TCP/IP and the European ATM Pilot

Olivier Bonaventure[a], André Danthine[a], Espen Klovning[b], Olivier Danthine[c]
[a] Institut d'Electricité Montefiore, B-28, Université de Liège, B-4000 Liège, Belgium
[b] Telenor Research and Development, P.O. Box 83, N-2007 Kjeller, Norway
[c] I.N.S, Consultant with Electricité de France, D.E.R. , F-92141 Clamart, France
Email: bonavent@montefiore.ulg.ac.be

Abstract

Very few measurements have been done on wide area ATM networks. In this paper, we first look at how a compliant traffic can be generated with standard ATM equipment. We show that in our environment, even with a 23.42 Mbps VP out of 34 Mbps links, it is necessary to use some spacing in the ATM adapters to generate a compliant traffic. Our measurements show that with a large window TCP is able to achieve a throughput of 19 Mbps when the ATM level traffic is compliant with the traffic contract. However, if the ATM level traffic is not entirely compliant the segment loss rate increases quickly, and the TCP throughput collapses. A 3% change in the cell rate of the ATM level traffic can cause the TCP throughput to drop from 19 Mbps down to 1 Mbps.

1. Introduction

During the last years, a large number of simulations and theoretical studies have been done concerning ATM networks, but most of the measurements have concentrated on ATM LANs [MKK94]. Several wide area ATM networks are currently being deployed in the world [ATM94a], [Net95]. These networks will initially be used for experimentation and measurements, but commercial deployment is expected to follow quickly.

Tomorrow's wide area ATM networks will be used to carry all types of traffic, ranging from computer-based applications or high-definition television to POTS. In the short term, however, we can expect that a large part of the traffic on these networks will be generated by computer-based applications. Most, if not all, these applications make an extensive use of the transport protocols (TCP, UDP, TP4, XTP, ...), and the performances of these protocols on the wide area ATM are critical for these applications.

In this paper, we will first present our measurement environment, with an emphasis on how to generate a compliant traffic, from a theoretical and a practical point of view. Then, we will present a simple model of TCP over ATM, and the main results of our measurements.

2. Measurement environment

Our measurement environment consisted of two distant ATM LANs interconnected with a wide area ATM network (the European ATM Pilot), two workstations and a set of measurement scripts built on top of ttcp[SlM85].

2.1 The European ATM Pilot

The European ATM Pilot [ATM94a], [ATM94b] is one of the first wide area ATM networks deployed in the world. This network has been built as a collaboration among 16 Public Network Operators from 15 countries, and covers most of Europe, from Finland to Portugal, and from Ireland to Austria. Each PNO participating in the Pilot network has installed at least one ATM switch and established links with its adjacent countries. Several brands and models of wide area ATM switches are used within the network. Most of the links in the European ATM Pilot are 34 Mbps E-3 ATM links [G703], but some links have already been upgraded to 155 Mbps SDH links [G707]. The measurements presented in this paper were done between the testbeds of the University of Liège in Liège, Belgium, and Electricité de France in Paris, France. These two testbeds were connected to their national ATM network with 34 Mbps links provided, respectively by Belgacom and France Telecom. As an international link between France and Belgium, a 34 Mbps link was used.

For the measurements presented in this paper, we used a VP with a requested bandwidth of 61000 cells per second (23.42 Mbps). Usage Parameter Control (UPC) functions were enabled in one public switch on the path between Liège and Paris. The geographical distance between the two testbeds is approximately 400km, and the round-trip-time (measured by "ping" with small packets) was approximately 8 milliseconds.

0-8186-7216-1/95 $04.00 © 1995 IEEE

2.2 The ATM LANs

Both ATM LANs consisted of one ASX-200 ATM switch from Fore Systems Inc. with one E-3 interface for the connection to the national ATM network, and one high-speed interface to connect the switch with the local workstations. The ASX-200 is a 2.5 Gbps bus-based, non-blocking, output buffered switch equipped with an internal Sparc 2 based controller. Each output port contains a buffer which can hold 256 cells. The UPC mechanism available in the ASX-200 was disabled during the measurements.

