Using token leaky bucket with feedback control for guaranteed boundedness of buffer queue.

V. Guffens, G. Bastin, H. Mounier

October 28, 2002

***Abstract*— A fluid flow model of a FCFS queueing system is presented and extended to the so-called token leaky bucket case. A simple feedback strategy that guarantees the boundedness of packets buffer queue is then introduced. Some simulations are presented and confronted to experiments run on a network made of Linux machines.**

***Keywords*— hop-by-hop control, pushback, traffic shaping, network modeling**

1 Introduction

It is well known that the congestion avoidance features of TCP have served the Internet for years in preventing congestion collapse. However, the multiplication of TCP implementations accessing the network and an increasing use of "delay sensitive" applications are augmenting the number of flows that are not sufficiently responsive to congestion notification. This problem is discussed in RFC2309 [1] where the use of active queue management techniques such as Random Early Detection (RED) are recommended in order to maintain an average queue size sufficiently small. Nevertheless, these methods don't apply to flow that are not responsive or not responsive enough to congestion signal.

It is also of widespread acceptance that TCP is not able to control the traffic at a time scale smaller than a few round-trip-times (RTT). This time scale, although suitable to ensure the global stability of the Internet, is not sufficient to provide sufficient quality of service. The end-to-end nature of TCP doesn't seem appropriate to ensure these type of service that require some knowledge about the state of intermediary hops.

In this paper, we present a simple hop-by-hop feedback control method that is acting at the layer 3 of the OSI model which makes it immune to the "unresponsive flow" problem mentioned above. In essence, this method ensures the "conservation of packets" at a hop-by-hop level.

2 A fluid flow model of the M/M/1 queue

Consider a host computer connected to the network through a single-server queuing system with constant service rate as depicted in fig. 1

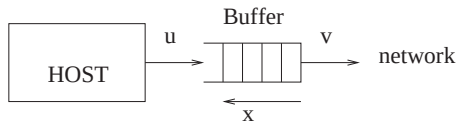

Figure 1: A single server queueing system.

The host is supposed to deliver packets to the buffer at a rate denoted $u(t)$ which can be highly bursty. The load of the buffer is denoted $x(t)$ (i.e. the number of packets in the buffer at time t). A continuous time fluid flow model of the buffer dynamics is as follows :

$$\dot{x} = -v + u \tag{1}$$

where $v(t)$ denotes the rate at which the packets are released to the network. Assuming that the buffer operates under a standard FCFS basis, the following model is proposed in [5]:

$$v(t) = \frac{\mu x(t)}{1 + x(t)} \tag{2}$$

where μ represents the constant service rate of the buffer. In the case of a constant inflow rate $\bar{u} = \lambda$, the corresponding steady state load $\bar{x}$ is given by

$$\frac{\mu \bar{x}}{1 + \bar{x}} = \lambda \quad \text{or} \quad \bar{x} = \frac{\lambda}{\mu - \lambda} \tag{3}$$

which is the classical formula of queueing theory for FCFS systems. This is a "traffic shaping" algorithm. Indeed, in case of bursts delivered by the host, the packets are temporarily stored in the buffer and released in the network at a rate μ which is constant and independent of how bursty the input is.

3 A fluid flow model of the token leaky buffer

An improvement of this algorithm is the so-called "token-leaky buffer"(TBF) which allows the output to speed up when large bursts arrive. In this algorithm, the buffer is furnished with a "token bucket" which controls the service rate of the buffer as shown in fig. 2. In this algorithm, the bucket is filled by

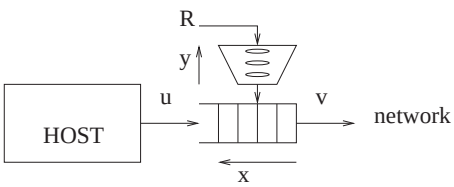

Figure 2: The token leaky buffer.

tokens at a constant rate $R > 0$, while a token is removed from the bucket each time a packet leaves the buffer. In addition, the service rate of the buffer is modulated by the level y of tokens in the bucket in such a way that $v = \mu$ when there are tokens in the bucket but $v = R < \mu$ when the bucket is nearly empty.

