Using QoS for High Throughput TCP Transport over Fat Long Pipes

Andrea Di Donato
Telecommunications Research Engineer
add@hep.ucl.ac.uk
www.hep.ucl.ac.uk/~add

Yee-Ting Li
e-science PHD Student
ytl@hep.ucl.ac.uk
www.hep.ucl.ac.uk/~ytl

Miguel Rio
Lecturer
M.Rio@ee.ucl.ac.uk
www.ee.ucl.ac.uk/staff/~mrio

Peter Clarke
Head of the Particle Physics Group
clarke@hep.ucl.ac.uk
www.hep.ucl.ac.uk/~clarke

University College London
Department of Physics and Astronomy
Gower Street
London, WC1E 6BT
United Kingdom

February 25, 2004

Abstract

A very practical network problem for the effective deployment of a transatlantic computational GRID which is the current under-utilisation of the newly available fat long pipes drives the following work towards the investigation of whether new TCP stacks can solve such issue. The paper proves the necessary use of IP-QOS and investigates the conditions under which IP-QOS and new/standard TCP stacks can coexist such that high utilisation as well as good stability and inter-class protection is achieved.

1 Introduction

More and more new distributed applications are designed to run on the GRID [**Grid**] network infrastructure to accomplish to their tasks on a worldwide scale. Most of these applications rely heavily on the performance of the TCP protocol.

The ratio of the link capacity over its price has been accelerating in the past few years and this makes it more cost effective to upgrade the capacity of the network rather than to engineer a lower speed one. This acceleration is much faster than the observed bandwidth usage from traditional Best Effort (BE) traffic. Thus in the short to medium-term scenario there is excess capacity available, especially in the core.

The GRID network infrastructure is firstly being developed in academic networks and as with the internet, the initial users of the above mentioned spare capacity are applications developed by scientists involved in areas such as particle physics, radio-astronomy and biology. Based on emerging applications within this areas, such as BaBar [**Babar**], the main requirement is a reliable and effective bulk data transfer at multi-gigabit per second speed over long distances. This invariably involves and depends on the performances shown by TCP as it is the protocol delegated to accomplish the end to end transport.

TCP's congestion control algorithm additive increase multiplicative decrease (AIMD) performs poorly over paths of high bandwidth-Round-Trip-Time product [**Stevens**][**Floyd**] and the aggregate stability is even more precarious if such aggregate consists of only few flows [**Low**].

These well known facts pose a serious question on how to effectively deploy GRID given that multi-gigabit per second capacity can be provisioned on trans-continental links for few users/flows at any one time. So, although there is in principle spare capacity, TCP as it is now, will impact negatively on the performance of these new applications and will compromise the whole concept of a high performance computational GRID.

Two recent developments can significantly contribute to tackle this problem: Proposals for high throughput TCP stacks and the availability of Differentiated Services [**Diffserv**] enabled networks at multi-gigabit speeds.

In this paper we investigate the relation between IP-QoS configuration in the routers and the dynamic of standard TCP as well as that of new proposals for high throughput TCP, including High Speed TCP [**Hstcp**] and Scalable TCP [**Scalable**]. We conduct extensive experimental tests in a high bandwidth, high propagation delay research network.
To perform our tests we used the DataTAG testbed [**Datatag**] which consists of a transatlantic link connecting Geneva to Chicago. We used Juniper [**Juniper**] M10 routers with Diffserv-enabled GigaBit Ethernet cards (a choice made after having benchmarked several router manufacturers). This testbed is unique in providing a Differentiated Services network with high propagation delay and bandwidth capacity on the order of Gigabits per second. To generate traffic we used high-end multiprocessor PCs running Linux 2.4.20 kernels.

2 Background

In [**Low**] a non linear distributed feedback system model of TCP and Active Queue Management [**Qos**] is provided. This is then linearised for small oscillations of the throughput or the sender congestion-window size (CWND) [**Stevens**] in time around the equilibrium. The derived inequality expression governing the stability condition for the TCP flow dynamic in time follows below.

$$\rho[(c^3 \tau^3)/(4N^4)] * [(c\tau/2N)+1+1/(\alpha c\tau)] < [\pi(1-\beta)^2] / \sqrt{[4\beta^2 + \pi^2(1-\beta)^2]} \quad \textbf{(Eq. 1)}$$

Where :
ρ = slope of drop probability in the router queues
α = weight by which the current router queue size is taken into account in the estimation of the average queue occupancy performed by the router.
c = Link Capacity
N = Number of flows belonging to the aggregate
τ = RTT

The non linear feedback system is stable if c and τ decrease and N increases which is exactly the opposite of what happens in the scenario envisaged in the introduction where few users/flows at any one time will use fat long pipes. This system can also be stabilised if ρ decreases and α increases.
If α increases, then the current queue value is taken into account more quickly (High-Pass filter), this way increasing the bandwidth of the controller which, in turn, improves the reactivity of the TCP flow.

3 Test-bed layout and equipment

This paper is based on the current work being done for the project DataTAG .The goal of the DataTAG project is to create a large-scale intercontinental test-bed for data-intensive Grids. The focus is mainly on the network research over a high-performance dedicated 10 Gbps path between CERN in Geneva (Switzerland) and Starlight in Chicago (USA).
The test-bed layout is shown below.

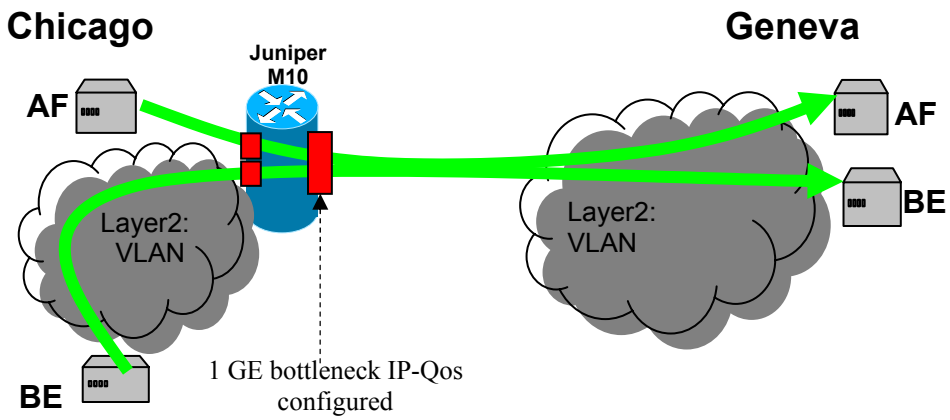

Figure 1. The layout of the testbed

Both Geneva and Chicago site used Supermicro 6022P-6 Dual Intel® Xeon [**Supermicro**] as high-end PCs. Each PC had a Syskonnect gigabit ethernet controller.
The PCs were running Linux kernel version 2.4.20.
As far the TCP/IP path concerns, the minimum RTT experienced is 118 ms and the bottleneck is of 1Gbps capacity.
The Juniper router used belongs to the M10 series. The IOS is "Junos 5.3R2.4" and the version of the card where congestion happens and therefore where the IP QOS is configured is the "1x G/E, 1000 BASE-SX REV 01".
Traffic is generated by a user space software tool [**iperf**].

