2

A SIMULATION STUDY OF TCP WITH THE GFR SERVICE CATEGORY

Olivier Bonaventure*

Research Unit in Networking,Université de Liège,Belgium
bonavent@montefiore.ulg.ac.be

Abstract: Recently, the Guaranteed Frame Rate (GFR) service category has been proposed to provide bandwidth guarantees with a simpler implementation than ABR in ATM networks. In this paper, we study the performance of TCP in LAN and WAN ATM networks supporting the GFR service category. We present simulations where each TCP connection is carried by one GFR VC with a minimum guaranteed bandwidth. We consider the proposed FIFO-based and WFQ-based switch implementations and evaluate their ability to efficiently support TCP traffic. Our simulations show that with the proposed FIFO-based implementation for the GFR service category, TCP is unable to benefit from the minimum guaranteed bandwidth of the underlying ATM VC. With the proposed WFQ-based implementation, the performance of TCP is good in a LAN environment when no losses occur, but it becomes degraded in a WAN environment.

1 INTRODUCTION

The ABR service category standardized by the ATM Forum [10] in 1996 is expected to be deployed in ATM networks to support data traffic in the next few years. However, due to its complexity, which imposes modifications to both the ATM adapters and the ATM switches, it may take some time before ABR is completely supported by products. Furthermore, most current applications are only connected to the ATM backbone via legacy networks such as Ethernet

*This work was partially supported by the European Commission within the ACTS AC051 OKAPI programme.

LANs. Until the widespread deployment of ABR compatible products, most ATM LANs will probably rely on the UBR service category. To fill the gap between UBR and ABR, Guérin and Heinanen have recently proposed [13] a new service category called Guaranteed Frame Rate (GFR)[1]. The primary target for the GFR service category is in ATM backbones in private (e.g. a corporate backbone that interconnects a large number of LANs) or public networks (e.g. an ATM backbone that interconnects the networks of several ISPs). However, if the GFR service category is adopted and supported by ATM switches, it might also be useful for other types of applications. The main advantage of GFR over UBR is that it allows each GFR VC to request a minimum guaranteed bandwidth [24]. In a private network, mission critical applications (e.g. mirroring of mainframes or remote vizualisation of supercomputer simulations) which require high bandwidth and some level of performance guarantee could benefit from the GFR service category. Today, a large number of applications (e.g. HTTP, NFSv3, XWindow, ftp, ...) rely on TCP, and thus the performance of TCP over the proposed GFR service category needs to be studied before the adoption of the GFR service category [2].

This paper is structured as follows. We first discuss the main characteristics of the GFR service category and the proposed FIFO-based and WFQ-based switch implementations. Then, we discuss the performance of TCP in LAN and WAN environments with these switch implementations. Finally, we present our conclusions.

2 THE GFR SERVICE CATEGORY

The main motivation behind the introduction of the GFR service category [13] was to keep the simplicity of the UBR service category (from an endsystem's point of view) which is used in most ATM LANs today. Compared with the UBR service category, the main advantage of the GFR service category is that it allows a minimum guaranteed bandwidth to be associated with each VC. Another difference is that the GFR service category explicitely requires the endsystems to utilize AAL5 and also requires the ATM switches to be aware of the AAL5-PDUs boundaries. This means that congested ATM switches should discard entire AAL5-PDUs instead of individual cells. More precisely, the traffic contract used for GFR VCs [6] is composed of four main parameters :

- Peak Cell Rate (PCR) and associated Cell Delay Variation Tolerance (τ_{PCR})
- Minimum Cell Rate (MCR) and associated Cell Delay Variation Tolerance (τ_{MCR})

[1] This service category was initially called UBR+[13], but was later renamed GFR. It should not be confused with "UBR and some packet discarding scheme" (e.g. Early Packet Discard).
[2] A draft version of this paper [2] was made available to the ATM Forum Traffic Management working group and to the ITU-T Study Group 13 (Q7/13) in June 1997 as a contribution towards the definition of the GFR service category.

- Maximum Burst Size (MBS)
- Maximum Frame Size (MFS)

The PCR has the same meaning as with the UBR service category : it is the maximum rate at which the endsystem is allowed to transmit. It can be expected that the PCR will often be set at the line rate of the ATM adapter of the endsystems. The MFS is the largest size of the AAL5-PDUs that the endsystems can send. For GFR SVCs, this parameter will be equal to the AAL5-CPCS SDU size parameter which is negotiated between the source and destination endsystems during connection setup [9].

With the GFR service category, the endsystem is allowed to transmit either CLP=0 AAL5-PDUs[3] or CLP=1 AAL5-PDUs. The CLP=1 AAL5-PDUs are considered as low priority AAL5-PDUs which should be transmitted by the network on a best-effort basis. The minimum guaranteed bandwidth is not applicable for CLP=1 AAL5-PDUs and these AAL5-PDUs should be discarded earlier than the CLP=0 AAL5-PDUs when congestion occurs.

The endsystems request a minimum guaranteed bandwidth by specifying a non-zero MCR and an associated MBS. The MCR, expressed in cells per second, corresponds to te long term average bandwidth which is reserved for the VC inside the network. It is similar to the Sustainable Cell Rate (SCR) used with the VBR service category [10], although the MCR provides a minimum guaranteed bandwidth to entire AAL5-PDUs while the SCR provides a minimum guaranteed bandwidth to individual cells. Intuitively, the meaning of the MCR is that is the endsystem transmits CLP=0 AAL5-PDUs at a rate smaller or equal to the MCR, then all these AAL5-PDUs should be correctly received by the destination. However, the GFR service category does not require the endsystems to shape their traffic and it can be expected that most users of this service category will always transmit at the negotiated PCR. In this case, each AAL5-PDU will appear as a burst of cells transmitted at the PCR. The MBS parameter of the GFR traffic contract is used to support this bursty behaviour. The MBS places an upper bound on the burstiness of the traffic to which the minimum guaranteed bandwidth applies. The value of the MBS is negotiated between the endsystems and the network, but this parameter must be always at lest equal to the MFS.

Formally, the minimum guaranteed bandwidth is specified by F-GCRA(T,f) [6] with parameters $T = 1/MCR$ and $f \geq \tau_{MCR} + (MBS - 1) * (1/MCR - 1/PCR)$. The F-GCRA (figure 1) is an adaptation of the GCRA used with the VBR service category. The F-GCRA declares entire AAL5-PDUs to be eligible or non-eligible for the minimum guaranteed bandwidth. The eligible AAL5-PDUs are those which should be delivered to the destination to fulfill

[3] A CLP=0 AAL5-PDU is an AAL5-PDU composed of CLP=0 cells. The GFR service category does not allow the endsystems to transmit AAL5-PDUs containing both CLP=0 and CLP=1 cells.

the minimum guaranteed bandwidth. While the F-GCRA is used to specify the CLP=0 AAL5-PDUs which are eligible for the minimum guaranteed bandwidth, it should be noted that the GFR service category explicitely allows the endsystems to transmit AAL5-PDUs in excess of this minimum guaranteed bandwidth. The GFR service category also expects the network to deliver this excess traffic on a best-effort basis to the destination endsystems and to "fairly" distribute the available bandwidth to the active VCs.

```
Cell Arrival at time t_a :
  First cell of an AAL5-PDU:
  if( ( t_a < TAT - f ) OR (IsCLP1(cell))
  {
  /* non-eligible cell */
  eligible=FALSE;
  }
  else
  {
  /* eligible cell */
  eligible = TRUE;
  TAT = max(t_a, TAT) + T;
  }

  Middle or last cell of an AAL5-PDU :
  if(eligible)
  {
  /* eligible cell */
  TAT = max(t_a, TAT) + T;
  }
  else
  {
  /* non-eligible cell

  }
```

Figure 1 F-GCRA(T,f)

As with other service categories (e.g. VBR), two conformance definitions have been defined for the GFR service category : GFR.1 and GFR.2. The only difference between the two conformance definitions is whether a F-GCRA is used to tag the non-eligible AAL5-PDUs at the ingress of the network or not.

With the GFR.2 conformance definition, the Usage Parameter Control (UPC) function at ingress of the network uses a F-GCRA to tag the non-eligible AAL5-PDUs. When this conformance definition is used, only the eligible AAL5-PDUs are accepted as CLP=0 AAL5-PDUs inside the network. Thus, there is a clear distinction between the eligible (CLP=0) and the non-eligible (CLP=1) AAL5-PDUs and the ATM switches may rely on this to decide whether an AAL5-PDU must be delivered to fulfill the minimum guaranteed bandwidth or not. As we will see in section 3, a simple switch implementation can be used to support the GFR.2 conformance definition.

With the GFR.1 conformance definition, the network is not allowed to modify the CLP bit of the AAL5-PDUs sent by the endsystems[4], but the endsystems are still allowed to send CLP=0 AAL5-PDUs in excess of the minimum guaranteed bandwidth (even if only a fraction of these AAL5-PDUs are actually eligible for the guaranteed minimum bandwidth). With the GFR.1 conformance definition, there is thus no "visible" distinction between an eligible and a non-eligible AAL5-PDU inside the network. Thus, to support the GFR.1 conformance definition, each ATM switch in the network must be able to de-

[4] This means that the UPC does not uses an F-GCRA with the GFR.1 conformance definition.

termine, by itself, which CLP=0 AAL5-PDUs must be transmitted to fulfill the minimum guaranteed bandwidth and which AAL5-PDUs are part of the excess traffic and thus could be discarded if congestion occurs. It can thus be expected that the simplest switch implementation which supports the GFR.1 conformance definition will be more complex than the simplest switch implementation which supports only the GFR.2 conformance definition.

The eligible AAL5-PDUs are those which must be delivered to the destination to fulfill the minimum guaranteed bandwidth. However, the GFR service category does not strictly require that the eligible AAL5-PDUs are exactly those which must be delivered to the destination to provide the minimum guaranteed bandwidth. The requirement is weaker. The GFR service category only requires the network to deliver enough entire CLP=0 AAL5-PDUs at the destination to provide the minimum guaranteed bandwidth, but it does not specify precisely which CLP=0 AAL5-PDUs must be delivered to the destination.