The ATM LAN in Liège was based on 155 Mbps SDH links, while the ATM LAN in Paris was based on 100 Mbps Taxi links. The workstations in Paris and Liège were Sparc 10 clones from Axil (model 311/5.x) running SunOS 4.1.3, and equipped with an SBA-200 ATM adapter from Fore. The Taxi 100 Mbps and SDH 155 Mbps SBA-200 ATM adapters from Fore offer similar performances. Measurements [MKK94] done in the local area have shown that the bottleneck on the SBA-200 is not the physical rate of the ATM link, but mainly the data transfers and the processing within the workstations. The maximum throughputs achieved by TCP with 100 Mbps Taxi or 155 Mbps SDH interfaces are similar (around 70 Mbps). For all the measurements presented in this paper, the sender and the receiver were in the same IP subnet. The sender was always the workstation with the 155 Mbps SDH ATM adapter.

2.3 Characteristics of a compliant traffic

In a wide area environment such as the European ATM Pilot, where the traffic contract is enforced by a UPC mechanism, it is essential for the user to generate a compliant traffic. If the traffic is not completely compliant, the UPC will discard the non-compliant cells.

When we performed the measurements presented in this paper, we had only limited capabilities to generate a compliant traffic. Since the ATM LAN switch we used did not include any device comparable to the spacer-controller [BSG92] to guarantee that the traffic sent by the switch is compliant with the traffic contract, we had to use the spacing capabilities of the ATM adapters.

Generally, spacing can be done at three different levels in the protocol stack : packet level spacing, cell-burst level spacing and cell level spacing. Figure 1 shows cell-burst and cell level spacing within a single AAL-PDU.

These three types of spacing differ by the "spacing unit". Packet level spacing uses the AAL-PDUs as the spacing units, while cell-burst level spacing is done per bursts of cells. Packet level spacing can be implemented in software, and a classical example of packet level spacing is the rate control mechanism used in transport protocols such as NetBLT or XTP. Cell-burst and cell level spacing require at least some hardware support from the ATM adapters.

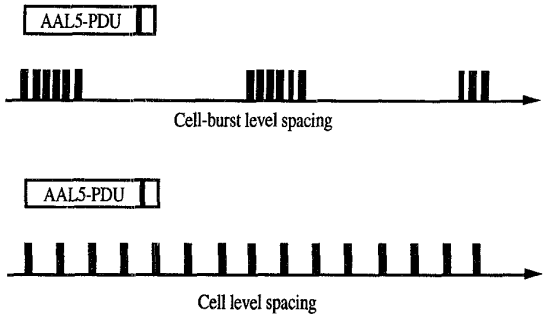

Figure 1 : Illustration of cell-burst and cell level spacings

2.4 Packet level spacing in our environment

The limitations of packet-level spacing can be made clear by looking at the main components of our environment. Figure 2 details the main components of the sending side of our environment.

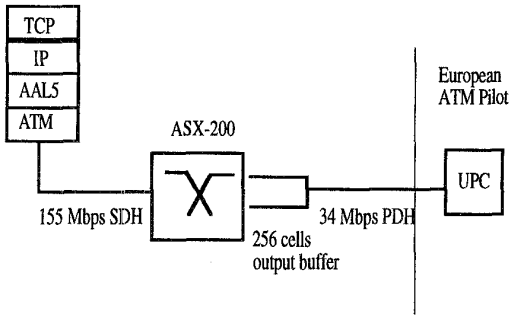

Figure 2 : Sending side of our environment

In our environment, cells can be discarded in two places. The local ATM switch will loose cells if there is an overflow in its output buffer (256 cells), and the UPC mechanism in the ATM Pilot will discard cells that do not comply with the traffic contract.

With packet level spacing, each AAL-PDU is sent as a burst of cells at the line rate (155 Mbps). If we assume that the output buffer was empty at the beginning of the burst, this burst will be absorbed, and transmitted on the 34 Mbps link by the ASX-200 provided its output buffer does not overflow. If T_i is the cell time on the input link of the switch, T_O the cell time on the output link of the switch, and L the size of the output buffer, measured in cells, the maximum burst size (in cells) is given by the following equation[1] :

[1] As a simplification, we neglect the physical layer framings on the SDH 155 Mbps and PDH 34 Mbps links.

$$maximum\_burst = L / (1- (T_i/T_O)) \quad (1)$$

In our local environment, T_i=2.726 μsec, T_O=12.39 μsec, L=256 cells, and thus the maximum burst size is 326 cells. After the 326 cells in the burst, an idle time of 256*12.39 = 3172 μsec is necessary to empty the output buffer.