4 Standard Vs New TCP stacks: the inadequacy of Standard TCP in fat long pipes

The first set of tests was conducted without configuring QOS. The purpose is to show that a small number of standard-TCP flows are not adequate to take advantage of the very large spare capacity when operating in a high RTT environment. For this reason, an "aggregate" of one TCP flow competing with various loads of CBR background traffic is the basic test chosen which is then repeated for all the different TCP flavours and their performance are evaluated.
Figure 2 shows how poor the Standard (Vanilla) TCP throughput is even when competing with very small CBR background traffic.
In Figure 3 the coefficient of variance (CoV) for the throughput in Figure 2 is plotted.

The CoV is simply defined as the ratio between the standard deviation and the mean value of the throughput [1]. It shows that all the TCP stacks oscillates. For Standard (Vanilla) TCP, both the average throughput of Figure 2 and the CoV of Figure 3 show little dependence on the available bandwidth unlike the other stacks. This rather passive behaviour reveals that Standard TCP is weak in a non-protected path of these ranges of bandwidth-delay products.

The outcome of these tests is that new TCP proposals can work well as far as the utilisation of the spare capacity is in a non protected environment, while standard TCP struggles.

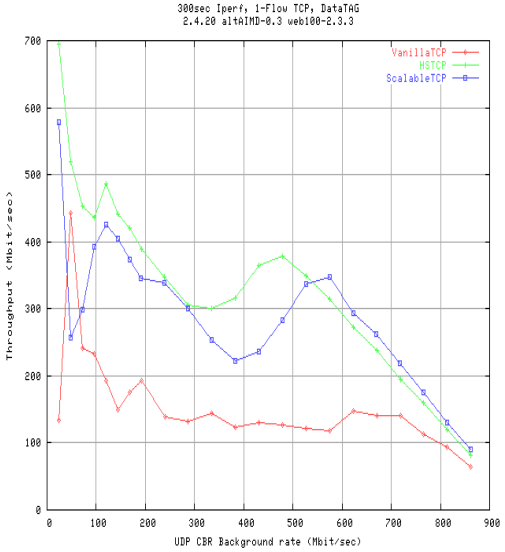

Figure 2 One TCP flow Throughput Vs. CBR background offered load

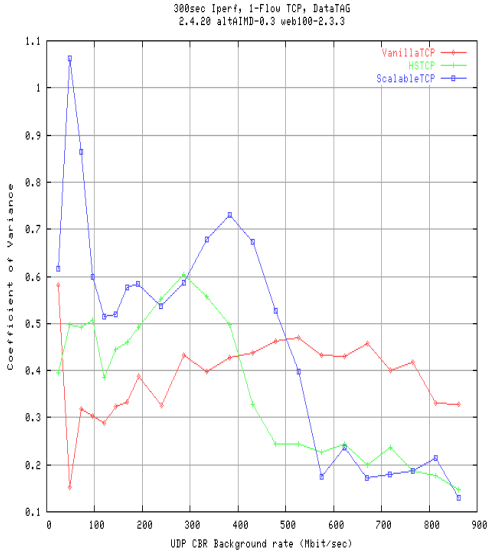

Figure 3 One TCP flow CoV Vs. CBR background offered load

5 New stacks: Impact Factors Investigation

Naturally, the outcome of the above tests suggests the use of new TCP proposals with the aim of getting better utilisation of the available bandwidth – hence good overall link utilization – and no disruption of the native traditional BE traffic. These two metrics are quantified through the computation of one parameter which we refer to as the "user impact factor" (UIF); an interpretation of fairness.

$$UIF = \frac{NVanillaTCPflowsThroughputWhenWith1NewTCP * \frac{1}{N}}{(N+1)VanillaS\tan daloneThroughput * \frac{1}{(N+1)}} \quad \textbf{Equation 2}$$

[1] Coefficient of variance = Stdev Throughput / Mean Throughput

UIF measures the impact of 1 new TCP on the throughput achieved by a legacy TCP user. A value of UIF < 1 means that the Throughput experienced by a legacy flow as a part of a legacy aggregate of N+1 flows is bigger than that experienced by the same legacy flow as part of an aggregate of N legacy flows mixed with one new TCP stack flow.

What is shown in figures 4 and 5 below is actually the inverse of UIF. The greater than one 1/UIF is the greater is the magnitude of the impact of 1 new TCP stack on the traditional BE traffic.

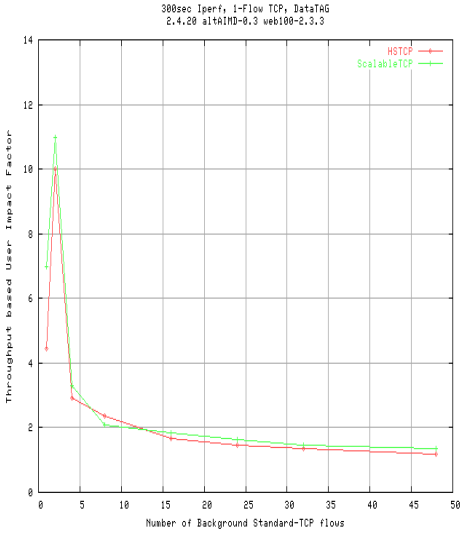

Figure 4 1 New TCP Flow User Impact Factor (UIF)

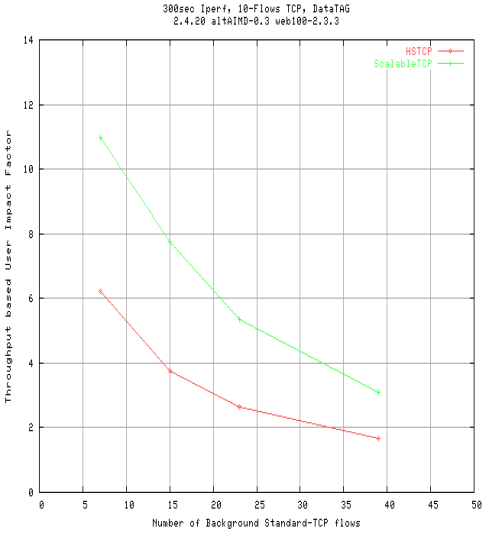

Figure 5 10 New TCP Flows User Impact Factor (UIF)

The composite analysis of the throughput achieved in figure 2 and 3 together with the values 1/UIF achieved, clearly shows that these new TCP proposals can work well as far as the utilisation of the spare capacity is concerned but that they fail in that they are invasive towards the traditional BE traffic.

6 The need for QoS

The results obtained so far suggest we need a mechanism for segregating traffic sharing the same packet-switched path. This is in order to protect traditional BE traffic from the possible aggression of the new TCP stacks. This segregation mechanism would also guarantee a level of end-to-end service predictability for the new TCP proposals which is sufficient to enforce a network resources reservation system through the use of the GRID middleware.

Two IP-QOS classes are therefore configured: BE for traditional Best Effort traffic (WEB-like traffic) and Assured Forwarding (AF) hosting those hungry few TCP flows belonging to the new networked GRID applications. By varying the bandwidth allocation and shooting CBR traffic with a negative offset of 30 sec, we will measure how well in terms of Received Throughput and Congestion Windows Dynamic the different TCP stacks adapt to the IP-QOS allocated bandwidth.
Traffic is marked with the Diffserv code point (dscp) [**Diffserv**] at the sender host and each class is put in a physically different router output queue. Each queue is then assigned a minimum guaranteed bandwidth by using weighted round robin [**Qos**]. AF is assigned a de-queuing priority (see Appendix) with respect to the BE class in order to help elastic TCP AF traffic when in competition with persistent User Datagram Protocol (UDP) BE traffic.

