{AF} \cdot (C - R_{EF}) < K_{AF} \cdot (C - R^{exp}_{EF})$ and $K_{AF} \cdot (C - R_{EF}) \leqslant C - R_{EF}$. The same analysis can be done for DE traffic, proving that $R^{*,sim}_{BLS/DE} = R^{*,th}_{BLS/DE}$ and $R^{*,th}_{WRR/DE} \geqslant R^{*,th}_{BLS/DE}$ which fits the results on Fig 8;