However, a burst absorbed by the ASX-200 must still be compliant with the traffic contract enforced by the UPC mechanism. The traffic contract enforced by the UPC mechanism was 61000 cells/sec (one cell every 16.39 μsec), with an allowed Cell Delay Variation (CDV) of 101μsec ([T=16.39 μsec, τ=101μsec] with the notations of the Virtual Scheduling Algorithm [I.371]). Generally, if T_j is the cell time at the input line of the UPC, T_u and τ_u respectively the cell rate and the cell delay variation enforced by the UPC, the maximum burst size declared as compliant by the UPC is given by the following equation :

$$maximum\_burst = 1+ (\tau_u / (T_u - T_j)) \quad (2)$$

In our environment, T_j=12.39μsec, T_u=16.39 μsec, τ_u=101μsec, and thus the maximum burst size declared as compliant by the UPC is 26 cells. This burst of 26 cells has to be followed by an idle time of at least 104 μsec to let the UPC return to the idle state. Thus, the most bursty traffic allowed by the [T=16.39 μsec, τ=101μsec] is composed of bursts of 26 cells separated by an idle time of 104 μsec.

As shown by equation (2), the maximum burst size declared as compliant by the UPC depends on both the cell rate of the traffic contract and the cell rate at the input of the UPC. Thus, it is interesting to look at the consequences of two possible changes in our environment on this maximum burst size.

If the traffic contract on the ATM Pilot was 11000 cells per second (4.22 Mbps) instead of 61000 cells per second, with the same Cell Delay Variation, the longest compliant burst would have contained 2 cells.

Similarly, if the workstation was directly connected to the ATM Pilot with a 155 Mbps SDH link, the longest burst declared as compliant by the [T=16.39 μsec, τ=101μsec] UPC would have contained 7 cells.

2.5 The ATM adapters

In section 2.3, we calculated the maximum usable burst sizes that can be accepted by the local ATM switch and the UPC mechanism in the European ATM Pilot. In practice, however, generating a compliant traffic is not necessary a trivial task.

It was impossible to use packet level spacing with TCP in our environment, because TCP does not include any rate control mechanism, and any SDU-level spacing introduced by the application will be completely jeopardised by the window-based flow control used by TCP. Thus, the only possible solutions were cell level spacing and cell-burst level spacing.

For our measurements, we used SBA-200 ATM adapters from Fore Systems Inc. Cell level and cell-burst level spacings were provided with two different releases of the ATM drivers.

Cell level spacing was provided with release 2.2.9 of the ATM driver and a modified version of the firmware for the i960 processor on the SBA-200. According to the documentation supplied by Fore, this firmware is able to insert an idle time of n *0.492 μsec between two consecutive assigned cells transmitted on the output link. This value n, and thus the peak rate of the output link, can be changed by the superuser.

Cell-burst level spacing was provided with release 2.3.0 of the ATM driver. We found that the behaviour of this driver was a function of the requested rate. The driver seemed to choose the burst length as a function of the requested rate. According to measurements presented in [KK95], the length of the bursts ranges from 6 to 11 cells.

In order to have a better understanding of the behaviour of the two types of spacing implemented on the SBA-200 adapters, we performed the following experiment at Telenor Research and Development. One workstation, equipped with a Taxi 140 Mbps SBA-200 sent a train of 8 kbytes (i.e. 172 cells) long UDP packets. This workstation was connected, through an ASX-200 ATM switch, with an Alcatel ATGA ATM tester equipped with a 155 Mbps SDH interface. The ATGA measured the cell interarrival time (IAT), i.e. the difference measured in cell slots between two successive assigned cells, of the ATM cells in the UDP packets. We performed this experiment with the two releases of the driver, and the requested throughput at the ATM level was in both cases 22.3 Mbps.