Before sending TCP traffic through a QOS-enabled network we must investigate the performance of the bandwidth scheduler itself.
This task is accomplished by sending UDP Constant Bit rate (CBR) traffic for both classes and by checking whether the throughput received is that expected throughout the whole bandwidth allocation set spanned.
The router configuration is changed on the fly in the following range of the bandwidth allocation: 90:10, 70:30, 50:50, 30:70, 10:90. The results shown in figure 6 are as expected. Specifically, the BE and AF flow are the darker and the lighter[2] lines respectively.
The received throughput is on the ordinate axis while the bandwidth allocation is changed every 60 to 80 seconds.

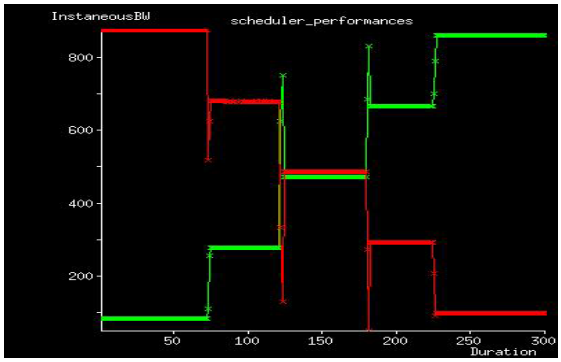

Figure 6: bandwidth scheduler benchmarking; BE (darker), AF (lighter)

7 TCP in the AF class

[2] If not explicitly stated in the legend, we will differentiate the curves based on their degree of grey scale. For the colour readers the lighter and the darker lines correspond to the green and red lines respectively. We will adopt this association in the rest of the paper.

In this section, the aim is to find the conditions under which IP-QOS and Standard/New TCP stacks can coexist, such that high bandwidth utilisation as well as good stability and inter-class protection is achieved.
Each TCP stack used (Standard, HS-TCP and Scalable) is tested against 1Gbps BE UDP CBR background and for a bandwidth allocation range of values: 90:10, 70:30, 50:50, 30:70, 10:90.
The aforementioned utilization, stability and class protection are quantified through the computation of two parameters which we refer to as the "BE QoS efficiency factor" (BE-QEF) and the "AF QoS efficiency factor" (AF-QEF) for the BE and AF class, respectively.

$$\text{BE-QEF} = \frac{UDP\_BE\_Throughput}{Allocated\_BE\_bandwidth} \qquad \textbf{Equation 3}$$

$$\text{AF-QEF} = \frac{TCP\_AF\_Throughput}{Allocated\_AF\_bandwidth} \qquad \textbf{Equation 4}$$

What follows is a survey of the AF/BE-QEF achieved throughout the bandwidth allocation set by each TCP stack used. Some modification to the test conditions are then presented in the remaining part of the paper with the aim of finding the host/network conditions which guarantee the best performance for each of the stacks used.

7.1 Survey results

Before discussing the results, it is worth remembering that the TCP dynamics suggests two things:

1. For the new TCP stacks tested, the bigger the allocated bandwidth, the higher the loss rate induced by the scheduler during congestion since the loss recovery (congestion avoidance gradient) for bigger CWND is more aggressive.
2. Also, a large bandwidth allocation means, for all the stacks involved, higher CWND values. This means a higher probability of dropping more packets in the same CWND.

The following results are obtained for a TCP connection whose maximum socket buffer size is of 100Mbytes, with "Congestion moderation" (see section 8) turned on and with "txqueuelen" and "backlog"[3] of 50000. The test were conducted by leaving a 1Gbps UDP BE flow running and by switching on and off different TCP stacks for different bandwidth allocation couples.
During the measurements, we noticed an initial period where the throughput of the TCP flow oscillates with high amplitude between zero and the allocated bandwidth. This is then followed by a period of relative stability at the allocated bandwidth. We refer to the combined period as the transient and steady state period (ts) and the relatively stable period

[3] txqueuelen and backlog are queues between the bottom of the kernel and the NIC for the send and receive queues respectively.

as the steady state (s). The BE UDP traffic doesn't back off in the event of loss and in this test, the BE class is always oversubscribed. Therefore, whenever AF underutilizes its allocated bandwidth, (i.e. AF-s < 100%) BE-s shows a proportional gain (BE-s > 100%). The difference (AF-ts) – (AF-s) gives an indication of the convergence speed. In fact, it shows how much of the steady state utilisation is wasted due to the transient phase. The smaller this difference is, the higher the utilization is during the transient phase.

Table 1 summarizes the results obtained. The table shows for a given TCP stack, the results obtained for different bandwidth allocations. The last column shows the results gathered under the clamping effect which is explained and results analysed in section 7.1.3.

	10-90	30-70	50-50	70-30	90-10	Clamping for 90-10
Scalable	AF-ts=100 AF-s=100 BE-s=100	AF-ts=99 AF-s=100 BE-s=100	AF-ts=89.4 AF-s=90.8 BE-s=109.3	AF-ts=81 AF-s=80.4 BE-s=146.6	AF-ts=72.99 AF-s=84.4 BE-s=258.6	AF-ts=96.5 AF-s=94.7 BE-s=105
Standard	AF-ts=100 AF-s=100 BE-s=100	AF-ts=100 AF-s=100 BE-s=100	AF-ts=99.53 AF-s=99.7 BE-s=100	AF-ts=99.1 AF-s=99.6 BE-s=100	AF-ts=94.15 AF-s=96.5 BE-s=132.1	AF-ts=97.5 AF-s=96.3 BE-s=102
Hs-TCP	AF-ts=100 AF-s=100 BE-s=100	AF-ts=100 AF-s=100 BE-s=100	AF-ts=98.5 AF-s=99.2 BE-s=101.4	AF-ts=93.1 AF-s=96 BE-s=107.7	AF-ts=83.7 AF-s=97.9 BE-s=131.2	AF-ts=97.2 AF-s=96.1 BE-s=102

Table 1. The utilisation of AF and BE for different bandwith allocations.

Figures 7, 8 and 9 represent a visual summary of the results in the table. AF-s and BE-s QEF are plotted against the different AF bandwidth allocations in Figure 7 and 8 respectively. The difference between the AF-s utilisation during the steady state and the AF-st utilisation during the steady and the transient state is plotted against the AF bandwidth allocations in Figure 9. This shows an indication of the speed of convergence to the steady state phase.

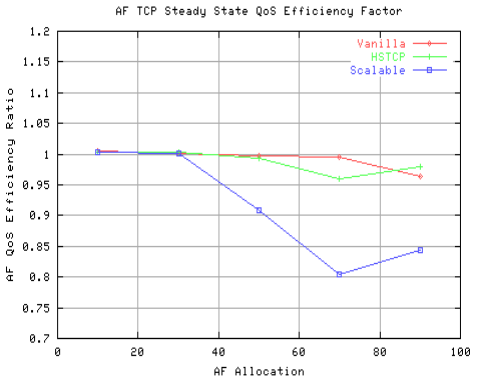

Figure 7 AF QEF in steady state (s)

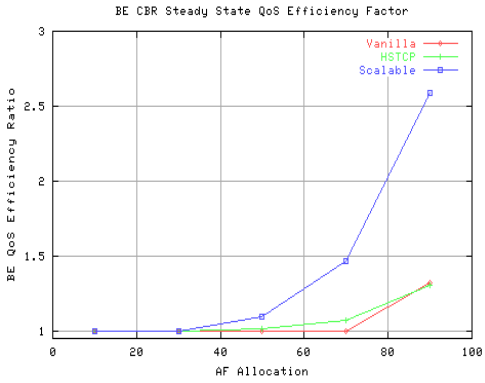

Figure 8 BE QEF in steady state (s)