3 PROPOSED SWITCH IMPLEMENTATIONS

The GFR service category definition [6] [13] contains two sample implementations to support the GFR service category in ATM switches. The FIFO-based implementation can be easily implemented in ATM switches, but it only supports the GFR.2 conformance definition. The WFQ-based implementation is more complex since it requires per-VC accounting, queueing and scheduling, but it can support both the GFR.1 and the GFR.2 conformance definitions.

The FIFO-based switch implementation

The FIFO-based switch implementation proposed in [13] is an adaptation of the Partial Buffer Sharing [16] buffer acceptance algorithm which is frequently used to support VBR.2 and VBR.3 VCs in ATM switches. It only supports the GFR.2 conformance definition. The FIFO-based switch implementation is an AAL5-aware buffer acceptance algorithm which relies two buffer thresholds. These two thresholds are the LBO and the HBO threshold. The highest threshold (HBO) is identical to a classical EPD threshold [23]. The lowest threshold (LBO) is used to limit the amount of non-eligible (CLP=1) AAL5-PDUs inside the buffer. The LBO threshold is used as an EPD threshold for the CLP=1 AAL5-PDUs. When the queue occupancy of the buffer is above the LBO threshold, then the newly arriving CLP=1 AAL5-PDUs are not accepted anymore in the buffer (but the newly arriving CLP=0 AAL5-PDUs are still accepted provided that the queue occupancy is below the HBO threshold).

The WFQ-based switch implementation

This implementation combines a buffer acceptance algorithm with a per-VC scheduler. It was first proposed in [13]. It provides the bandwidth guarantees required to support the GFR service category by maintaining one logical queue for each GFR VC and by serving these queues with a WFQ-like scheduler at a rate at least equal their MCR. The utilisation of this scheduler guarantees that when active, each VC will be allocated its reserved bandwidth as well as some fairshare of the available excess bandwidth (if any). Many schedulers have been proposed in the litterature [27]. For this work, we have chosen to use Virtual Spacing [21], which is equivalent to SCFQ [11] with fixed-size packets as it is particularly suited for ATM switches. Furthermore, Virtual Spacing appears to be implementable at broadband speeds and cell sorters necessary to implement Virtual Spacing in ATM switches have already been proposed [22] [5]. The Virtual Spacing algorithm maintains one state variable for each VC (VS_i) and a global state variable (*Spacing Time*) per output buffer. A weight (r_i) is associated to each VC. For the GFR service category, this weight will be equal to the MCR of the VC. The Virtual Spacing algorithm associates a timestamp to each arriving cell as follows :

- On a cell arrival from VC i
  1. $VS_i \leftarrow \max\{\text{Spacing time}, VS_i\} + 1/r_i$
  2. timestamp the cell with the value of VS_i
- All the cells are served in increasing order of timestamp
- *Spacing time* is set to the timestamp of the last served cell

In addition to the per-VC scheduler, this implementation also relies on a buffer acceptance algorithm. This algorithm relies on a global counter for the buffer occupancy and on one counter for the occupany of each per-VC queue. It then uses two thresholds (HBO and LBO) on the buffer occupancy and one threshold for each per- VC queue (T_i). [13] proposes to set these per-VC thresholds to the MBS of the GFR traffic contract for each VC. The LBO threshold has the same role as with the FIFO-based implementation : the arriving CLP=1 AAL5-PDU are only accepted if the total buffer occupancy is below the LBO threshold. The HBO threshold is used together with the per-VC thresholds. When the buffer occupancy is between the LBO and the HBO thresholds, then all the arriving CLP=0 AAL5-PDUs are accepted. When the buffer occupancy is above the HBO threshold, then an arriving CLP=0 AAL5-PDU is accepted only if the queue occupancy of its VC is below its T_i threshold.

4 THE SIMULATION MODEL

Our simulations were performed with the STCP simulator developed by Sam Mathorpe at EPFL [18]. STCP is an event-driven simulator which has been written to study the behaviour of TCP in ATM networks. STCP provides models of queues, links, leaky buckets, background sources, switches, ... The main characteristic of STCP compared with other "TCP-over-ATM" simulators is that STCP does not contain a simplified model of TCP which is used for the simulations. Instead, STCP contains the real 4.4 BSD Lite [26] working TCP code. This TCP implementation supports all the important TCP mechanisms (slow-start and congestion avoidance, fast retransmit and fast recovery, Nagle algorithm, delayed acknowlegements, large windows and timestamp options[25], ...). Furthermore, STCP emulates the socket layer, and thus the interactions between TCP and the application using it are also taken into account. We have added the FIFO-based and WFQ-based switch implementations, as well as the F-GCRA discussed in the previous sections and some trace facilities to STCP.

We consider only one-way traffic and greedy TCP sources in this paper. Each TCP source is driven by an application which opens a TCP connection, sends 10 MBytes of data, closes the TCP connection, waits for some idle time, opens a new TCP connection, performs a new 10 MBytes transfer... These idle times are exponentially distributed with a mean duration of 0.1 second. They are used to introduce some randomness in the simulations to avoid synchronisation effects. Unless otherwise noted, the simulation results reported in this paper correspond to average values for a simulation corresponding to 250 seconds of simulated time. This is much longer than most of the simulations reported in the literature and gives us a high confidence in the simulation results.

The ATM switch is modelled as a non-blocking output buffered switch. In this switch, the only cause of congestion is the possible overflow of its output buffers. For our simulations, we used a small ATM network (figure 2). All the links in this network have a bandwidth of 155 Mbps (365566 cells/sec). For the LAN simulations, we use a delay of 10 μsec on the UNI links and a delay of 100 μsec on the NNI link. For the WAN simulations, we use a delay of 2.5 msec on the UNI links, and a delay of 10 msec on the NNI link. We consider that one of the sources (the privileged source) is more important than the other sources (called the background sources). The privileged and the background sources differ only in their respective GFR traffic contracts. All the background sources use a GFR traffic contract with a low MCR. The privileged source uses a larger MCR (up to 50% the NNI link bandwidth). The MFS is set to 200 cells for both types of sources, and the PCR is set to the line rate (365566 cells per second) for both types of sources. In all the simulations with the FIFO-based switches, the MBS was set to allow a burst of two MFS-sized AAL5-PDUs sent

at PCR to be found eligible for the minimum guaranteed bandwidth. This is in line with the initial proposal for the GFR service category [13].

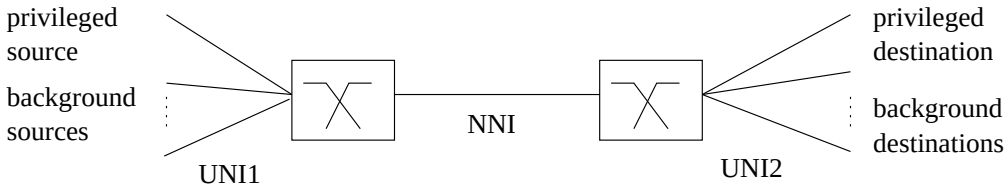

Figure 2 The simulated ATM network

5 LAN SIMULATIONS WITH FIFO-BASED SWITCHES

Instead of implementing the F-GCRA shown in figure 1, we choose to perform the tagging inside the model of the ATM adapters. This choice was motivated by the fact that the F-GCRA was not completely defined by the ATM Forum when we performed the simulations with the FIFO-based implementation, but also because the endsystem is a better place to tag the cells on a per AAL5-PDU basis than inside a UPC. The AAL5-PDU tagging was performed by the algorithm shown in figure 3 (where $T_{MCR} = 1/MCR$, $\tau_{MCR} = (2 \times MFS - 1) * (T_{MCR} - T_{PCR})$, $T_{PCR} = 1/PCR$). We configured the tagging in the ATM adapters to allow a burst of $2 \times MFS$ cells CLP=0 cells to be sent at PCR by the adapters. With MFS sized AAL5-PDUs, the tagging performed by the ATM adapters is equivalent to FGCRA$[T_{MCR}, MFS*(T_{MCR}-T_{PCR})]$, but the tagging performed by the ATM adapters may accept some additional small CLP=0 AAL5-PDUs with variable-sized AAL5-PDUs. However, both algorithms should tag the same percentage of AAL5-PDUs although they may not tag exactly the same AAL5-PDUs.

The TCP sources and destinations were configured with send and receive socket buffers (i.e. maximum window sizes) of 196608 bytes. The delayed acknowledgements were disabled on each TCP destination. The granularity of the retransmission timer was set to 0.2 seconds. These two timer granularities are lower than the default values used by 4.4 BSD Lite [26], but they correspond to the values used by commercial TCP implementations (e.g. Solaris 2.x). The fast retransmit mechanism was enabled and the retransmission threshold was set to the suggested default (3 duplicate acks). The TCP maximum segment size was set to 9140 bytes. The main TCP parameters used for this first simulation are summarised in table 2.1. Throughout this paper, we will refer to this set of parameters as $TCP_{default}$.

```
Beginning of AAL5-PDU transmission at
time t_a :

if (t_a +(AAL5_PDU_length − 1) * T_PCR >=
    max(t_a, TAT_MCR) + (AAL5_PDU_length − 1) * T_MCR − τ_MCR )
    {
        TAT_MCR = max(t_a, TAT_MCR) + AAL5_PDU_length * T_MCR
        /* send whole AAL5-PDU at PCR with CLP=0 cells */
    }
    else
    {
        /* send whole AAL5-PDU at PCR with CLP=1 cells */
    }
```

Figure 3 AAL5-PDU tagging in the ATM adapters

Table 2.1 TCP parameters for $TCP_{default}$

Parameter	Value
retransmission timer	0.2 seconds
fast retransmit threshold	3 duplicate acks

Before discussing the GFR simulations, it is interesting to first examine an artificial UBR simulation that could be considered as a baseline for the GFR simulations. For this artificial UBR simulation, we considered an ATM LAN similar to the one shown in figure 2. In this LAN, the UNI1 and NNI links used a PCR of 365566 cells per second, while the UNI2 links had a lower PCR. We used 9 background sources, and the PCR of the UNI2 links connected to the background destinations was set to 20000 cells per second, while the PCR of the UNI2 link connected to the privileged destination varied from 20000 to 180000 cells per second. The PCR of all the sources was set to the UNI1 PCR (365566 cells per second).The ATM switches had a 8192 cells buffer per output port, and the EPD threshold was set to 7168 cells. This simulation scenario is completely artificial since the sum of the bandwidth on the UNI2 links is always smaller than the bandwidth on the NNI link and thus the NNI link is not congested. We use this artificial scenario to verify whether the TCP

sources are able to "discover" and utilize efficiently the bandwidth available on the UNI2 links. The simulations performed with this artificial scenario showed that $TCP_{default}$ was able to completely utilize the bandwidth available on the UNI2 links, both for the privileged and the background sources (figure 4).