Figure 3a shows that release 2.2.9 produces an almost regular traffic. The only significant variation in the IAT is an increase of 7-8 cell slots[2] every 21 or 22 cells. This regular pattern coincides with the boundaries between 1 kbyte cluster mbufs in the UDP packet. The reason for this sudden increase is most likely caused by the operation of the DMA controller during the spacing operation. That is, it looks like the DMA controller uses some sort of m-cell deep pipeline to manage the spacing.That is, it looks like the DMA controller uses an m-cell pipeline to manage the spacing. The actual implementation is not known[3] but our measurements can be explained with a pipeline design. Thus, the pipeline will be empty after the last cell of the previous mbuf was transmitted, and it will take 7 or 8 cell slots to fill the pipeline and enable spaced cell transmission again. The smaller variations, of one cell slot, are probably caused

[2] As the IAT is measured on the 155 Mbps SDH link, one cell slot is 2.726 μsec

[3] Fore Systems Inc. did not supply detailed information about the internal workings of the spacing.

by the ASX-200 switch used for the conversion between 140 Mbps Taxi and 155 Mbps SDH.

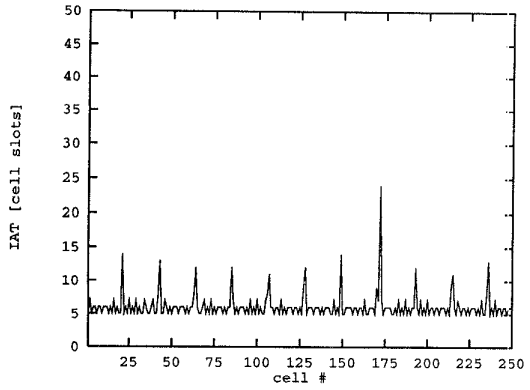

Figure 3a - Spacing with the Fore 2.2.9 driver

Figure 3b shows that driver 2.3.0 transmits 6 cells back-to-back in a burst, with an inter-burst time of almost 40 cell slots. The average IAT is 6-7 cell slots. The traffic generated with release 2.3.0 is less regular than the traffic generated with release 2.2.9 of the ATM driver. Additional details about the behaviour of the spacing in these two versions of the ATM drivers with an emphasis on CDV are presented in [KK95].

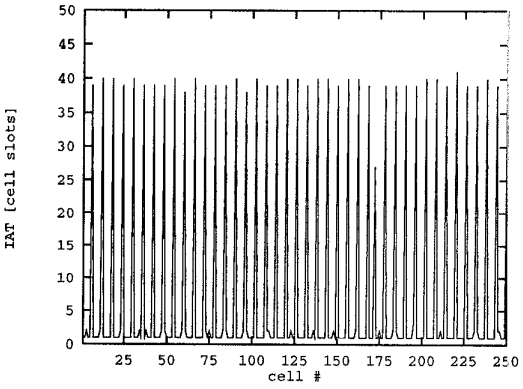

Figure 3b - Spacing with the Fore 2.3.0 driver

For our measurements, we choose to use release 2.2.9 of the Fore ATM driver because it generates a much more regular traffic than release 2.3.0.

3. A simple model of TCP over ATM

Before looking at the results of our measurements, it is interesting to have in mind a simple model of the behaviour of TCP over ATM. The goal of this simple model is mainly to deduce an estimate of the maximum TCP throughput achievable in our environment. For this simple model, we make the following assumptions :

- No cells are lost or discarded in the network
- The sender and the receiver are infinitely powerful, and thus there is no delay due to the processing of the segments and the acknowledgements
- The TSDU size (i.e. the size of the buffers transmitted with the send() system call) does not have an influence on the achieved throughput.[4]

In this simple model, we mainly consider the influence of the window size, and the acknowledgement policy on the achievable throughput.

3.1 The flow of data

To achieve a high utilisation of the links, TCP always tries to use the largest possible segment size to transfer the user data. When TCP is used in an ATM subnet, the maximum segment size (MSS) is given by the difference between the MTU size of the ATM subnet, and the TCP+IP headers (40 bytes). In our environment, we used VC-based multiplexing, and thus the default MTU size was 9188 bytes [Atk94], and TCP computed an MSS of 9148 bytes.

3.2 The window size

TCP uses window-based flow control, and the maximum window size is 65535 bytes [Pos82]. In most TCP implementations, it is possible to select the window size on a per connection basis. However, due to the Silly Window Syndrome (SWS) avoidance mechanism [Cla82] and the Nagle algorithm [Nag84], TCP will only use part of the allocated window size. The SWS avoidance mechanism prohibits TCP to send segments smaller than one MSS (or half the maximum window size advertised by the receiver) bytes long. The Nagle algorithm prohibits TCP to send a small segment (i.e. a segment smaller than one MSS) when other segments are still unacknowledged. As we neglect the influence of the TSDU size, we can consider that when the requested window is larger than 2*MSS bytes, the usable window is the integer number of MSS-long segments contained in the window. For example, a window of 48 Kbytes will be actually reduced to a usable window of 45740 bytes (i.e. 5*MSS). With a window of 16 Kbytes, however, these rules allow TCP to send two segments of 8192 bytes, and thus the window can be completely utilised.