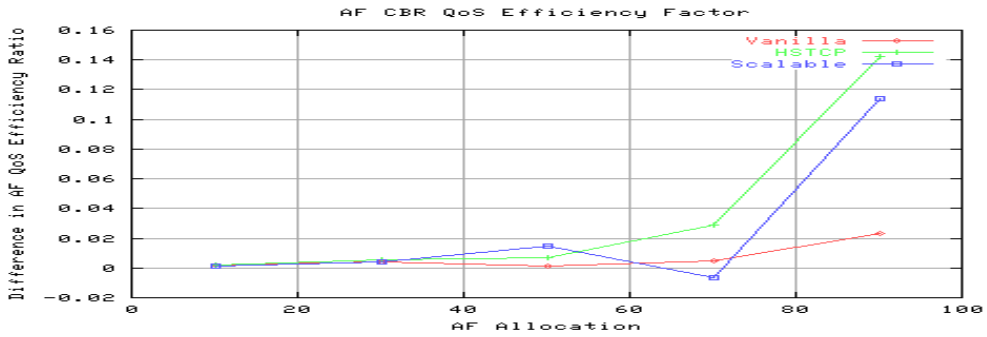

Figure 9. The difference between AF QEF in steady state (s) and AF QEF in steady and transient state (ts)

Looking at Figure 7, the QEF for standard (Vanilla)TCP is always close to 1. Compared to Figure 2 where the achieved throughput is always poor (the utilization there can be calculated as *achieved_throughput*/*available_bandwidth* where 1000-x_i is the amount of available bandwidth. x_i being the amount of CBR offered load), the improvement Standard TCP achieves when in an IP-QOS protected environment is dramatic.

From Figure 7 and Figure 9, Scalable TCP achieves the worst performance while Standard TCP surprisingly shows the best one. The utilization of HS-TCP is closer to Standard TCP than to Scalable. Standard TCP is also, by far, the quickest protocol to reach the steady state. The three protocols showed the same performances when 100 Mbps and 300 Mbps was the bandwidth allocated to TCP in the AF class. This result is expected as a small AF bandwidth allocation means a small CWND which in turn means that although some protocol may have a more aggressive congestion avoidance gradient, there are so few packets in flight that a major multiple drop in the same CWND is avoided. Thus when the bandwidth allocated to the TCP flow is small (below 300 Mbit/s), all the stacks are expected to perform in the same

way. Also, small bandwidth allocation means small bandwidth-delay products and is notorious its effect of normalization to Standard TCP of the different TCP variation protocols.

7.2 In-depth study of the protocols dynamics

The objective of the remaining part of the paper is to investigate in more depth, the way Standard, and in part HS-TCP, maintains the throughput whereas Scalable TCP deeply oscillates. For this reason the typical behaviour of each protocol is presented in a series of figures (Figure10 to Figure15) where the CWND in time is shown along with the associated throughput performance. CWND and slow-start threshold (ssthresh) [**steven**] are, respectively the darker and the lighter curve in the CWND plots whereas the BE and the AF throughput are, respectively, the darker and lighter curves in the throughput plots.
Web100 [**Web100**] is the tool used to print out of the kernel the values of CWND and ssthresh while the received throughput is taken from the output of iperf [**iperf**].

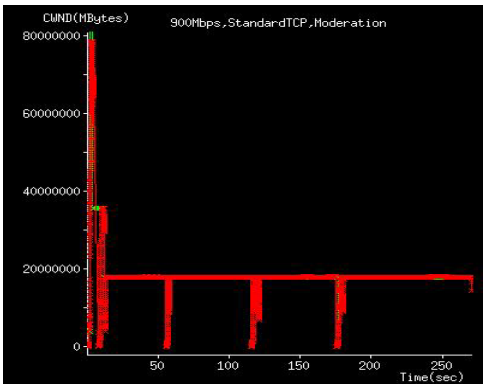

Figure 10 Standard TCP, 900Mbps, CWND(darker)[4]

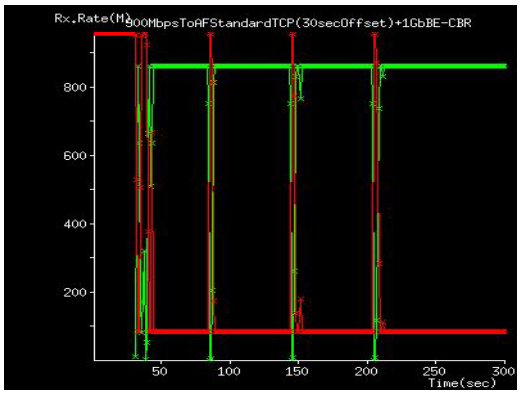

Figure11 Standard TCP,900Mbps BE(darker)AF(lighter) Rx. Throughput

[4] Note that the CWND in Fig 10 is actually (slowly) growing as expected from Standard TCP

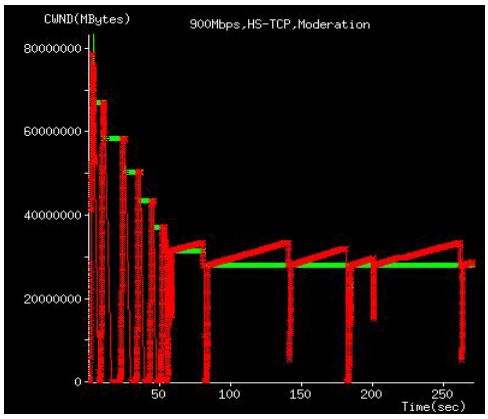

Figure 12 HS-TCP,900Mbps,CWND(darker)

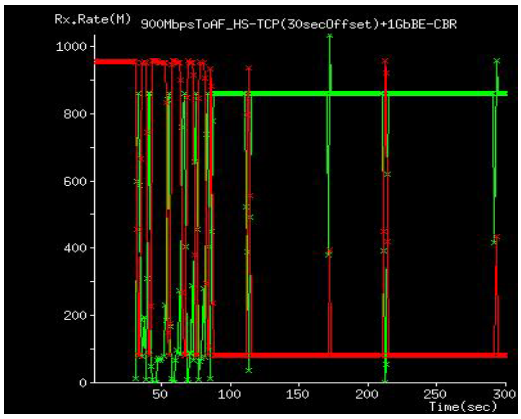

Figure 13 HS-TCP,900Mbps,BE(darker)AF(lighter) Rx. Throughput

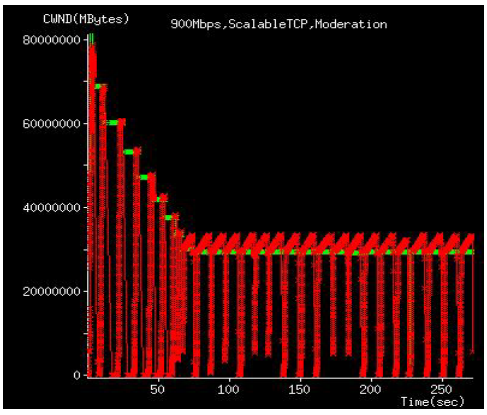

Figure 14 Scalable TCP, 900Mbps, CWND(darker)

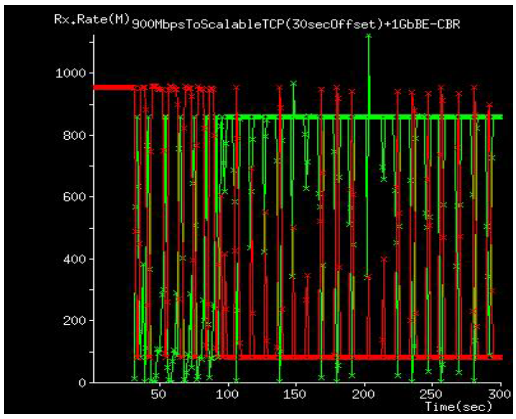

Figure 15 Scalable TCP,900Mbps,BE(darker)AF(lighter) Rx. Throughput

Comparing Figure 10 (standard TCP), Figure 12 (HS-TCP) and Figure 14 (Scalable TCP), The main difference between the three is the high frequency with which Scalable TCP re-enters the slow start phase.