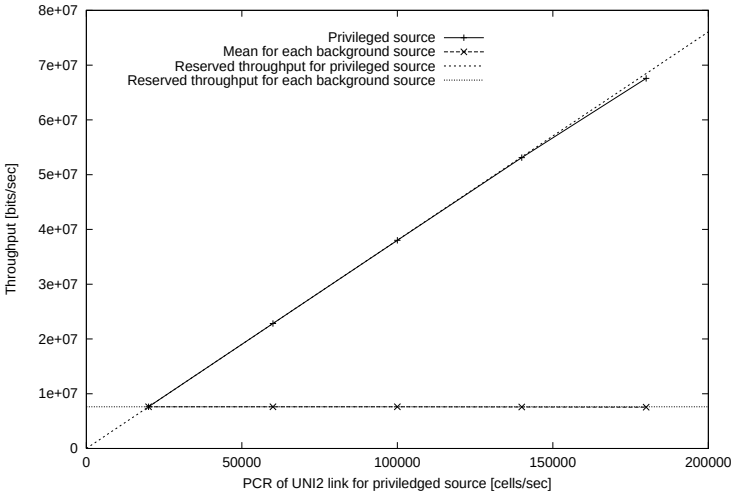

Figure 4 $TCP_{default}$ throughput with bottleneck on UNI2 links

Figure 4 shows the mean user-level throughput (i.e. TCP goodput) achieved by the privileged source as well as the mean throughput achieved by each background source with $TCP_{default}$. In addition to these simulation results, figure 4 also recalls the amount of reserved bandwidth for both the privileged and each background source. The reserved bandwidth shown in figure 4 accounts for the protocol overhead (i.e. ATM, TCP and IP headers, TCP timestamp option and AAL5 trailer), and thus can be considered as the "application-level" reserved throughput.

For our first GFR simulations, we used 9 background sources and one privileged source. Each background source had a reserved bandwidth (MCR) of 20000 cells per second. Thus, 50% of the NNI link bandwidth was reserved for the background sources. The MCR of the privileged source varied from 20000 to 180000 cells per second. The AAL5-PDU size of all the sources was set to 200 cells (i.e. slightly more than the default CPCS-PDU size for IP over ATM). The PCR of all the sources was set to the line rate (365566 cells per second). The FIFO-based ATM switches were configured with a buffer capacity of 8192 cells per output port. This output queue size corresponds to the buffer sizes

used by current commercial ATM switches. The HBO and LBO thresholds were set to 7168 and 6144 cells respectively.

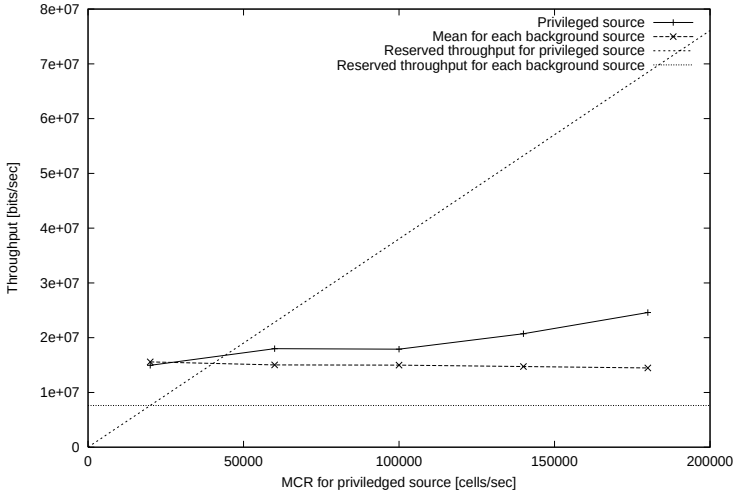

Figure 5 $TCP_{default}$ throughput for privileged and background sources

Figure 5 clearly shows that the privileged source has a lot of difficulties to actually use its reserved bandwidth. With its MCR set to 100000 cells per second the privileged source achieved only about 18 Mbps, less than 50% of its reserved bandwidth. With its MCR set to 180000 cells per second, the privileged source only achieved a throughput of about 25 Mbps, only one third of its reserved bandwidth.

A look at the simulation traces revealed two reasons for the low performance of $TCP_{default}$. The first one is the large granularity of the TCP retransmission timer with $TCP_{default}$. In a low speed network, a delay of a few hundred milliseconds is not important, but in a high speed network, it may correspond to the transmission of a few megabytes of data. During the simulations reported in figure 5, and with the MCR of the privileged source set to 180000 cells per second, the privilegded source had to retransmit about 200 KBytes during each 10 MBytes transfer, and the retransmission timer expired on average slightly less than four times during each transfer. As the minimum value of the retransmission timer is equal to the retransmission timer granularity, the average idle time may easily reach one second per 10 MBytes transfer, almost as long as the time to transfer 10 MBytes without losses at a rate of 180000 cells per second. Another factor which limits the throughput of the privileged source is

the low average value of the congestion window (figure 6) combined with the high average occupancy of the output buffer of the bottleneck switch.

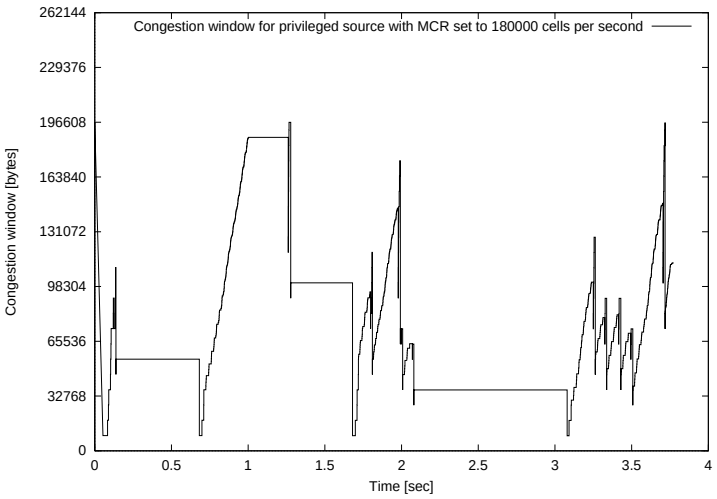

Figure 6 $TCP_{default}$: sample congestion window trace

These expirations of the retransmission timer were of course caused by packet losses in the bottleneck switch. With the FIFO-based switch implementation, CLP=1 AAL5-PDUs are discarded when the size of the output buffer of the bottleneck switch is larger than the LBO threshold, and CLP=0+1 AAL5-PDUs are discarded when the size of the output buffer of the bottleneck switch is larger than the HBO threshold. A look at the simulation traces revealed that the output buffer of the bottleneck switch did not reach the HBO threshold, and that only CLP=1 AAL5-PDUs were discarded at the bottleneck switch. With its MCR set to 100000 cells per second, slightly less than 50% of the AAL5-PDUs sent by the privileged source were tagged, while with its MCR set to 180000 cells per seconds 30% of its AAL5-PDUs were tagged. This large percentage of tagged AAL5-PDUs explains why the privileged source suffered from packet losses even though it was unable to efficiently use its reserved bandwidth.

In the following sections, we will first study whether it is possible to improve the performance of TCP by changing some of its parameters (i.e. timer granularities and retransmission mechanisms). Then, we will change the background load, the number of background sources and the LBO threshold to see

if they have an influence on the simulation results. Finally, we will look at the influence of TCP's maximum segment size.

TCP timers and retransmissions

Several TCP mechanisms and parameters may influence the TCP throughput in our environment. These mechanisms are mainly the timer-based retransmissions and the fast retransmit algorithm.

The timer-based retransmission is essential to TCP, but its negative impact on the achieved throughput may be reduced by setting its granularity to a low value. In most BSD-derived implementations, the minimum value of the retransmission timer cannot be lower than the period of the real-time hardware clock. In Unix variants, the period of this clock is typically set to 10 milliseconds. As the ATM layer guarantees the in-sequence delivery of the data, another possibility to lower the impact of the retransmission timer is to reduce the value of the fast retransmit threshold down to two duplicate acks. Throughout this paper, we will use the acronym TCP_{fast} for a TCP source which uses the parameters shown in table 2.2.