[4] This assumption is not always valid for small window sizes

3.3 The acknowledgement policy

Another important parameter that we must take into account in our simple model of TCP is the acknowledgement policy. The original TCP specification [Pos81] does not impose a particular acknowledgement policy. A simple, and inefficient, policy would be to generate one acknowledgement when a complete window has been received. Another policy would be to generate one acknowledgement for each segment received. [Bra89] states that a TCP implementation should send one acknowledgement for at least every second segment. Other acknowledgement policies are possible. In SunOS, for example, an acknowledgement is sent if :

a- The window sequence number edge will slide at least 35% of the maximum socket receive buffer.
b- The highest announced window sequence number edge will slide by at least twice the MSS size.

3.4 The simple model applied to our environment

The simple model presented above can be used to calculate an estimate of the maximum throughput achievable with TCP in our environment. The round-trip-time, measured by ping, was 8 msec, and the traffic contract allowed a cell rate of 61000 cells per second (23.42 Mbps). Figure 4 shows the calculated maximum throughput as a function of the window size for the three acknowledgement policies presented in the previous section.

Figure 4 can be divided into three regions depending on the window size :

a-window $\leq$ 2* MSS

In this region, the Nagle and SWS avoidance mechanisms do not constraint the usable window, and thus the whole window can be utilized. However, the window size is smaller than the bandwidth*delay product, and thus the VP is not 100% utilized.

b- 2*MSS < window $\leq$ 4*MSS

In this region, only the fraction of the window corresponding to an integer number of MSS-long segments is effectively utilized. This is due to the Nagle and the SWS avoidance algorithms. By acknowledging every second segment, SunOS further reduces the usable window to an even number of MSS-long segments.

c- 4*MSS < window

In this region, the window is larger than the bandwidth*delay product, and thus the VP can be completely utilized.

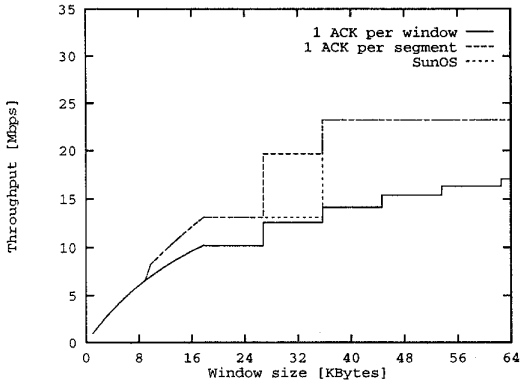

Fig. 4 - Maximum throughput calculated with the simple model

4. Selected throughput measurements

In order to measure the performances of TCP over the European ATM Pilot, we performed a large number of measurements, and transferred several hundred gigabytes of data through the network. Each TCP measurement was a memory-to-memory transfer of 16 megabytes between the sender and the receiver with a measurement tool based on ttcp [SIM85]. The workstations, the local ATM switches, and our VP in the ATM Pilot were exclusively used for the measurements. Thus, there were no contentions due to the sharing of our VP by several TCP connections.

The results showed that the TCP throughput is very sensitive to segment losses. In the following, we will present results from several individual measurements instead of average values corresponding to a large number of measurements. Due to the negative influence segment losses can have on the throughput, average values are not always meaningful. However, the results presented here are representative of the problems we encountered during our measurements.

4.1 Measurements without spacing

A common belief within the data communication community is that TCP can adapt to any kind of bandwidth. This belief is mainly based on TCP's excellent track record. For example, if two FDDI rings are interconnected with routers and a two hundred kilobits per second leased line, a TCP connection established between two workstations on the distant

rings will adapt itself dynamically to the bottleneck link, and TCP will usually achieve a high utilisation of that link. It is tempting to extrapolate this example to TCP over ATM. In our case, we have two high-speed ATM LANs interconnected (through ATM switches, and not routers) by a 61000 cells/sec VP in an E-3 link. However, one should remember that when congestion occurs, or when the traffic contract is exceeded in an ATM network, the network looses ATM cells, but not complete TCP segments. This may result in a flow of ATM cells but no flow of correct TCP segments, and thus no real data-transfer.