Nearly every time Scalable experiences some packet drop there is a period of silence in web100. The typical dynamic in time of a period of silence for a CWND (the darker line) and ssthresh (the lighter line) plot is shown in Figure 16. As clear from such Figure, Web100 doesn't print out of the kernel any values from the 86.3th to the 86.4th sec of the test, this causing CWND to enter slow start at the 86.4th sec. as if the sender host reset all its TCP variables. According to the theory, the reason for entering the slow start phase can be the expiration of a timeouts but not as many timeouts were actually observed.

In a fat long pipe it is highly recommended to employ the Selective Acknowledgement (SACK) [**Stevens**] option in the TCP stack in order to cope with a major multiple drop occurring in the same CWND but it is clear that the SACK processing in the sender can be an issue depending on the magnitude of such multiple drops. We believe the deep oscillations experienced by Scalable TCP and partially HS-TCP and Standard TCP, are mainly due to the high frequency of the SACKs arriving whose effect is that of temporarily stalling the sender which causes the periods of silence as in Figure 16. This can have either a direct effect such as entering slow-start (as if a timeout occurred) or an indirect detrimental effect on part of the TCP code as discussed in section 8. Our belief in the above statement is reinforced by the studies in sections 8 and 9. As a matter of fact, all our TCP stacks already implement a state of the art SACK patch [**SACKKellypatch**].

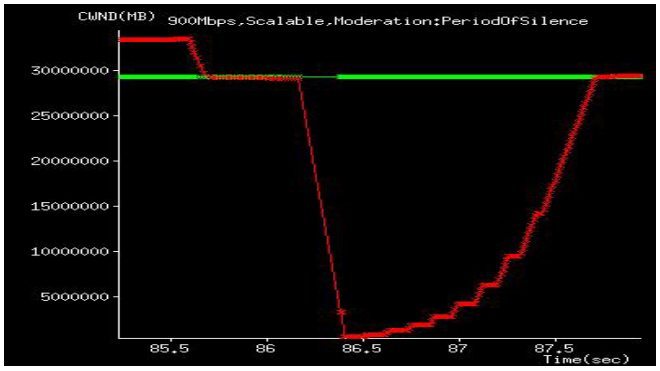

Figure 16: zoom on period of silence experienced from all the TCPs CWNDs (darker) but with different magnitude

7.1.2 Clamping

This section focuses on the results presented in the last column of Table 1. This shows very good performance for all the three TCP stacks.

TCP receiver and sender are tuned in such a way that the maximum CWND is limited by the hosts memory buffers and not by the smallest capacity along the network path. The outcome of this tuning is that both CWND and throughput clamp at (do not grow beyond) the "tuned" values without experiencing any major oscillation. The reason why all three protocols perform in the same way is that drops never occur since the tuning sets the maximum CWND below the bandwidth-delay product. The bandwidth is the 900 Mbps allocated to AF

and the delay is the RTT experienced by the transfer. The limitation of this method is that, it only applies if the bandwidth at which it clamps is guaranteed along the path (see QoS) and if the RTT does not vary too much - the latter being trickier to guarantee.

The clamping effect is obtained by configuring 50 MBytes of maximum socket buffer size instead of the test default value of 100 MBytes.
The typical CWND (the darker line) and its relative Throughput performance (the lighter line) in time for an allocation of 900 Mbps are presented in Figure 17 and Figure 18 respectively.

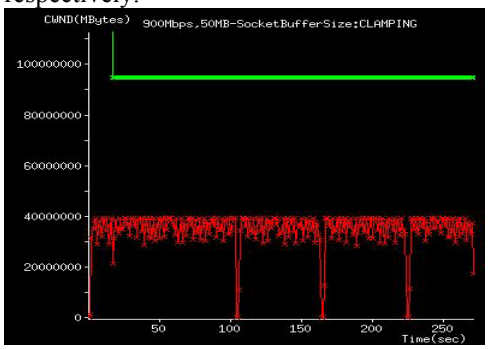

Figure 17 Clamping: 900Mbps, CWND(darker) Vs. time

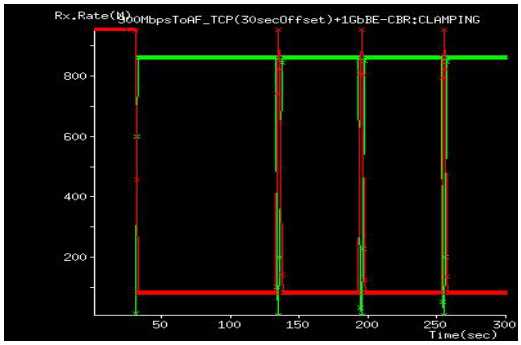

Figure 18 Clamping: BE(darker)AF(lighter)Rx. Throughput Vs. time

From these observations, we can say that if a network path is bandwidth protected (IP-QoS for instance) and reasonably stable in RTT, then a "clamped" transport is beneficial. A further development could be the removal of the bandwidth and RTT constraints by the use of an automated maximum CWND tuning. This tuning, as a function of the modified network conditions, must guarantee that the maximum CWND is always just below the current bandwidth-delay product.

8 Congestion Moderation

The rationale behind the decision of turning "Congestion Moderation"[5] [**CongMod**] off is that all protocols experienced—although with very different magnitudes—"unusual" slow start periods. Unusual since the frequency of the slow-starts greatly outnumbered the occurrences of timeouts. The only possible cause is "Congestion Moderation", a non-IETF

[5] Congestion moderation sets the CWND to be equal to the number of packets in flight (which is the number of packets sent and not acknowledged yet) upon the reception of a "dubious" ack (typically an ack acknowledging more than 3 packets as an indication of something bad happening).

standard addition in the Linux Kernel. What follows are the results obtained by switching the "Congestion Moderation" off.

8.1 Standard TCP

For standard TCP, there is no difference in the throughput performance with or without Congestion Moderation throughout all the AF bandwidth allocations[6].
There is also no significant difference in the CWND dynamic with or without congestion moderation for all the AF allocations but 900 Mbps. This suggests that the lower the allocated bandwidth the lower the probability of dubious (see footnote 4) events. This can be related with the SACK issue since the lower the bandwidth allocated, the lower the bandwidth-delay product and the smaller the magnitude of a multiple drop in the same CWND. This alleviates the SACK processing issue.

Congestion moderation turned off allows the CWND to grow linearly (see Figure 21) without frequent reductions as observed in Figure 19 (This plot is actually a zoom of Figure 10). Both CWND dynamics results in similar overall throughput performances (see Figure 20 and Figure 22). However the CWND dynamic without Congestion Moderation is smoother, has a wider range of values and has a larger absolute values compared to that with congestion moderation on. We are not in a position to say which is better.