Table 2.2 TCP parameters for TCP_{fast}

Parameter	Value
retransmission timer	0.01 seconds
fast retransmit threshold	2 duplicate acks

Figure 7 shows the throughput achieved by the privileged and the background sources with TCP_{fast} and a 192 KBytes window. Surprisingly, the throughput achieved with TCP_{fast} by the privileged and the background sources in these conditions is slightly lower than the throughput achieved by $TCP_{default}$. With TCP_{fast}, the privileged and the background sources are much more agressive. During a simulation with TCP_{fast}, the total number of retransmitted packets for all the sources is roughly three times larger than with $TCP_{default}$. This explains why TCP_{fast} achieves a throughput slightly lower than $TCP_{default}$ in our LAN environment. While on average the retransmission timer expired slightly less than four times during each 10 MBytes transfer on the privileged source with $TCP_{default}$ and a 180000 cells per second MCR, it expired on average almost 10 times during each 10 MBytes transfer with TCP_{fast}. Furthermore, with TCP_{fast} the privileged source retransmitted on average about 500 KBytes per 10 MBytes transfer, while $TCP_{default}$ retrans-

mitted only about 200 KBytes. Thus, TCP_{fast} is not necessarily a better solution than $TCP_{default}$

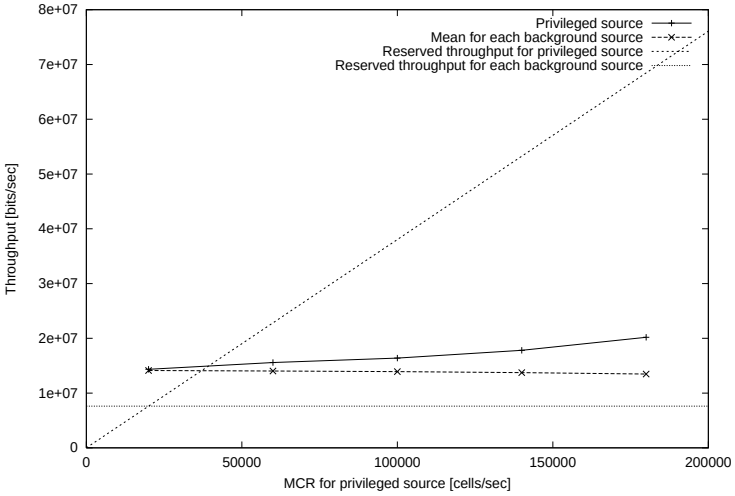

Figure 7 TCP_{fast} throughput for privileged and background sources

A third solution to improve the performance of TCP when packet losses occur would be to use the recently (re)proposed Selective Acknowledgements [19]. To evaluate the impact of this proposed TCP extension, we have patched the TCP code used in the STCP simulator with a SACK implementation [17]. This TCP_{SACK} implementation is rather conservative in its handling of retransmissions and follows [7]. Besides the use of the selective acknowledgements, TCP_{SACK} uses the same parameters as $TCP_{default}$ (table 2.3).

Table 2.3 TCP parameters for TCP_{SACK}

Parameter	Value
retransmission timer	0.2 seconds
fast retransmit threshold	3 duplicate acks
Selective Acknowledgements	Enabled

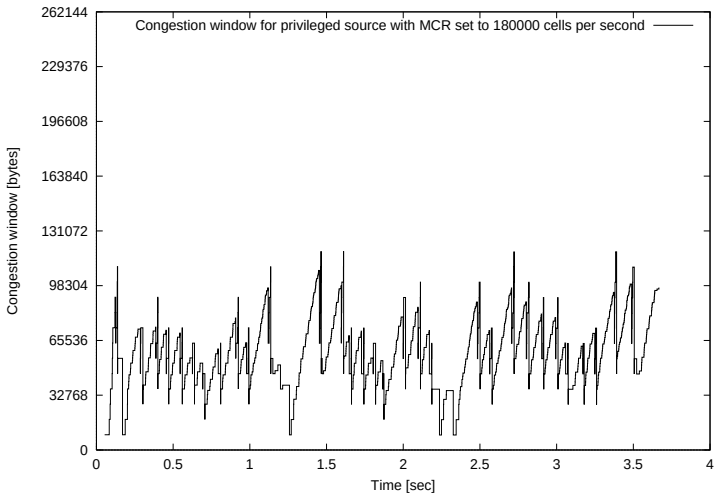

Figure 8 TCP_{fast} : sample congestion window trace

The simulations performed with TCP_{SACK} (figure 9) show that it achieves a better througput than $TCP_{default}$ and TCP_{fast}. But TCP_{SACK} is still not able to efficiently use the guaranteed minimum bandwidth of the underlying ATM VC. A look at the simulation traces with a 180000 cells per second MCR showed that the percentage of CLP=1 AAL5-PDUs with TCP_{SACK} was similar to the percentage of CLP=1 AAL5-PDUs with $TCP_{default}$, but that the number of expirations of the retransmission timer was much lower (about one expiration per 10 MBytes transfer on average).

With TCP_{SACK}, the large granularity of the retransmission timer is not the only cause for the low TCP performance. Here, the low TCP performance is due to several related factors. First (and foremost), TCP is not able to adapt is behaviour to the F-GCRA. This causes the percentage of tagged AAL5-PDUs to be much larger than what could be expected if TCP was able to transmit at exactly its fairshare. Second, the output buffer at the bottleneck switch is heavily used, and its mean occupancy is close to 5000 cells for the whole simulation. This large output buffer occupancy corresponds to a relatively large round-trip-time for the sources (for example, with a 180000 cells per second MCR, the average packet round-trip-time for the privileged source was slightly larger than 14 milliseconds). To use its reserved throughput with such a round-trip-time, the privileged source would need a window of at least 120 KBytes. Unfortunately, the large number of packet losses combined with TCP's conges-

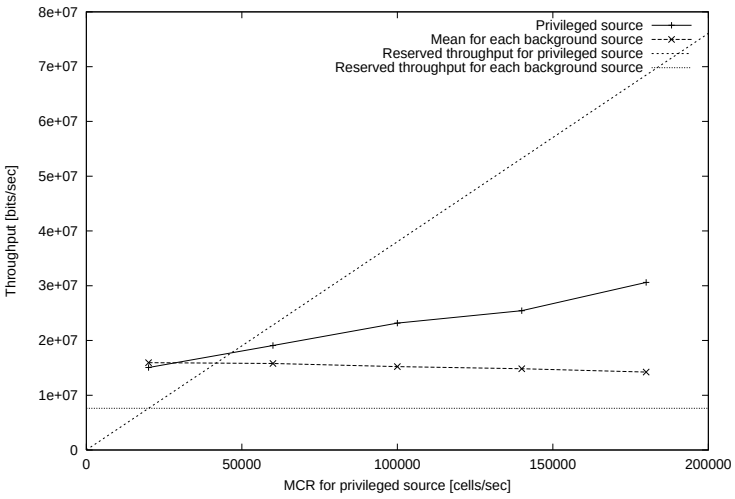

Figure 9 TCP_{SACK} throughput for privileged and background sources

tion control algorithm force the congestion window of the privileged source to be much lower than this required value on average (see figure 10 for a sample trace of the congestion window with TCP_{SACK}).

Influence of the background load

In the previous sections, we have shown the throughput achieved by the privileged source with nine background sources with an MCR set to 20000 cells per second. To verify that this particular value was not the reason for the low performance of TCP, we also performed simulations [2] with nine background sources, but with their MCR set to 10000 and 5000 cells per second. These simulations produced similar results to those discussed in the previous sections. We even performed simulations with 9 background sources with an MCR of 0 cell per second (i.e. all the AAL5-PDUs sent by these sources are tagged). In this case, the privileged source had the same difficulties to use its reserved throughput as when the background sources used a non-zero MCR.

Influence of the number of background sources

Most of the simulations reported in this paper were done with nine background sources. To verify that this particular number of background sources was not

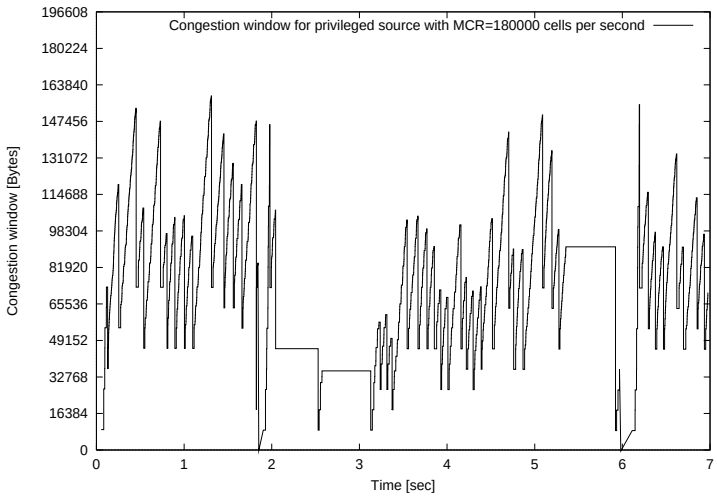

Figure 10 TCP_{SACK} : Sample congestion window trace

the cause of the low throughput achieved by the privileged source, we performed simulations with 3 and 6 background sources. In both cases, we set the sum of the MCRs of the background sources to 180000 cells per second. Figure 11 shows that even with three background sources, the privileged source still cannot use its reserved bandwidth with $TCP_{default}$. Simulations performed with TCP_{fast} and TCP_{SACK} produced similar results.

Influence of the LBO threshold

Most of the simulations shown in this paper were performed with output queues of 8192 cells and HBO and LBO thresholds of respectively 7168 and 6144 cells on the ATM switches. With these thresholds, no CLP=0 AAL5-PDUs were discarded in the bottleneck switch. Simulations performed with the LBO thresold set to 4096 and 2048 cells produced similar results as those with the LBO threshold set to 6144 cells. With TCP_{fast}, setting the LBO threshold to 2048 cells allowed the privileged source to attain a slightly higher throughput, but still less than 50% of its reserved throughput. However, this should not suggest the use of a LBO threshold set to 0 cell. In this case, all the CLP=1 AAL5-PDUs are discarded at the bottleneck switch. Simulations performed with this value of the LBO threshold show that the throughput achieved by all the sources collapses. With $TCP_{default}$ and TCP_{SACK}, each TCP source is

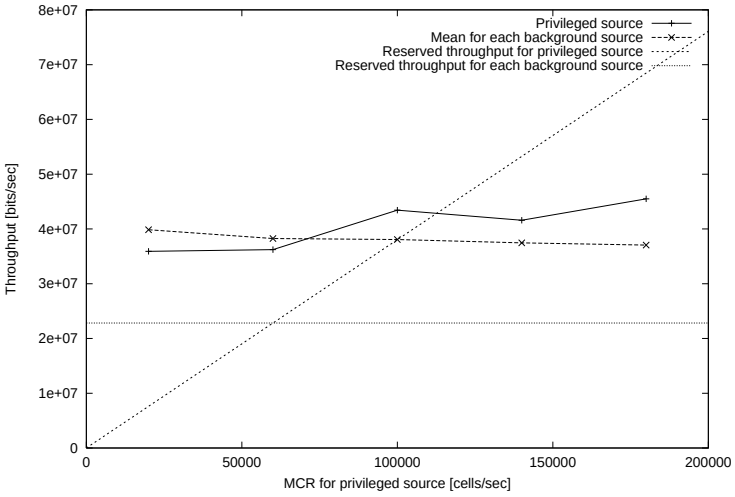

Figure 11 $TCP_{default}$: 3 background sources

only able to achieve a throughput of 275 kilobits per second with their MCR set to 20000 cells per second. A look at the segment traces revealed that TCP successfully sends three 9140 bytes segments every 800 millliseconds. With its MCR set to 180000 cells per second, the privileged source still achieves only 275 kilobits per second. With TCP_{fast}, the situation if slightly better but far from satisfactory. TCP_{fast} successfully sends three 9140 bytes segments every 40 milliseconds. This collapse of the TCP throughput is due to the fact that TCP is not able to adapt its behaviour to a traffic contract enforced by a F-GCRA. Similar problems with the VBR GCRA are discussed in [4].