A continuous time fluid model of the server-bucket system is as follows:

$$\begin{aligned} \dot{x} &= -\frac{\mu x}{1+x}\frac{y}{\epsilon+y} + u \\ \dot{y} &= \begin{cases} -\frac{\mu x}{1+x}\frac{y}{\epsilon+y} + R & \text{if } 0 \leqslant y \leqslant \sigma \\ 0 & \text{if } y = \sigma \end{cases} \end{aligned} \tag{4}$$

with $v(t) = \frac{\mu x}{1+x}\frac{y}{\epsilon+y}$

In this model, the term $y/\epsilon + y$ is the modulation function mentioned above, with $0 < \epsilon \ll 1$. When $y \gg \epsilon$, it is clear that the bucket system is transparent and therefore operates as a standard single-server queueing system with service rate μ but when y is small ($y \ll \epsilon$), then the outflow rate of the buffer becomes close to R. σ is the size of the bucket which is initially full.

3.1 Burstiness Constraint

For the fluid flow model (4), we have the following positivity and boundedness property :

If $u(t) \geqslant 0\ \forall t$, $x(0) \geqslant 0$ and $0 \leqslant y(0) \leqslant \sigma$ then $0 \leqslant x(t)$ and $0 \leqslant y(t) \leqslant \sigma\ \forall t$

By integrating the second equation of the model (4), we get :

$$\int_{t_0}^{t_1} v(\tau)d\tau = y(t_0) - y(t_1) + R(t_1 - t_0) \tag{5}$$

which implies the following inequality :

$$\int_{t_0}^{t_1} v(\tau)\ d\tau \leqslant \sigma + R(t_1 - t_0) \qquad \forall t_0, t_1 | t_1 > t_0 \tag{6}$$

This inequality is well known and is discussed for instance in [3] and [2]. If R is a time varying function $R(t)$, the description given above is still valid. The burstiness constraint(6) is generalized as

$$\int_{t_0}^{t_1} v(t)\ dt \leqslant \sigma + \int_{t_0}^{t_1} R(t)\ dt \qquad \forall t_0, t_1 | t_1 > t_0 \tag{7}$$

This extension of the token leaky bucket is the core of the feedback strategy that is presented later in this paper as $R(t)$ is used as control variable.

4 A token leaky buffer with feedback

Let us now consider the interconnection of two buffers as shown in fig. 3. These buffers belong to two neighboring routers in a network. The first buffer is equipped with a token bucket as presented above. The second buffer is just a standard FCFS buffer. The point of interest here is the introduction of the feedback strategy: indeed, we can see that the token bucket is no longer fed at a constant rate R but rather at the rate at which its neighbor is sending its traffic. The set of equations corresponding to this system is :

$$\begin{cases} v_1 = \phi(y_1)\psi(x_1)\mu_1 \\ \dot{y}_1 = v_2 - v_1 \\ \dot{x}_1 = u_1 - v_1 \\ \\ v_2 = \psi(x_2)\mu_2 \\ \dot{x}_2 = v_1 - v_2 \end{cases} \tag{8}$$

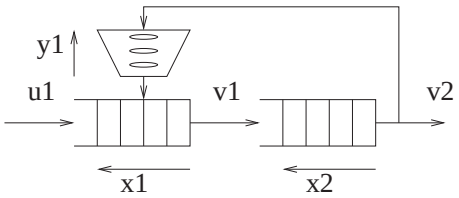

Figure 3: Interconnection of two buffers with feedback.