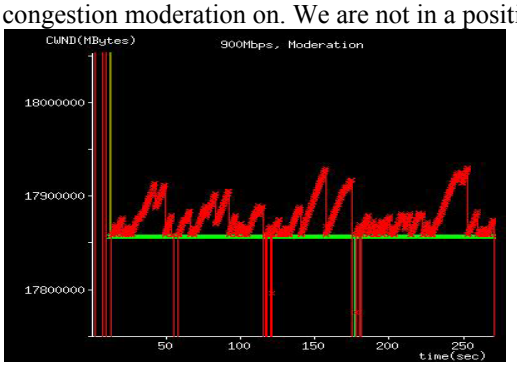

Figure 19 900Mbps,Moderation, CWND(darker) Vs Time zoom

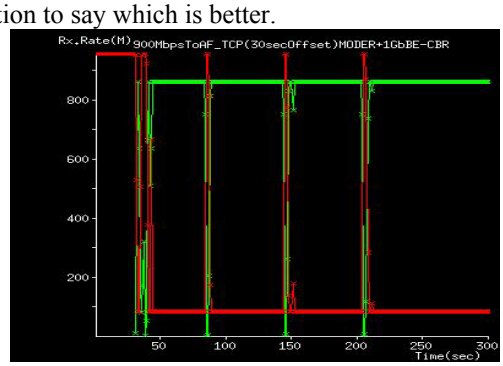

Figure 20 900Mbps,Moderation,BE(darker)AF(lighter) Throughput Vs Time

[6] The few drops experienced in throughput/CWND are only due to UNDOs [**Undo**]. During UNDOs, the ssthresh and cwnd reduce as a loss is thought to have happened, but the subsequent acknowledgment of receipt makes both cwnd and ss-tresh to reset to the previous value.

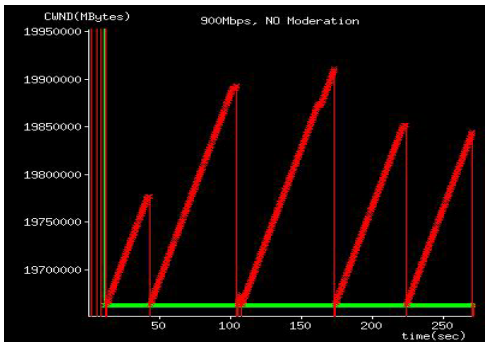

Figure 21 900Mbps,NOModeration, CWND(darker) Vs Time zoom

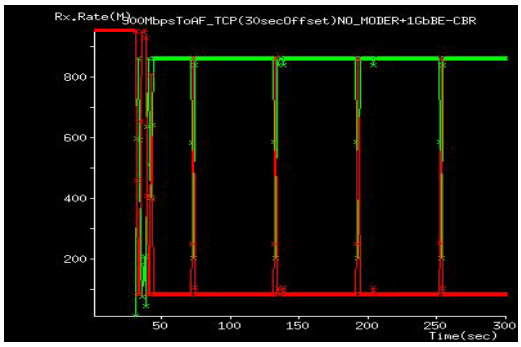

Figure 22 90%, NO Moderation,BE(darker)AF(lighter)Rx.Throughput Vs Time

8.2 HS-TCP

Turning Congestion Moderation off, HS-TCP shows an appreciable difference in throughput performance for the 700 and 900 bandwidth allocations as the probability of dubious events is higher as discussed in section 8.1.

Congestion moderation off is more beneficial for a 700 Mbps bandwidth allocation, whereas the problem is only partially solved for a 900 Mbps allocation.

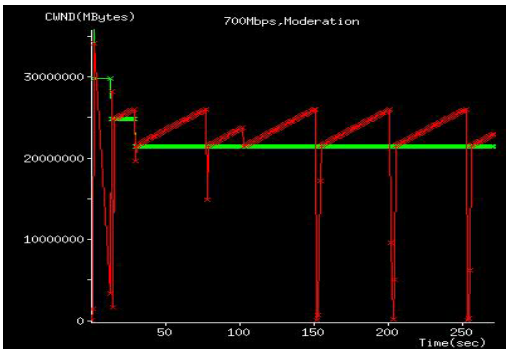

Figure 23 700Mbps,Moderation,CWND(darker) Vs. Time

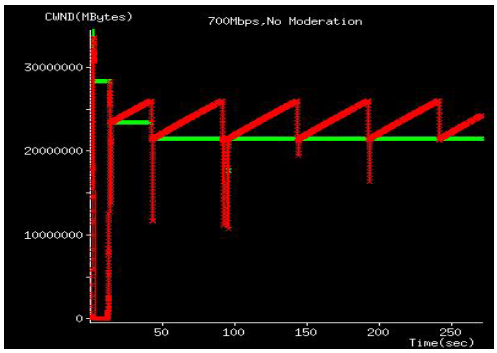

Figure 24 700Mbps,NOModeration,CWND(darker) Vs. Time

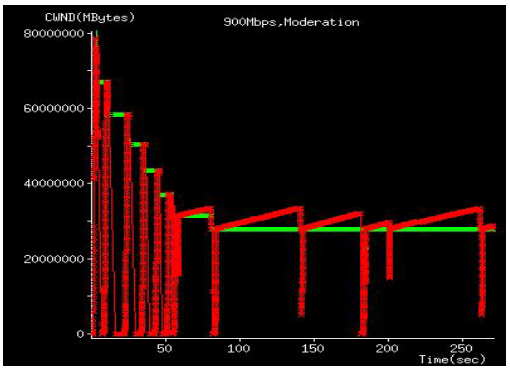

Figure 25
900Mbps,Moderation,CWND(darker) Vs Time

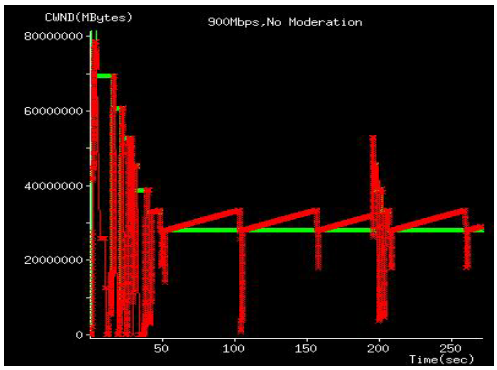

Figure 26
900Mbps,NOModeration,CWND(darker) Vs Time

8.3 Scalable TCP

What is shown in Figure 27 and Figure 28 are the CWND dynamics of Scalable TCP with and without Congestion Moderation when the AF flow is allocated 700 Mbps out of the interface capacity.

The improvement gained without Congestion Moderation is dramatic based on the frequency of slow-starts observed.

The CWND behaviour with and without moderation when AF is allocated 900 Mbps is shown in Figure 29 and Figure 30.