We tried to perform some measurements without enabling the spacing on our ATM adapters, and it was impossible to transfer any data with TCP. The TCP connection was easily established (the SYN segments and the corresponding ACKs are contained in a single ATM cell). However, **it was impossible to perform any data transfer**. This problem was caused by the MTU size of 9188 bytes selected for IP over ATM [Atk94]. When spacing is not enabled, the ATM adapters send the AAL-PDUs containing TCP segments as bursts of 192 cells, at the line rate. While a single burst of 192 cells can be absorbed by the output buffer of the local ATM switch, the resulting burst on the 34 Mbps link is not compliant, and at least one cell from each TCP segment containing data is discarded by the UPC of the ATM Pilot (see section 2.3). Thus, ATM level spacing was really necessary, even with TCP, in our environment.

4.2 Measurements with a compliant traffic

Our first measurements were done with a compliant traffic. The spacing was set to one cell every 19.45 µsec, while the traffic contract was one cell every 16.39 µsec. Figure 5 presents the results of a typical measurement with the spacing set to one ATM cell every 19.45 µsec. A spacing of 17.98 µsec per cell gave similar results.

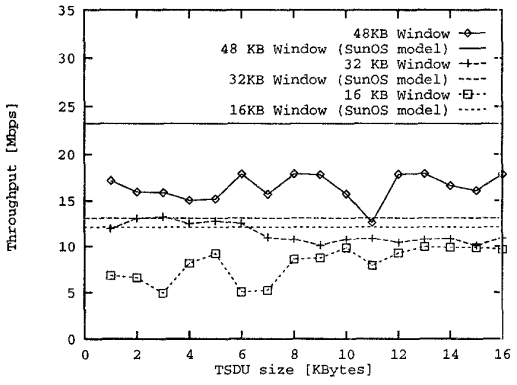

Figure 5 - TCP throughput with spacing set to 1 cell every 19.45 µsec

As expected, the highest throughput is obtained with the largest window size. In SunOS 4.1.x, the largest window size is 52428 bytes. However, as shown in section 3.1.2, this window size is equivalent to the 48 KB window we used.

The throughput variations shown in figure 5 are mainly caused by limited segment loss (the segment loss rate was lower than 0.15%) and corresponding retransmissions. As each TCP segment contains 192 cells, the cell loss rate is approximately $7 * 10^{-6}$. We notice that the losses mainly occur when the window size is large, but the reason for these segment losses is not known.

The requested rate on the ATM Pilot was one cell every 16.39 µsec, with an allowed CDV of 101 µsec. Due to the granularity of the spacing used by our SBA-200 ATM adapters, it was impossible to achieve exactly this cell rate. The closest possible value for the spacing, while still remaining compliant was one cell every 16.50 µsec. As shown in figure 6, the throughput achieved with the spacing set to 16.50 µsec per cell is comparable with the throughput achieved with the spacing set to 19.45 µsec per cell. Thus, the traffic sent with the spacing set to one cell every 16.50 µsec is still a compliant traffic. The throughput drops shown in figure 6 are caused by a small number of segment losses (the segment loss rate was lower than 0.2%) and the corresponding retransmissions.

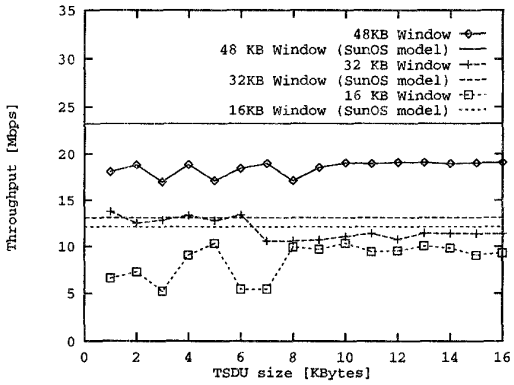

Figure 6- TCP throughput with spacing set to 1 cell every 16.50 µsec

4.3 Measurements with an almost compliant traffic

In order to see what happens when the ATM level traffic is not completely compliant, we performed the same throughput measurements with a rate of one cell every 16.01 µsec (this was the closest non-compliant cell rate

achievable with our SBA-200 ATM adapters). Figure 7 shows the throughput measured with this cell rate.