On the other side, it should also be noted that using a LBO threshold larger than the sum of the window sizes used by the TCP sources is neither a solution. In this case, there are no losses in the switch buffers, and thus the AAL5-PDUs sent by all the sources are served independently of their CLP, and the throughput is shared fairly among all the competing sources, and thus the TCP throughput is independent of the MCR.

Influence of the TCP Maximum Segment Size

All the simulations presented in the previous sections were performed with a TCP Maximum Segment Size (MSS) of 9140 bytes. This value corresponds to the default value for Classical IP over ATM [1]. However, several papers

[23] [3] have shown that the TCP performance in an ATM network may also depend on the MSS. To verify that the large value of the default TCP MSS used for our simulations was not the cause of the TCP performance problems, we performed new simulations with smaller values for the TCP MSS of all the sources (but with the same GFR traffic contracts as in the previous sections).

The throughput achieved by $TCP_{default}$ with a 512 bytes MSS is slightly lower than the throughput achieved with a 9140 bytes MSS (figure 5). The throughputs achieved by $TCP_{default}$ with the MSS sizes corresponding to an Emulated Ethernet and an Emulated Token Ring are between these two values. The 512 bytes MSS corresponds to the default MSS size used by TCP implementations which do not use Path MTU Discovery when they communicate with a destination which is in a different IP subnet. The fact that the throughput achieved by $TCP_{default}$ is almost independent of the MSS is not surprising as with $TCP_{default}$ the main limitation for the throughput is the large granularity of the retransmission timer.

The throughput achieved by TCP_{SACK} with the 1460 bytes MSS (figure 12) corresponding to an Emulated Ethernet is lower than the throughput achieved with the 9140 bytes MSS (figure 9). This is not surprising, as with TCP_{SACK} one of the reasons for the low throughput for the privileged source is the time spent with a small congestion window during congestion avoidance. During congestion avoidance, the congestion window is increased by one MSS-sized segment every round trip time, and thus a lower MSS means that the increase of the congestion window is slower.

Effect of heterogeneous MSS sizes. For the simulations presented in the previous sections, all the sources used the same MSS. However, a real network may not be so homogeneous. For example, LAN Emulation supports four different maximum packet sizes. Sources which are part of different types of Emulated LANs may travel the same bottleneck link, and thus it is important to study how TCP behaves with the FIFO-based switch implementation when the privileged and the nine background sources do not use the same MSS[5].

In figures 13 and 14, we report simulations performed with heterogeneous MSS sizes with $TCP_{default}$. Figure 13 shows the throughput achieved when the privileged source uses a 9140 bytes MSS while the background sources use a 512 bytes MSS. Figure 14 shows the throughput achieved when the privileged source uses a 512 bytes MSS, while the background sources use a 9140 bytes MSS. When the privileged source uses a much larger MSS that the background sources, it acquires a large share of the bottleneck link, and its throughput is almost independent of its MCR. On the opposite, when the privileged source uses a much smaller MSS than the background sources, its throughput is very

[5]It should however be noted that all GFR traffic contracts use the same MFS (200 cells) and MBS during all the simulations

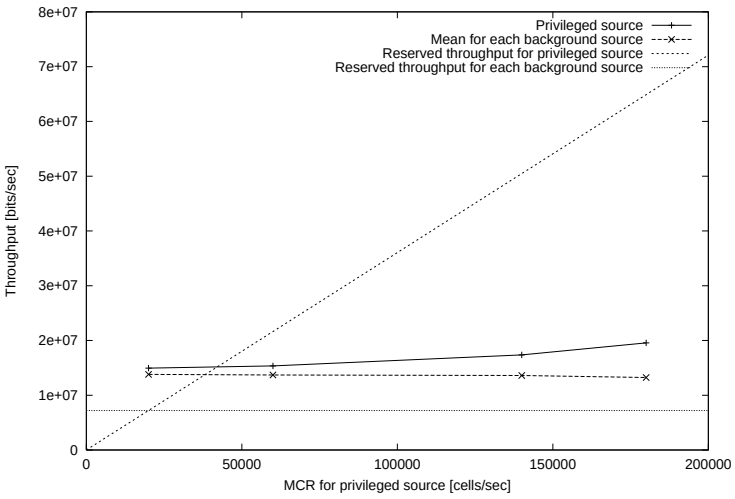

Figure 12 TCP_{SACK} : 1460 bytes MSS

low with a low MCR, and even with a 180000 cells per second MCR, it has huge difficulties to achieve a higher throuhgput than the background sources.

This unfairness for the sources which use a low MSS is again mainly due to the fact that the increase of the congestion window during congestion avoidance is proportional to the MSS size. For example, the simulations performed with an MCR of 180000 cells per second and a 512 bytes MSS for the privileged source revealed the following behaviour. Less than 1% of the segments sent by the privileged source were CLP=1 segments, and the retransmission timer expired on average twice per 10 MBytes transfer. These expirations of the retransmission timer are responsible for a fraction of the throughput drop, but the main cause for the throughput drop is the congestion avoidance mechanism. Most of the 10 MBytes transfer occured as follows. The privileged source performs slow-start, and several CLP=1 segments sent during this phase are discarded by the bottleneck switch. After the expiration of the retransmission timer, the privileged source retransmits these segments and performs congestion avoidance. Due to the small value of the MSS, the congestion window increases slowly. Unfortunately, the output buffer of the bottleneck switch is on average always more than 50% full. Such a high occupancy for the output buffer of the bottleneck switch, corresponds to an average round-trip-time of 12.6 milliseconds. Combined with such a large round-trip-time, the low average value of the congestion window for the privileged source explains its low

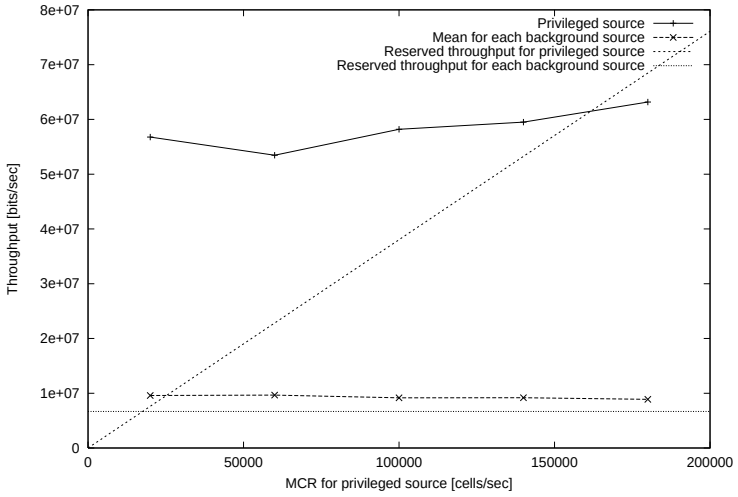

Figure 13 $TCP_{default}$ throughput with 9140 bytes MSS for privileged source and 512 bytes MSS for background sources

throughput. Similar unfairness occurs with $TCP_{default}$ and TCP_{fast} when the privileged source is part of an Emulated Token Ring and the background sources are part of an Emulated Ethernet or the reverse.

It should be noted that this unfair advantage for TCP sources with a large packet size over TCP sources with a smaller packet size also occurs with the UBR service category when the EPD buffer acceptance algorithm is used.

6 WAN SIMULATIONS WITH THE FIFO-BASED IMPLEMENTATION

From the low performance of TCP in a LAN with the proposed FIFO-based implementation, there is little hope that TCP will achieve a better performance in a wide area network. Figure 15 presents the throughput achieved by TCP_{SACK} with a 9140 bytes MSS in a wide area network (UNI1 and UNI2 links have a delay of 2.5 msec, while the NNI link has a delay of 10 msec) whose switches uses 8192 cells wide output buffers with LBO and HBO thresholds set to respectively 6144 and 7168 cells. This simulation shows clearly that TCP is not able to benefit from the GFR service category with the proposed FIFO-based switch implementation in a WAN environment.

During the simulations with TCP_{SACK}, there were few expirations of the retransmission timer, and the main reason for the low throughput of the priv-

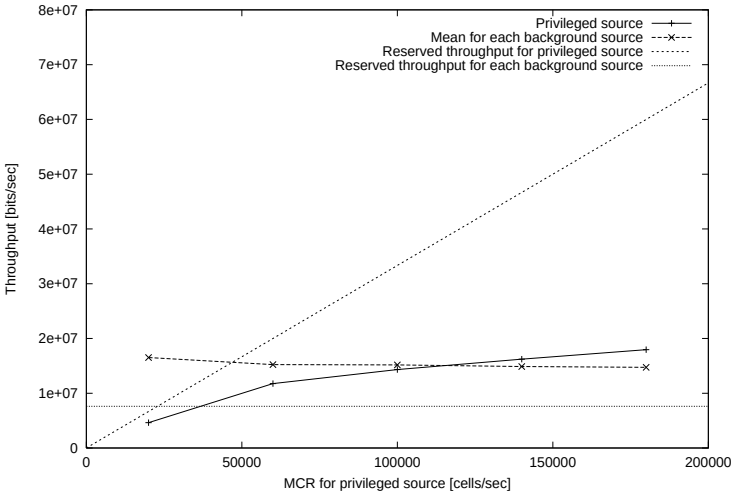

Figure 14 $TCP_{default}$ throughput with 512 bytes MSS for privileged source and 9140 bytes MSS for background sources

ileged source was the low average value of its congestion window due to the congestion avoidance mechanism.