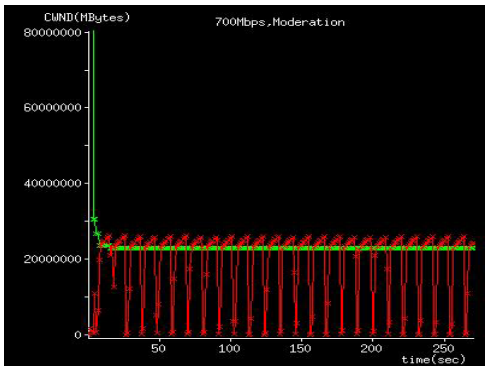

Figure 27
700Mbps,Moderation,CWND(darker) Vs Time

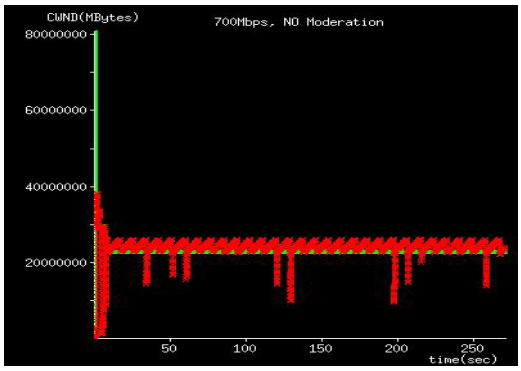

Figure 28
700Mbps,NOModeration,CWND(darker) Vs Time

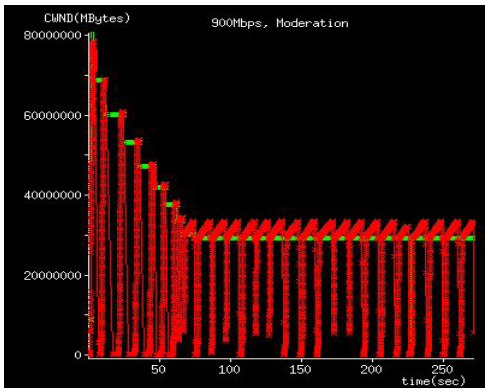

Figure 29
900Mbps,Moderation,CWND(darker) Vs Time

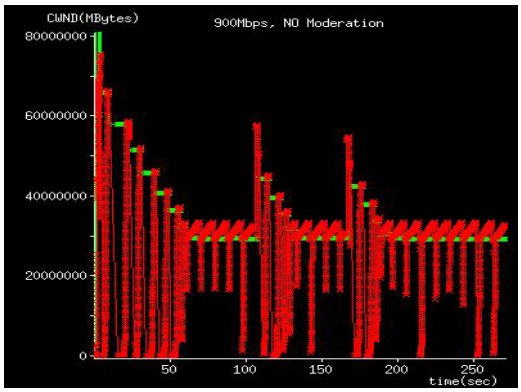

Figure 30
900Mbps,NOModeration,CWND(darker) Vs Time

The main difference between Scalable TCP and HS-TCP (see Figure 23 to Figure 26) is that the occurrence of the slow start events for HS-TCP is much lower, making the effects of the Congestion Moderation much less appreciable. In this, HS-TCP CWND dynamic is much closer to Standard TCP rather than to Scalable TCP.

Having Congestion Moderation turned off has less overall effect on the CWND dynamics when AF is allocated 900 Mbps. The indication here is that above 900 Mbps the dominant effect on the CWND is the SACK processing, causing frequent periods of silence which lead to slow-start.

Figure 31 shows a magnification of a typical period of silence present in Figure 30. These periods of silence are not caused by the congestion moderation code as it is switched off. They are a direct consequence of the SACK processing in the sender.

We suspect that the frequency of SACKs determines the CWND dynamics. When the AF allocation is 700 Mbps, the frequency of SACKs causing the CWND to enter slow-start is enhanced by the Congestion Moderation. In fact to such an extent that removing Congestion Moderation completely resolves CWND entering slow-start. When the AF allocation is 900 Mbps, the frequency of SACKs is so high that removing Congestion Moderation only partially resolves CWND entering slow-start.

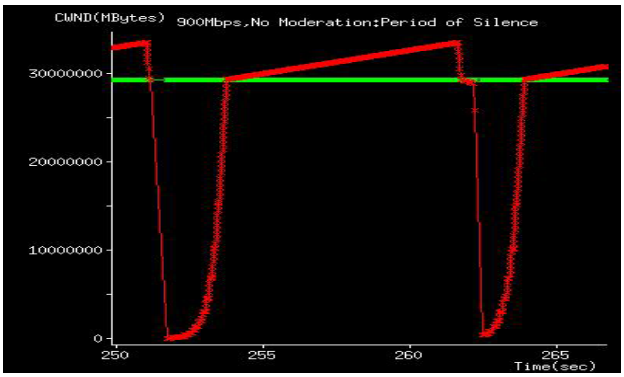

Figure 31: Typical remaining periods of silence due to the sack processing itself. A magnification of Figure 30.

In the next section, we try to alleviate the aforementioned problem by mitigating the effect of sack processing using an active queue management solution. The objective is to lower the rate of SACK events after multiple drops.

9 Active Queue Management solution

Neither Standard TCP nor HS-TCP experienced as many slow-starts as Scalable. This is the reason why an AQM solution is tried for Scalable TCP only and for the most challenging allocation of 900Mbps.

The aim of these tests is to smooth out such oscillations by means of configuring different Weighted Random Early Detection [**Qos**] drop profiles for the AF class. This solution is based on a flow-level interpretation (see equation 1) where the oscillations are the effect of protocol instability. This being the case, a gentle (small ρ in the equation 1) WRED drop profile should be able to validate the inequality expression governing the stability condition for the TCP flow [**Low**].

From a packet-level perspective, the justification for using WRED is based on a our belief that a smoother distribution of the loss-pattern in the AF queue helps lower the burst size of SACKS drops and therefore it will help avoid deep stalls in the sender host.

To accomplish to this task, three WRED drop profiles are configured and shown in fig. 32 below:

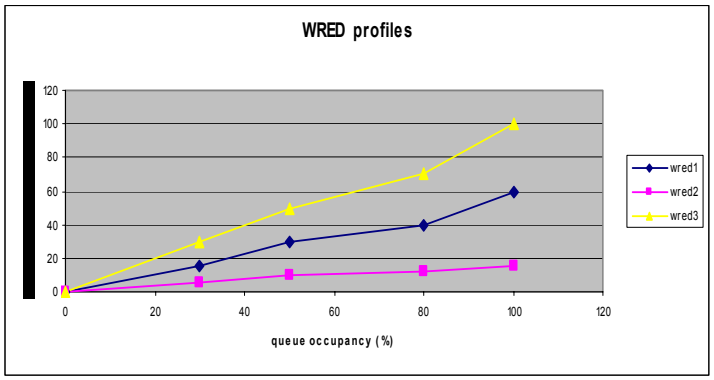

Fig. 32: WRED drop profiles

The rationale is that of trying three different drop profiles where the corresponding slope ρi is such that ρ2<ρ1<ρ3. Congestion Moderation is kept off during the test since it improves the performance as discussed above.

The improvement when ρ2 (the most gentle profile) is employed is dramatic as shown in Figure 33 and 34. This result validates both the flow-level approach which requires a gentle ρ and the packet-level approach which suggests to gently redistributing the loss pattern in order to avoid the introduction extra loss and therefore extra SACKs.

This AQM solution for making Scalable TCP perform better under QOS is important regardless of how well future implementations of the SACK code perform. It will surely help in relaxing the efficiency requirements of such code.