The most affected traffic is the traffic sent with a 48 KB window. Figure 7 shows that with this window size, the throughput drops below 1 Mbps (compared to 19 Mbps when the spacing was set to one cell every 16.50 µsec). This collapse of the TCP throughput is due to the high segment loss rate, as shown in figure 8.

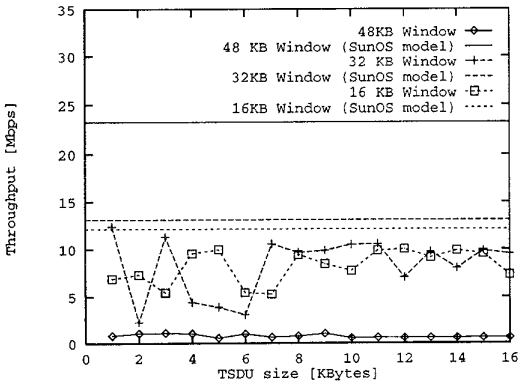

Figure 7- TCP throughput with spacing set to 1 cell every 16.01 µsec

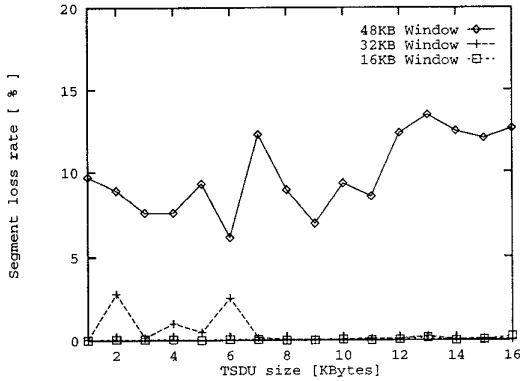

Figure 8 - Segment loss rate with spacing set to 16.01 µsec per cell

In order to explain this high segment loss rate, we traced the TCP packet flow during these measurements, and found that when a segment was lost with a 48 KB window, it was always the third segment in a burst of three segments. However, in a few three segment bursts, the third segment was not lost. From the packet traces, it seemed that bursts of two TCP segments were compliant, while bursts of three TCP segments were not compliant. In our environment, and with the spacing set to one cell every 16.01 µsec, the ATM traffic corresponding to a burst of three segments looks like figure 9. Each TCP segment contains 192 cells.

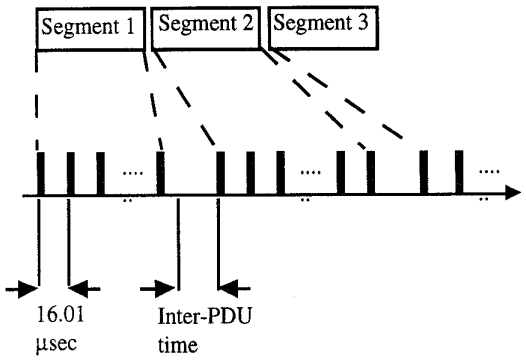

Figure 9 - ATM traffic corresponding to a burst of three TCP segments

We did not measure the inter-PDU accurately with TCP. However, in section 2.5 (figure 3a), we have shown that 52 µsec was a lower bound on the inter-PDU time measured with UDP. This inter-PDU time is caused by the firmware processing on the SBA-200 ATM adapter, and not by the processing in the higher layer protocols, and thus we can assume that the same value also holds for TCP segments.

According to the Virtual Scheduling Algorithm [I371], [ATM93], all the cells of these three segments will be compliant with the traffic contract [T=16.39 µsec, τ=101µsec] if[5] :

$\forall$ i=0...191 :

$$i * 16.39 \leq i * 16.01 + 101 \quad (3)$$

$\forall$ j=0...191 :

$$(192+j)*16.39 \leq 192*16.01+52+j*16.01+101 \quad (4)$$

$\forall$ k=0...191 :

$$(2*192+k)*16.39 \leq 2*(192*16.01+52)+k*16.01+101 \quad (5)$$

Conditions (3) and (4) are true, and thus all the cells of the first two segments are compliant. However, condition (5) is false, and thus at least one cell of the third segment is declared as non-compliant and discarded by the UPC in the ATM Pilot. It should be noted that condition (5) could be true if the inter-PDU time was a bit larger, and this explains that some bursts of three segments found in the traces were compliant.