7 LAN SIMULATIONS WITH THE WFQ-BASED IMPLEMENTATION

To validate the WFQ scheduler used in the simulator, we performed several simulations in the ATM LAN[6] of figure 2 with a small TCP window (32 KBytes) so that the 8192 cells long output buffer of the ATM switches does not overflow. We used nine background sources for these simulations, and each background source had its MCR set to 20000 cells per second. As there are no cell losses in this environment, $TCP_{default}$, TCP_{fast} and TCP_{SACK} have identical performances. As expected, the simulations performed with a 9140 bytes MSS showed that there were no cell losses, and that the available throughput was fairly shared among the background and the privileged sources (figure 16).

This simulation shows that with the proposed WFQ-based implementation in our simple LAN environment, the unreserved bandwidth is allocated in proportion to the MCR of the sources. This implies that a source with a much larger MCR than the other sources will achieve a much higher throughput, and

[6] For the simulations with the WFQ-based switch implementation discusseed in sections 7 and 8, we considered the GFR.1 conformance definition and assumed that the sources were only sending CLP=0 AAL5-PDUs. This is the most natural utilisation of the GFR.1 conformance definition with TCP/IP traffic.

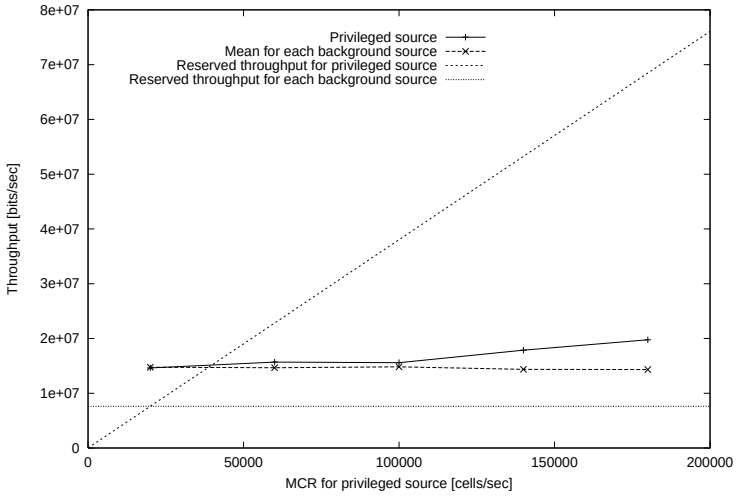

Figure 15 FIFO-based switches : TCP_{SACK} throughput in WAN

may even use almost all the unreserved bandwidth. For example, if we consider the same ATM LAN but where the MCR of each background source is set to 5000 cells per second, the simulations show that the privileged source acquires a large proportion of the NNI link (figure 17) , and this kind of fairness may not be desireable in every environment. Furthermore, the delay experienced by the background sources may become very large. For example, with the MCR of the privileged source set to 180000 cells per second, the average packet delay from the privileged source to the privileged destination is equal to 1.16 milliseconds, while the average packet delay from a background source to the corresponding background destination is 32.7 milliseconds. While the GFR service category is not expected to provide a fair treatment in terms of transmission delay [14] a shorter delay for the background sources would probably be desireable.

LAN simulations with limited buffers

The simulations discussed in the previous section showed that TCP performed well with the proposed WFQ-based GFR switch when there were no packet losses. This is an important difference with the FIFO-based implementation as when there are no losses, the FIFO-based implementation does not allow the privileged source to use its reserved bandwidth. Unfortunately, in a real network, packet losses will probably occur due to the limited size of the switch

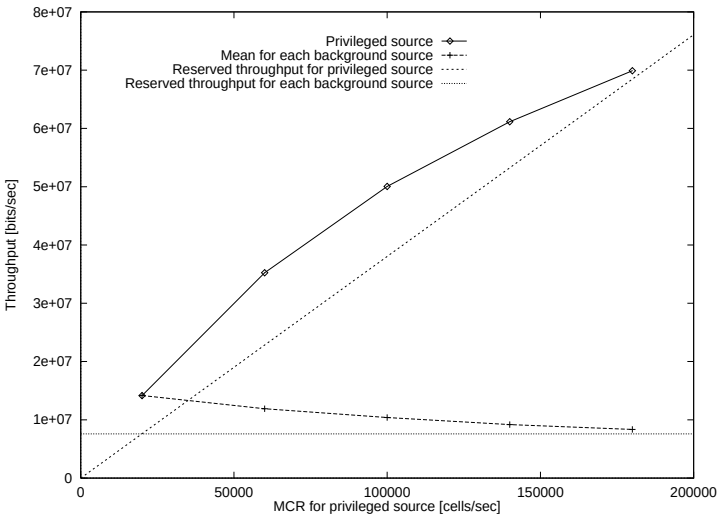

Figure 16 WFQ-based switches : TCP throughput for privileged and background sources

buffers. In our simple LAN, with a buffer of 8192 cells, packet losses will occur when the TCP sources use a 64 KBytes window.

With a 9140 bytes MSS, a 64 KBytes window and TCP_{SACK}, the privileged source had a lot of difficulties to efficiently use its reserved throughput (figure 18). However, with a 1460 bytes MSS it was almost able to achieve its reserved throughput (figure 19).

The large difference between the 1460 bytes and the 9140 bytes MSS is due to the fact that the retransmission and congestion control mechanisms are closely related in TCP_{SACK}. With a 9140 bytes MSS, a maximum window size of 65536 bytes corresponds to 7 segments. When a packet loss is detected by TCP_{SACK}, it performs congestion avoidance and thus its congestion window is reduced by a factor of two. If a new segment is lost, this loss will only be taken into account by the TCP_{SACK} sender when it has received three duplicates acknowledgements. Thus, if the number of segments lost is larger than the difference between the current value of the congestion window and the retransmit threshold (3 with TCP_{SACK}), the TCP_{SACK} sender will be forced to wait for the expiration of the retransmission timeout. The packet traces gathered during the simulations revealed that for the privileged source, 50% of the segment losses occured in groups of at least two segments, with a sequence gap of less than 65535 bytes between the lost segments. with a 9140 bytes MSS, the retransmission timer expired on average 10 times per 10 MBytes transfer,

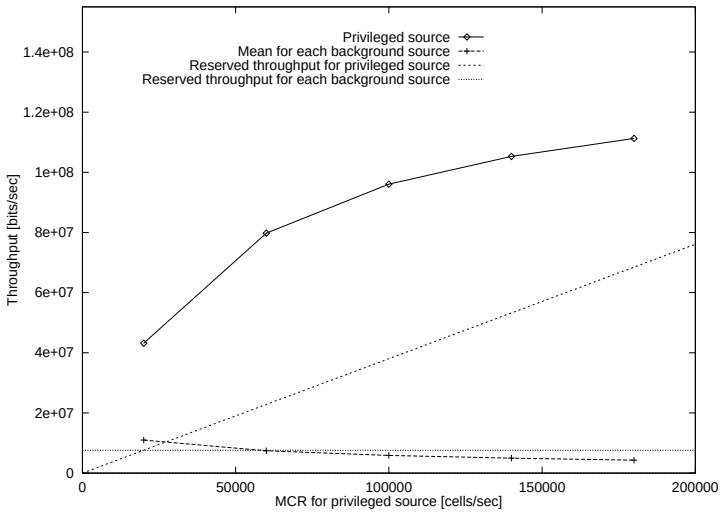

Figure 17 TCP throughput with $\sum MCR_{background}$ = 9 * 5000 cells/sec

while with the 1460 bytes MSS, it expired on average only once per 10 MBytes transfer. This explains the large difference between the 9140 and the 1460 bytes MSS.

Simulations performed with TCP_{SACK} and heterogenous MSS sizes in a LAN environment showed that the WFQ-based implementation achieved a better fairness that the proposed FIFO-based implementation. With TCP_{SACK}, there were almost no difference when the privileged source used a 512 bytes MSS while the background sources used a 9140 bytes MSS or the opposite [2]. In both cases, the privileged source was able to achieve almost its weighted fair throughput.

8 WAN SIMULATIONS WITH THE WFQ-BASED IMPLEMENTATION

To evaluate the performance of TCP in a wide area network, we performed new simulations with the same network topology as in figure 2 , but with the delay on the NNI link set to 10 milliseconds and the delay on the UNI links set to 2.5 milliseconds. Thus, without taking into account the queueing delays, the round-trip-time time in this network is 30 milliseconds. To fully utilize the link, the TCP sources should use a window size at least equal to the bandwidth delay product. The simulations performed with a slightly larger window size (550 KBytes) showed that with TCP_{SACK} the privileged source was not able

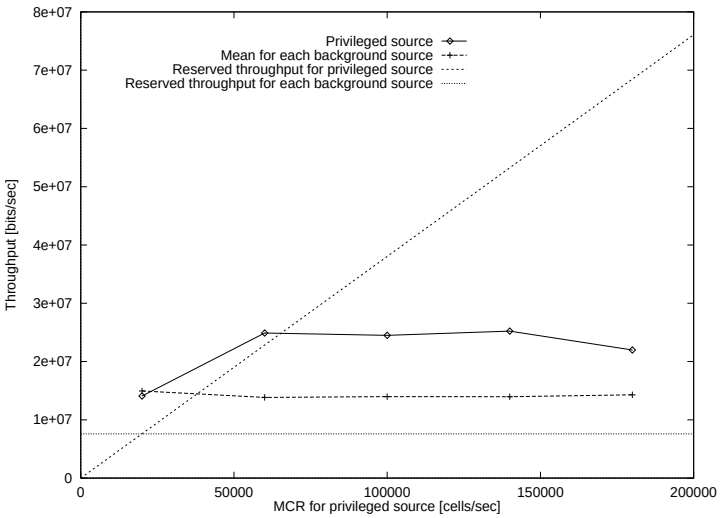

Figure 18 TCP_{SACK} with 64 KBytes window and 9140 bytes MSS in LAN

to utilize its reserved throughput (figure 20), although the throughput of the privileged source was much higher than with the FIFO-based implementation. The main reason for the low TCP throughput is again the long time spent by the privileged TCP source in congestion avoidance phase with a congestion window which is smaller than $MCR \times rtt$. Further work is needed to evaluate whether the WFQ-based implementation can be improved. A fair buffer allocation scheme such as the one proposed in [15] might improve the performance of the simple WFQ-based implementation as in a WAN, the per-VC average occupancy of the output buffer of the bottleneck switch indicates that the privileged source uses less buffer space on average than each background source.