where $\phi(y) = y/\epsilon + y$ and $\psi(x) = x/1 + x$

4.1 Property

If the fluid flow model is initialized as follows :

$$x_1(0) = 0 \qquad x_2(0) = 0 \qquad y_1(0) = \sigma_1 > 0$$

And if the inflow rate u_1 is non-negative :$u_1(t) \geqslant 0$ for all t then

$$\begin{array}{ll} \text{a)} & x_1(t) \geqslant 0 \quad x_2(t) \geqslant 0 \quad y_1(t) \geqslant 0 \quad \forall t \\ \text{b)} & y_1(t) + x_2(t) = \sigma \quad \forall t \\ \text{c)} & y_1(t) \leqslant \sigma_1 \quad x_2(t) \leqslant \sigma_1 \quad \forall t \end{array}$$

From this property , we observe that the presence of the feedback loop guarantees that the buffer queue x_2 is naturally bounded by the size of the token bucket and therefore that the transmission is operated without packet loss. This is, at a hop-by-hop level, very similar to the principle of "conservation of packets" or "conservative flow" discussed in this famous paper from Jacobson [6].

4.2 Burstiness constraint

From the fluid flow model (8), the following inequality can be derived :

$$\int_{t_0}^{t_1} v_1 \; dt \leqslant \sigma_1 + \int_{t_0}^{t_1} v_2 \; dt \qquad \forall t_0, t_1 | t_1 > t_0 \qquad (9)$$

This inequality can be interpreted as a flow constraint, the left buffer is shaping its output according to the output of its neighbor. It will send at most a burst of σ_1 packets and will then send its traffic at a rate that can be sustained by its neighbor.

5 Interconnection with delay

Let's now consider again the system depicted in Fig. 3 with the addition of a transport delay τ, both in the link in between the two buffers and in the feedback loop. Although the addition of a delay doesn't break the boundedness property of the buffer queue length, it may intuitively be thought that a long delay will eventually cause the token bucket to empty itself before the feedback information is received. This situation can be analyzed by considering the flow of packets around the bucket and around the second buffer :

$$\left\{ \begin{array}{lcl} \dot{y}_1(t) & = & v_2(t-\tau) - v_1(t) \\ \dot{x}_2(t) & = & v_1(t-\tau) - v_2(t) \end{array} \right.$$

By time shifting the second equation and replacing the expression obtained for v_2 in the first one, it comes after integrating from 0 to t.

$$y_1(t) - \sigma = \int_0^t v_1(\xi - 2\tau) - v_1(\xi) \; d\xi - x_2(t-\tau)$$

If $v_1(t) = 0 \; \forall t \leqslant 0$,

$$y_1(t) - \sigma = -\int_{t-2\tau}^t v_1(\xi) \; d\xi - x_2(t-\tau)$$

Therefore, as $y_1(t) \geqslant 0 \; \forall t$, the following inequality is also true :

$$\frac{1}{2\tau} \int_{t-2\tau}^t v_1(\xi) \; d\xi \leqslant \frac{\sigma - x_2(t-\tau)}{2\tau} \qquad (10)$$

The presence of a propagation delay limits the maximum achievable average throughput of the system. This problem is typical of systems with high bandwidth-delay product and is discussed, for instance in [7].

6 Implementation of the feedback loop

In practice, the feedback loop cannot be implemented at a per packet basis as it would generates too much overhead traffic. Instead, the number of outgoing packets are counted and this information is sent back at regular intervals, Δ, to the neighbor who originated these packets. As in the previous section, this modification doesn't break any of the properties discussed so far but puts some limits on the maximum

average throughput of the system. The following equation can be written for $\dot{y}_1$:

$$\dot{y}_1(t) = \sum_{k=1}^{\infty} \int_{(k-1)\Delta}^{k\Delta} v_2(\xi)\ d\xi\ \delta(t-k\Delta) - v_1(t)$$

After integration, it comes :

$$\begin{aligned} y_1(t) - \sigma &= \textstyle\sum_{k=1}^{\infty} \displaystyle\int_{(k-1)\Delta}^{k\Delta} v_2(\xi)\ d\xi\ 1_+(t-k\Delta) \\ &\quad -x_2(t) - \int_0^t v_2(\xi)\ d\xi \end{aligned}$$