As shown above, bursts of two segments are compliant with the traffic contract. This explains why the segment loss rate with a 16 KB window is still less than 0.25%. The segment loss rate with a 32 KB window is higher,

[5] Here we assume that the UPC is in the idle state upon arrival of the first cells.

especially with the smaller TSDU sizes, and this segment loss rate affects the TCP throughput. When the ATM level traffic is not completely compliant, a large window may give a much lower throughput than a smaller window.

In a forthcoming paper, we will discuss, based on packet traces gathered during the measurements, the TCP behaviour in more details and suggest some possible solutions to avoid some of the problems we found during our measurements.

5. Conclusions

In this paper, we first looked at how we could generate a compliant traffic in our measurement environment, which consisted of two ATM LANs connected with a 23.42 Mbps VP through the European ATM Pilot (which uses 34 Mbps links). We have shown that spacing was really necessary, even if our VP uses a large fraction of the raw bit rate of the access line. We have also presented how the spacing was actually performed in our Fore SBA-200 ATM adapters.

Then, we have presented a simple model of TCP over wide area ATM and the results of our measurements. A single VP was exclusively used for these measurements, and at any time only one TCP connection was using this VP. Thus, there were no contentions due to different TCP connections sharing the same VP.

These measurements showed that when the ATM level traffic is compliant, the TCP throughput is, as expected, mainly dependent on the window size. However, if the ATM level traffic is not completely compliant, the segment loss rate increases rapidly, and the TCP throughput collapses. A small variation in the cell rate can have a high impact on the segment loss rate and the TCP throughput. Thus, when working over a wide area network such as the ATM Pilot, it is really necessary to generate a compliant traffic, and TCP alone cannot by itself overcome the problems caused by the non-compliance of the ATM level traffic.

Acknowledgements

This work was partially supported by the European Commission within the RACE 2060 CIO project and by a contract between the University of Liège and Belgacom.

References

[Atk94] R. Atkinson, "Default IP MTU for use over ATM AAL 5", RFC 1626, May 1994

[ATM93] ATM User Network Interface Specification, Version 3.0, ATM Forum, September 1993

[ATM94a] Info-Booklet of the ATM-Pilot Network, ATM Pilot Coordination Group, April 1994

[ATM94b] Wide Area ATM Deployment in Europe, White Paper, The European Market Awareness and Education Committee, ATM Forum

[Bra89] R. Braden, Ed., "Requirements for Internet Hosts -- Communication Layers", RFC 1122, Oct. 1989

[BSG92] P. Boyer, M. Servel, F. Guillemin, "The SPACER-CONTROLLER : An Efficient UPC/NPC for ATM Networks", XIV Int. Switching Symposium, Yokohama, Oct. 1992

[Cla82] D. Clark, "Window and Acknowledgement Strategy in TCP", RFC 813, July 1982

[G703] ITU-T, "Physical/electrical Characteristics of Hierarchical Digital Interfaces", ITU-T G.703

[G707] ITU-T, "Synchronous Digital Hierarchy Bit Rates", ITU-T Rec. G.707

[I371] ITU-T, "Traffic Control and Congestion Control in B-ISDN", ITU-T Rec. I.371, 3/1993

[KK95] E. Klovning, O. Kure. "Host adapter spacing in SBA-200", Kjeller, Telenor Research & Development, 1995. (TF scientific document N17/95)

[MKK94] K. Moldeklev, E. Klovning, O. Kure, "TCP/IP Behaviour in a High-Speed Local ATM Network Environment", Proc. 19th Conference on Local Computer Networks, Minneapolis, October 2-5, 1994, pp. 176-185

[Nag84] J. Nagle. "Congestion control in TCP/IP internetworks". Internet Engineering Task Force, RFC 896, 1984.

[Net95] IEEE Network Magazine, Special Issue on the North Carolina Information Highway, Nov./Dec. 1994

[Pos81] J. Postel, "Transmission Control Protocol", RFC 793, Sept. 1981.

[SlM85] T. Slattery, M. Muss, TTCP.C, available as ftp://ftp.brl.mil/pub/ttcp.c