9 IMPACT OF TAGGING ON THE TCP PERFORMANCE WITH THE WFQ-BASED SWITCH IMPLEMENTATION

In the previous sections, we have discussed the performance of TCP with the WFQ-based switch implementation used with the GFR.1 conformance definition. In this section, we evaluate the performance of TCP with the same implementation used with the GFR.2 conformance definition. Thus, we inserted an F-GCRA on the UNI links to tag the AAL5-PDUs which are not eligible for the bandwidth guarantee.

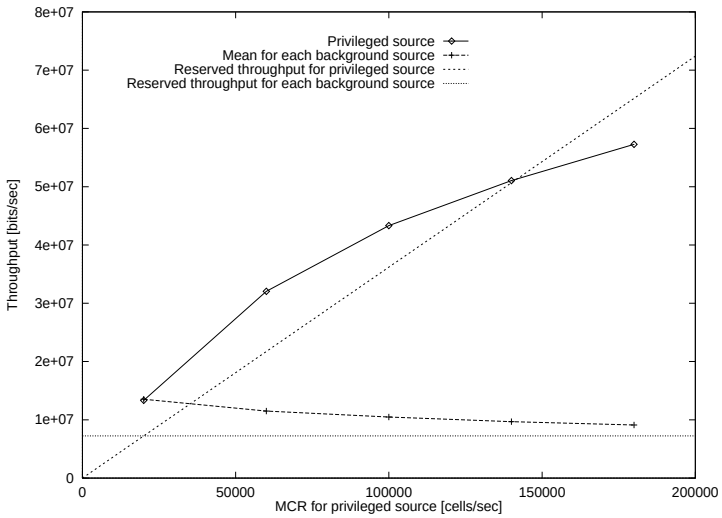

Figure 19 TCP_{SACK} with 64 KBytes window and 1460 bytes MSS in LAN

LAN Simulations

Simulations performed with 64 KBytes windows, TCP_{SACK} and homogeneous MSS sizes (9140 or 1460 bytes) for the privileged and the background sources produced similar results as those presented in section 17. Thus, in this case, the tagging performed by the FGCRA did not seem to have an influence on the throughput of the privileged and background sources.

Simulations performed with heterogeneous MSS sizes show that unfairness may occur, but the unfairness differs from the unfairness discussed with the FIFO-based implementation. Figure 21 shows the throughput achieved by TCP_{SACK} in a LAN when the MSS size of the privileged source is set to 9140 bytes, while the background sources use a 512 bytes MSS size. In this case, there is no significant unfairness. However, the simulations show that when the privileged source uses a 512 bytes MSS size, while the background sources use a 9140 bytes MSS size, a large unfairness occurs (figure 22). In this case, the lower throughput for the privileged source is mainly due to the large number of expirations of its retransmission timer. This is in contrast with the simulations performed with the proposed FIFO-based implementation (figure 14) where the number of expirations of the retransmission timer of the privileged source was very low and the low throughput was caused by the slow increase of the congestion window. With an MCR set to 20000 cells per second, the re-

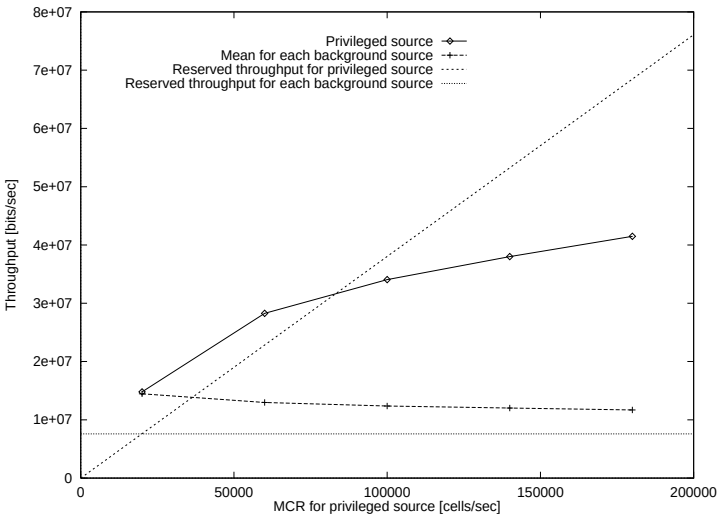

Figure 20 TCP_{SACK} throughput with 550 KBytes window in a WAN

transmission timer of the privileged source expired more than 20 times per 10 MBytes transfer during the simulations with the WFQ-based implementation reported in figure 22 while it expired only once per 10 MBytes transfer during the simulations with $TCP_{default}$ and the FIFO-based implementation reported in figure 14. Similar unfairness occured with TCP_{fast}.

WAN Simulations

In a WAN, the impact of tagging on the TCP throughput is much higher than in a LAN. The simulations performed with a 550 KBytes window size showed that with TCP_{SACK} the privileged source was not able to utilize its reserved throughput (figure 23). It should be noted that when tagging is used at the network access point, the TCP_{SACK} throughput of the privileged is much lower in a WAN than when no tagging is used.

The main reason for the low performance of TCP lies in how TCP is able to adapt its rate to a traffic contract enforced by th F-GCRA. Let us for example consider the WAN simulation with TCP_{SACK} shown in figure 23. When the MCR of the privileged source is set to 180000 cells per second, while the MCR of each background source is set to 20000 cells, the privileged source should only transmit at a rate slightly above 180000 cells per second on average as this is the rate it should receive from the WFQ scheduler on the bottleneck switch.

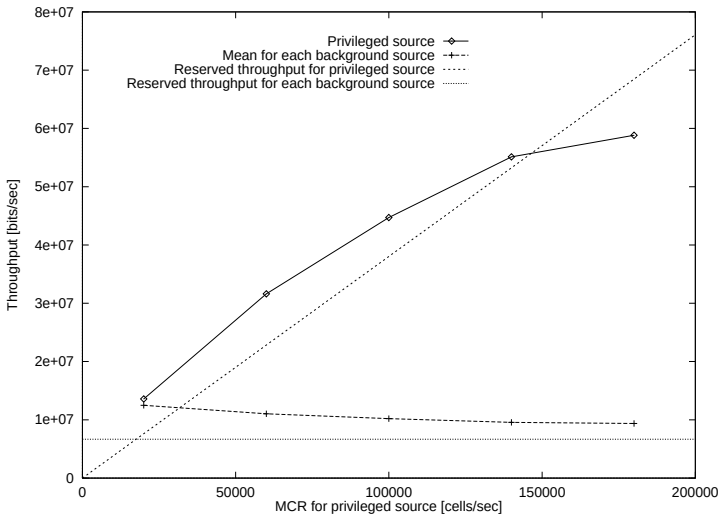

Figure 21 TCP_{SACK} throughput with 64 KBytes window and 9140 bytes MSS for privileged source and 512 bytes MSS for background sources

The simulations show that this does not happen. In fact, the measurements of the arrival rate (averaged during a period of 100 milliseconds) of the privileged source at the bottleneck switch show that the privileged source almost does not even reach its reserved throughput during a 100 milliseconds period (figure 24). However, even if on average the privileged source does not utilize its reserved bandwidth its AAL5-PDU flow is too burtsy for the F-GCRA. On average, more than one fifth of the AAL5-PDUs sent by the privileged source are tagged by the F-GCRA. This is mainly due to the fact that the congestion control scheme used by TCP forces the traffic to be bursty. During slow-start, the traffic is bursty because of the exponential increase of the congestion window. During congestion avoidance, the traffic is also bursty. When the congestion window is smaller than the bandwidth delay product, TCP sends a congestion window worth of segments, then is idle until the return of the acknowledgements.

10 RELATED WORK

A few simulation studies of TCP with the GFR service category have been discussed at the ATM Forum [20] [12]. [20] discusses the performance of TCP in a LAN environment with a FIFO scheduler, a round-robin scheduler and a weighted round robin (WRR) scheduler. Their simulations show, like our

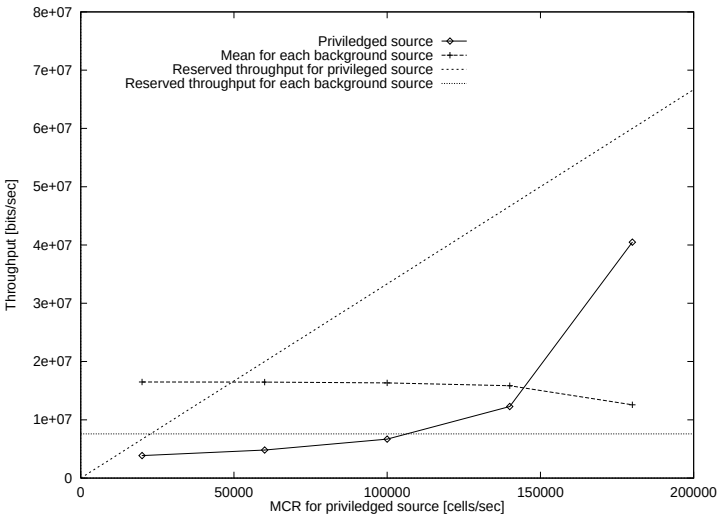

Figure 22 TCP_{SACK} throughput with 64 KBytes and 512 bytes MSS for privileged source and 9140 bytes MSS for background sources

simulations, that the FIFO scheduler and the round-robin scheduler are not sufficient, even when combined with a F-GCRA to support the GFR service guarantees. They note that with such schedulers, "*the way to ensure a minimum rate guarantee is with a combination of a large packet size and LBO close to zero*". Our simulations disagree with this conclusion. This difference between our simulations and those discussed in [20] is due to the fact that they use a 100 μsec granularity for the TCP retransmission timer in their simulation model. We do not consider such a low granularity to be a good model of current TCP implementations. [12] proposes another implementation for the GFR service category in FIFO switches and studies briefly the performance of TCP with this switch implementation.