And finally, as $y_1(t) \geqslant 0\ \forall\ t$,

$$\frac{1}{\Delta}\int_{(k-1)\Delta}^{k\Delta} v_2(\xi)\ d\xi \leqslant \frac{\sigma - x_2(t)}{\Delta} \qquad (11)$$

$$(k-1)\Delta < t < k\Delta,\ k = 1, \ldots, \infty$$

7 Simulations and experimental results

In this section, fluid flow simulations of the TBF and the TBF with feedback (TBFFB) are presented and confronted to experimental results. The experimental setup is realized with User Mode Linux (UML)[4], a user mode port of the Linux kernel into itself. With the help of a backend switch daemon, virtual machines are connected together to form a virtual network. The network sniffer "tcpdump"[8] is used to collect network traces.

7.1 The simple token leaky bucket

The parameters used for the TBF are fixed as follows :

μ	=	50	[pps]	(packets per seconds)
R	=	25	[pps]	(packets per seconds)
σ	=	10	[p]	(packets)

The input flow $u(t)$ is set to observe the shaping action of the TBF, i.e some value greater than R, in this case $u(t) = 50$ [pps].

The actual traffic is made of ICMP echo packets of 1024 bytes sent every 0.02 [s], yielding a rate of 50 [pps]. The implementation of the TBF in Linux uses the bytes count, but the packets being of equal size, we can present the result here in packets per second for ease of comparison with the fluid flow model.

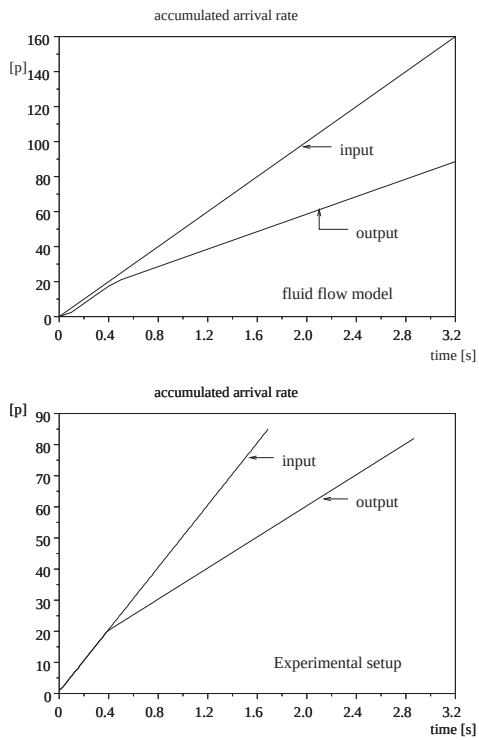

Figure 4: Fluid flow simulation and experimental result showing the typical shaping action of a token bucket filter.The dashed line is the output of TBF. The two graphics are identical.

The presented curves are cumulated flow, defined as the number of packets seen on the time interval $[0, t]$.

The result are shown in Fig. 4 where it can be seen that the model and the experiment give very similar results. After a time τ that can be approximated by the following theoretical formula,

$$\tau = \frac{\sigma}{\mu - R} = 0,4[s]$$

the token bucket is empty and the output rate $v(t)$ is limited to 25 [pps]. The input rate $u(t)$ remains constant. Packets accumulate in the output interface queue and are finally dropped when the queue is full.

7.2 The token leaky buffer with feedback

Fig. 5 shows the experimental setup used to illustrated the properties of the token bucket with feedback. The source is configured with a token leaky bucket with feedback while the router is configured

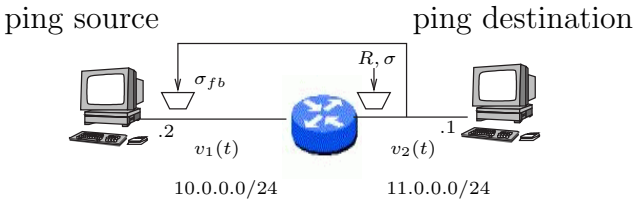

Figure 5: Experimental setup

with a classical token leaky bucket with the following parameters :

$$\begin{array}{rcll} R & = & 25 & \text{[pps]} \quad \text{(packets per seconds)} \\ \sigma & = & 10 & \text{[p]} \quad \text{(packets)} \end{array}$$