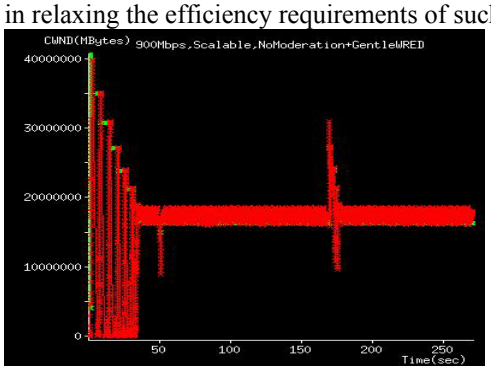

Figure33 Scalable TCP with No Moderation and gentle WRED: CWND(darker) Vs. Time

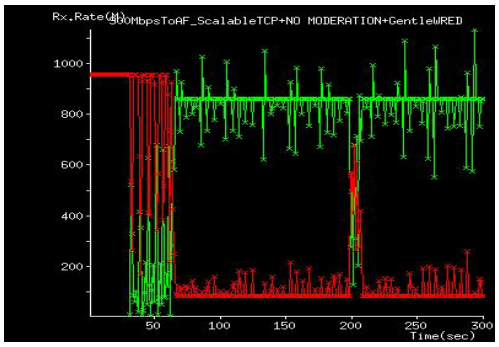

Figure 34: Scalable TCP with No Moderation and gentle WRED: AF(lighter) and BE(darker) Throughput Vs. Time

The successful use of WRED validates the interpretation that the SACK processing is dominant in causing directly or indirectly deep oscillations.

10 Conclusions

We highlighted that the whole idea of a trans-continental high performance GRID heavily relies on the network performance of the current state of the TCP and we demonstrated that, as it is, such a protocol would invariably fail. Standard TCP is shown to be too weak and performs poorly regardless of the amount of available bandwidth in fat long pipes. This drives the development on new and more reactive TCP proposals.

We then proved that making use of new TCP proposals would only partially solve the problem since the new TCP stacks can affect the performance of traditional BE traffic.

Some form of QoS is therefore needed in order to guarantee the protection of traditional BE traffic against the possible aggression of the new TCP stacks. QoS would also guarantee full utilisation of the pipe reserved and, as a consequence, of the whole link. Moreover, it would guarantee a certain level of service predictability for such GRID transfers which is necessary if a resource reservation system–implemented through the use of middleware software– is to be deployed on top of the network infrastructure.

Surprisingly, Standard TCP not only works under QoS but also shows the best performance, followed by HS-TCP and then, at some distance, Scalable. Standard is able to keep the bandwidth allocated as nicely as Hs-TCP does but in addition to this it shows the best convergence speed towards its steady state regime.
Scalable TCP, instead, has to be tuned properly in order to perform better. Scalable has the steepest congestion avoidance gradient which is an inherent cause of severe multiple drops for those CWND values associated with the high bandwidth-delay product the path presents. Therefore, the more bandwidth allocated to Scalable the bigger the operating ranges of CWND and the higher the SACK rate is which in turn stalls the sender host.

The tendency for CWND to enter slow-start depends on the SACK burst size.
The SACK burst size is proportional to the bandwidth allocated and to the congestion avoidance gradient.
Therefore, for protocols with a steep congestion avoidance gradient like Scalable and for high bandwidth allocated, the SACK processing becomes an issue. Removing part of the code as with Congestion Moderation solves the problem. For even higher bandwidth allocations, a direct network action in order to reduce the SACK burst size is needed.

The network action we chose was an AQM solution (WRED). Specifically, a drop probability in the router AF queue with a gentle gradient ρ has been proven to be extremely effective.
Such result validates the belief that the SACK processing is the dominant reason for having poor performance.

This AQM solution can relax the requirement of any further optimization of the SACK code. However, we believe that the main limitations in using aggressive TCP protocols under fat long IP_QOS-enabled pipes lies in not having enough CPU speed to cope with the major multiple drops in the same CWND that they induce.

In conclusion, in a QoS-enabled high bandwidth delay product path, Standard TCP performs better than the other protocols because it is protected and the CWND grows so slowly that it never hits the CWND limit; therefore real drops are prevented. Standard TCP's capability of promptly recovering from even a single drop is poor by design. A BER (Bit Error Rate) greater than zero always exist in real networks due to the physical layer characteristics. For transfers lasting longer than 400 seconds is likely that a BER event is encountered. In this case we recommend HS-TCP above Standard and Scalable for two reasons. Firstly, it has a good loss recovery dynamic as does Scalable but during steady state it performs in a very similar manner to Standard.

The above results suggest that in an environment where the available bandwidth is greater than 1 Gbit/s the problems are further compounded due to the increase frequency of BER events and the larger range of the CWND values in order to utilize the available bandwidth.

11 Acknoledgments

We would like to thank everybody supporting this work and collaborating at UCL and in the DataTAG project. Special mention to people at UCL: Yee-Ting Li , at Daresbury(UK): Robin Tasker, at Caltech: Silvain Ravot and at CERN: Edoardo Martelli, Paolo Moroni, Olivier Martin and Jean-Philippe Martin Flatin.

Appendix

A1: Juniper priority implementation

The WRR [**Qos**] queue weight ensures the queue is provided a given minimum amount of bandwidth which is proportional to the weight. As long as this minimum has not been served, the queue is said to have a "positive credit". Once this minimum amount is reached, the queue has a "negative credit".
A queue can have either a "high" or a "low" priority. A queue having a "high" priority will be served before any queue having a "low" priority.
For each packet, the WRR algorithm strictly follows this queue service order:

1. High priority, positive credit queues;
2. Low priority, positive credit queues;
3. High priority, negative credit queues;
4. Low priority, negative credit queues.

The following explanation tries to clarify the WRR mechanism.

The positive credit ensures that a given queue is provided a minimum bandwidth according to the configured weight (for both high and low priority queue). On the other hand, negative credit queues are served only if one positive credit queue has not used its whole dedicated bandwidth and no more packets are present in a "positive credited" queue.
The credits are decreased immediately when a packet is sent. They are increased frequently.

References

[**Grid**]The Physiology of the GRID: An Open Grid Services Architecture for Distributed Systems Integration, Ian Foster, Carl Kesselman, Jeffrey M. Nick and Steven Tuecke
[**Babar**] http://www.slac.stanford.edu/BFROOT/
[**Stevens**] Stevens, "TCP/IP Illustrated, Volume 1", July 2004.
[**Floyd**] RFC 3649 – HighSpeed TCP for Large Congestion Windows. Sally Floyd, ICSI,
[**Low**] linear stability of TCP/RED and a scalable control. Steven H. Low, Fernando Paganini, Jiantao Wang, John C. Doyle, May 6, 2003
[**Diffserv**] RFC 2475 – An Architecture for Differentiated Services, S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss. December 1998
[**Hs-tcp**] RFC 3649 – HighSpeed TCP for Large Congestion Windows, Sally Floyd, ICSI,
[**Scalable**] Tom Kelly, Scalable TCP: Improving Performance in High Speed Wide Area Networks, First International Workshop on Protocols for Fast Long-Distance Networks, February 2003
[**Datatag**] http://datatag.web.cern.ch/datatag/
[**Qos**] Grenville Armitage, "Quality of Service in IP networks", April 2000.
[**Juniper**] http://www.juniper.com
[Supermicro] www.supermicro.com/PRODUCT/SUPERServer/SuperServer6022P-6.htm
[**Iperf**] http://dast.nlanr.net/Projects/Iperf/
[**Web100**] http://www.web100.org/
[SACKKellypatch] http://www-lce.eng.cam.ac.uk/~ctk21/code/
[**CongMod**] http://www.hep.ucl.ac.uk/~ytl/tcpip/linux/tcp_moderate_cwnd/
http://article.gmane.org/gmane.ietf.tsvwg/1626
[**Undo**] http://www.cs.helsinki.fi/research/iwtcp/papers/linuxtcp.pdf