11 CONCLUSION

In this paper, we have studied the performance of TCP when each TCP connection is carried by one ATM VC with a minimum guaranteed bandwidth provided by the GFR service category. We have shown that TCP has several difficulties to utilize the minimum guaranteed bandwidth of the underlying VC. We have discussed the performance of three variants of TCP with the proposed

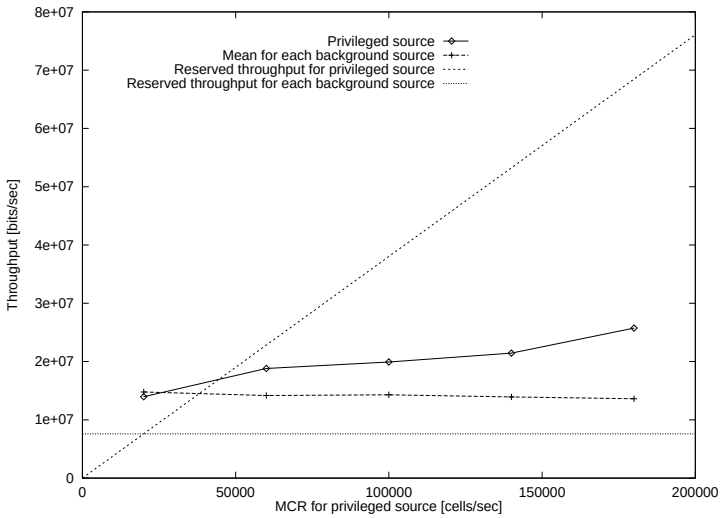

Figure 23 TCP_{SACK} throughput with 550 KBytes window in WAN

FIFO-based and WFQ-based switch implementations and shown their respective limitations.

With the proposed FIFO-based switch implementation and the GFR.2 conformance definition, the performance of TCP was never satisfactory. With infinite buffers in the switches, the TCP throughput is almost independent of the MCR of the underlying VC. This is not desireable since a VC with a large MCR should achieve a higher throughput than a VC with a much lower MCR. When the switch buffers were smaller, the performance of TCP was still not satisfactory. In this case, the TCP sources with a large MCR could not efficiently utilize their minimum guaranteed bandwidth. The low performance of TCP with the FIFO-based switch implementationwas caused by two main factors. First, the TCP traffic is bursty. Due to this burstiness, the F-GCRA used in the UPC at the ingress of the network tag a large fraction of the AAL5-PDUs, even when the long term average throughput of the VC is smaller than the MCR. Second, the FIFO-based switch implementation serves the tagged AAL5-PDUs as best-effort independently of the MCR of their VCs.

With the proposed WFQ-based switch implementation and the GFR.1 conformance definition, the performance of TCP was much better than with the proposed FIFO-based implementation and the GFR.2 conformance definition. When no losses occured (e.g. in a LAN environment with small TCP windows), the proposed WFQ-based implementation allowed each TCP source to

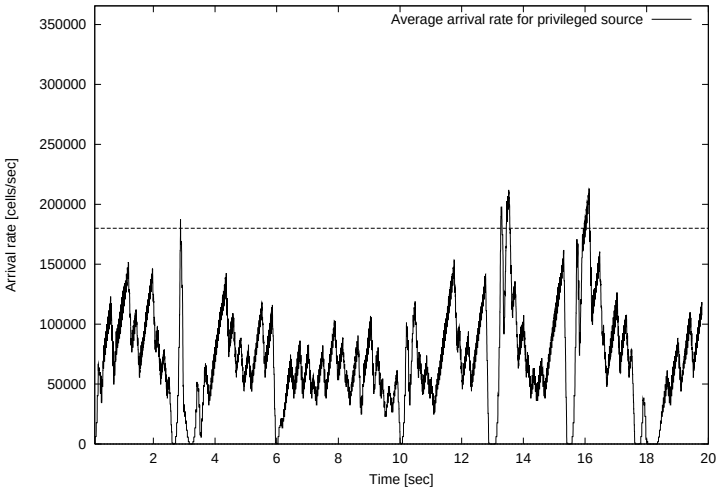

Figure 24 Arrival rate on bottleneck switch for privileged source

efficiently utilize its minimum guaranteed bandwidth and the unreserved bandwidth was shared among the different VCs in proportion to their respective MCR. When losses occured (e.g. in a LAN environement with small switch buffers or in a WAN environment), the performance of TCP was lower, but still much better than with the FIFO-based implementation. This lower performance was mainly caused by the retransmission and congestion avoidance mechanisms used by TCP which are not aware of the minimum guaranteed bandwidth of the underlying GFR VC. These performance problems could probably be partially solved by modifying TCP in a similar way as proposed in [8] for the controlled load service in an integrated services Internet.

When we used the GFR.2 conformance definition with the proposed WFQ-based switch implementation, the TCP performance was lower than when we used the GFR.1 conformance definition. This is mainly due to two factors. First, the TCP traffic is very bursty and a fraction of the AAL5-PDUs transmitted on a VC may be tagged by the UPC even if the long term average throughput of this VC is much below its MCR. Second, the proposed WFQ-based implementation discards these CLP=1 AAL5-PDUs when the buffer occupancy is relatively small.

Acknowledgments

We would like to thank Prof. J.-Y. Leboudec and S. Manthorpe of EPFL for having allowed us to use their STCP simulator. Without STCP, this work would probably not have been done. We would also like to thank J. Heinanen and R. Guérin for their comments on a draft of this paper.

References

[1] R. Atkinson. Default IP MTU for use over ATM AAL5. Internet RFC 1626, May 1994.

[2] O. Bonaventure. A simulation study of TCP with the proposed GFR service category. Technical Report SART 97/07/14, Research Unit in Networking, University of Liège, June 1997. Available from http://www-run.montefiore.ulg.ac.be.

[3] O. Bonaventure, E. Klovning, and A. Danthine. Behaviour of TCP in the European ATM Pilot. *Computer Communication*, 19(3):264–275, March 1996.

[4] O. Bonaventure, E. Klovning, and A. Danthine. Is VBR a solution for an ATM LAN ? In W. Dabbous and C. Diot, editors, *Proc. Fifth IFIP Workshop on Protocols for High Speed Networks*, pages 60–74. Chapman and Hall, 1996.

[5] H. J. Chao. A novel architecture for queue management in ATM networks. *IEEE Journal on Selected Areas in Communications*, 9(7):1110–1118, September 1991.

[6] J. Kenney (Ed.). Traffic management baseline text document. ATM Forum document BTD-TM-01.01, April 1998.

[7] K. Fall and S. Floyd. Simulation-based comparisons of Tahoe, Reno and SACK TCP. *ACM Computer Communication Review*, 26(3):5–21, July 1996.

[8] W. Feng, D. Kandlur, D. Saha, and K. Shin. On providing minimum rate guarantees over the internet. IBM Research Report 20618, version 2, May 1997.

[9] ATM Forum. ATM User Network Interface (UNI) Signaling specification, version 4.0. ATM Forum specification af-sig-0061.000, July 1996.

[10] ATM Forum. Traffic Management 4.0. ATM Forum specification af-tm-0056.001, 1996.

[11] S. Golestani. A self-clocked fair queuing scheme for broadband applications. In *IEEE INFOCOM94*, pages 636–646, 1994.

[12] R. Goyal, R. Jain, S. Fahmy, B. Vandalorre, S. Kalyanaraman, S. Kota, and P. Samudra. GFR – providing rate guarantees with FIFO buffer to TCP traffic. ATM Forum contribution 97-0831, September 1997.

[13] R. Guerin and J. Heinanen. UBR+ Service Category Definition. ATM Forum contribution ATM96-1598, December 1996.

[14] R. Guerin and J. Heinanen. UBR+ enhancements. ATM Forum contribution 97-0015, February 1997.

[15] J. Heinanen and K. Kilkki. A fair buffer allocation scheme. Available from ftp://lohi.dat.tele.fi/atm/papers/heinanen-fair-buffer.ps, 1996.

[16] H. Kroner, G. Hébuterne, P. Boyer, and A. Gravey. Priority management in ATM switching nodes. *IEEE Journal on Selected Areas in Communications*, 9(3):418–427, April 1991.

[17] J. Mahdavi. Experimental TCP selective acknowledgment implementation. Available from http://www.psc.edu/networking/tcp.html, 1996.

[18] S. Manthorpe. STCP 3.0 User Manual. Technical report, EPFL, September 1996.

[19] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgment options. Internet RFC 2018, October 1996.

[20] S. Pappu and D. Basak. TCP over GFR implementation with different service disciplines : a simulation study. ATM Forum contribution atm97-0310, April 1997.

[21] J. Roberts. Virtual spacing for flexible traffic control. *International Journal of Communication Systems*, 7:307–318, 1994.

[22] J. Roberts, P. Boyer, and M. Servel. A real time sorter with application to ATM traffic control. In *International Switching Symposium, ISS 95*, pages 258–262, 1995.

[23] A. Romanow and S. Floyd. Dynamics of TCP traffic over ATM networks. *IEEE Journal on Selected Areas in Communications*, 13(4):633–641, May 1995.

[24] K. Siu. A new data service in ATM. *IEEE Network Magazine*, 11(4):4–5, July-August 1997.

[25] W. Stevens. *TCP/IP Illustrated, volume 1 : The protocols.* Addison-Wesley, 1994.

[26] G. Wright and R. Stevens. *TCP/IP Illustrated Vol. 2, The Implementation.* Addison-Wesley, 1995.

[27] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83(10), October 1995.