In order to implement the feedback loop, A new protocol has been registered in the Linux kernel which is used to transmit the feedback information. As can be seen in Fig. 6 (see [8] for details about the trace format), the payload of this protocol is a single 4 bytes field holding a long integer carrying the following value (see Section 6) :

$$\int_{(k-1)\Delta}^{k\Delta} r(\xi)\ d\xi$$

The remaining part of the packet is filled with random data to reach the minimal Ethernet frame size.

```
0.0 fe:fd:a:0:0:1 fe:fd:a:0:0:2 0888 78:

0000 0005 0000 0000 0000 0000 0000 7cee
8ea0 7cee 8ea0 0000 0000 0000 0000 009c
8fa0 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000
```

Figure 6: Sniffer trace showing a packet used in our experiment to carry the feedback information (five tokens are sent back).

Such a packet is sent every $\Delta = 200$[ms] in an Ethernet frame with type $0x0888$. The size of the feedback bucket is set to $\sigma_{fb} = 15$.

The router is in charge of updating this value as it forwards traffic on the network. Upon reception of such a frame, the source extracts the long integer and adds its value to the amount of tokens still present in its bucket. Apart from this modification, the behavior of the token leaky bucket is left unchanged.

The source tries to emit a constant ICMP stream at a rate of 50 [pps].

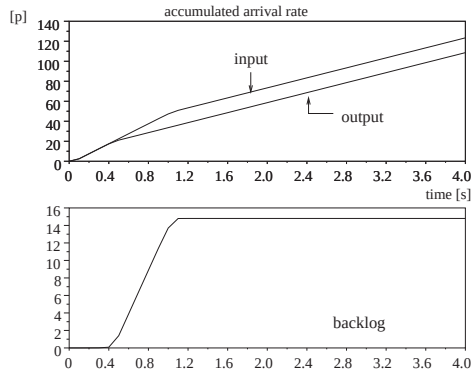

Figure 7: Fluid flow simulation showing at the top, the accumulated flow $V_1(t)$ and $V_2(t)$, at the bottom, the buffer length $x_2(t)$, clearly bounded by σ_{fb}

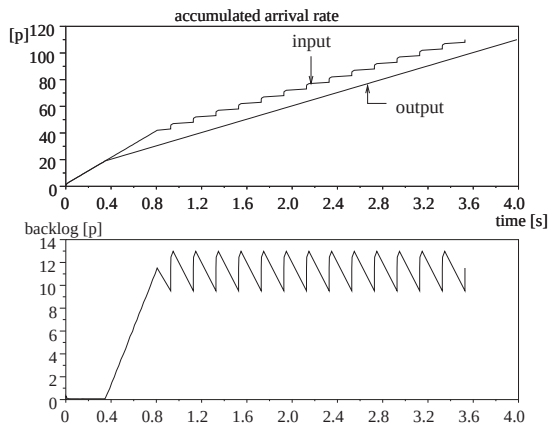

Figure 8: Experimental result showing the accumulated flow and the vertical deviation between the two curves, bounded by σ_{fb}

As in the previous experiment, the router will limits its output rate $v_2(t)$ to 25 [pps] after 0.4 [s] but in contrast with the case without feedback, the source is now adapting its sending rate to this new network condition. In effect, it can be verified in fig. 7 and 8 that the input flow v_1 is shaped so as to track the output v_2 as expressed by the inequality (9), ensuring the boundeness of the buffer queue length to $\sigma_{fb} = 15$.

7.3 A more complex setup

The hop-by-hop feedback strategy is now confronted to a more realistic topology made of five nodes, 2 sources and one destination, shown in Fig. 9.

Source 1 is trying to emit at a constant rate of 50 [pps] during the time interval [0,14]. Source 2 will

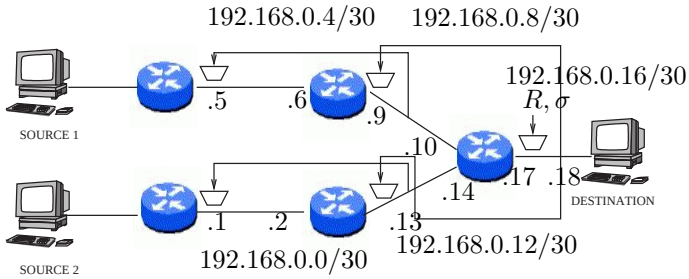

Figure 9: Setup used to test the feedback buffer on a more complex topology

emit two small constant bursts of 10 [pps] during the two time intervals [8,10] and [18,24]. The bottleneck is realized with a classical token leaky bucket placed on .17 which is configured with the following parameters :

$$\begin{array}{lcll} \sigma_{tb} & = & 10 & [\text{p}] \\ R_{sust} & = & 25 & [\text{pps}] \\ \text{burst_rate} & = & 50 & [\text{pps}] \end{array}$$

While this bucket is not empty, the burst rate is limited to 50 [pps]. The time needed to empty this bucket is therefore about 0.5 [s]. The parameters used to implement the token leaky buffer with feedback are :

$$\begin{array}{lcll} \sigma_{fb} & = & 15 & [\text{p}] \\ \Delta & = & 0.2 & [\text{s}] \end{array}$$

The results are displayed in Fig. 10 where the effect of the bottleneck can be readily seen on probe .18. After half a second, the throughput of the system is limited to 25 [pps] which causes the buckets placed on .13 and later, .1 to empty themselves and limit the throughput of their own link to 25 [pps] (See probe .2 and .14).

At time t=4 [s], Source 2 starts to transmit at a constant rate of 10 [pps](See probe .6). The TBFFB placed on .17 is now sharing its tokens between .9 and .13. The throughput of source 1 is automatically reduced to a value close to 15 [pps].

8 Conclusion

A fluid flow model that accurately describes the behavior of typical queueing systems such as the token leaky buffer has been presented. It has been shown that it is possible to derive mathematical properties from this model. In particular, a feedback control scheme that guarantees the boundedness of buffer queue length has been analyzed. Finally, the fluid flow model has been confronted to a real implementation and the properties of the token bucket with feedback have been validated.

References

[1] B. Davie S. Deering D. Estrin S. Floyd V. Jacobson G. Minshall C. Partridge L. Peterson K. Ramakrishnan S. Shenker J. Wroclawski L. Zhang B. Braden D. ClarkJ. Crowcroft. *RFC2309 - Recommendations on Queue Management and Congestion Avoidance in the Internet*, April 1988.

[2] C. Chang. Stability, queue length, and delay of deterministic and stochastic queueing networks. *IEEE Transactions on Automatic Control*, 39:913–931, May 1994.

[3] Rene L. Cruz. A calculus for network delay, part i: Network element in isolation. *IEEE transactions on information Theory, vol.37, NO. 1*, January 1991.

[4] Jeff Dike. User mode linux kernel home page. http://user-mode-linux.sourceforge.net/.

[5] G. Bastin H. Mounier. Compartmental modelling for traffic control in communication networks. *submitted to IEEE transactions on automatic control*, 2002.

[6] V. Jacobson. Congestion avoidance and control. *ACM Computer Communication Review; Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988*, 18, 4:314–329, 1988.

[7] Kleinrock L. The latency/bandwidth tradeoff in gigabit networks. *IEEE com. mag.*, vol 30, n 4:36–40, 1992.

[8] Craig Leres Van Jacobson and Steven McCanne. tcpdump/lipcap home page. http://www.tcpdump.org/.

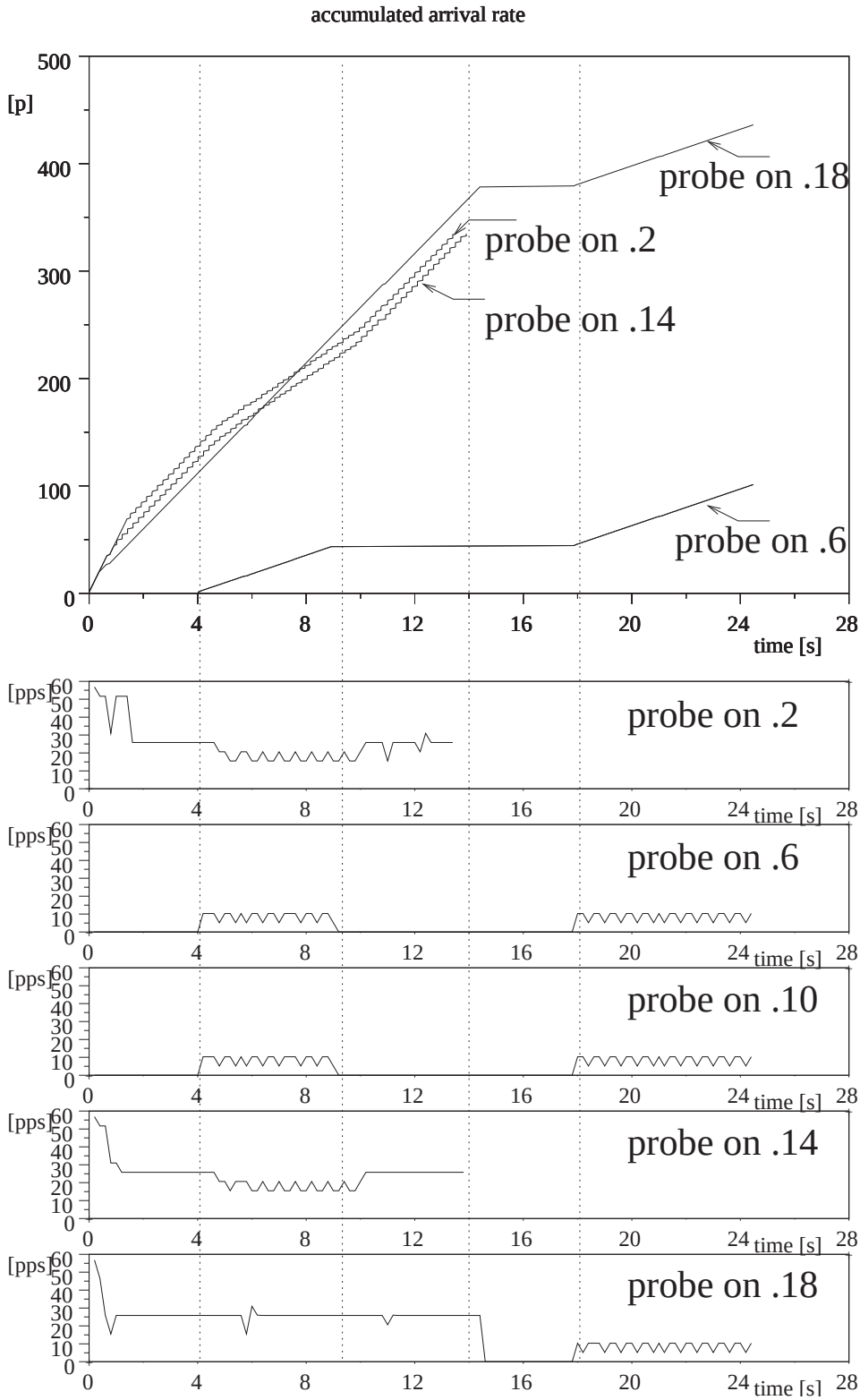

Figure 10: The rates measured by the different probes show the shaping action of the token leaky buffer with feedback that guarantees the boundedness of the buffer queues.