An investigation on the relationship of aggregated TCP and UDP traffic

M.Davy and B.V.Ghita

Network Research Group, University of Plymouth, United Kingdom
e-mail: info@network-research-group.org

Abstract

The internet quality of service and reliability is currently managed by the TCP protocol. TCP is responsible for setting up the streams speed in a managed way, and if only a limited bandwidth is available, for slowing down the transmission rate of the streams. However, the internet is changing, no longer the dominant applications are HTTP, FTP or POP, but multimedia streaming. As the far most of them are based on the UDP protocol, which doesn't the control features of TCP, the streams oppose a threat to TCP transmissions which are no longer able to manage the quality of service. This paper investigates the relationship between TCP and UDP traffic, by using a network simulation approach and the analysis of real network trace data to set the fundamentals for a more extensive exploration of the aggregated traffic issues and some solutions that could be developed. This research has been sponsored by France Telecom R&D UK.

Keywords

Aggregated traffic, TCP, UDP, network simulation, trace analysis.

1. Introduction

Since the early development of networks and Internet, most applications have been running on top of TCP due to the quality of service (QoS) it provides. The recent evolution of networking applications towards the delivery of more multimedia content and their ability to use high data rates to deliver content with an in-time delivery preference rather than in-order delivery pushes upwards the usage of UDP as a transport protocol instead of TCP. But lacking the feedback and adaptation TCP implements, UDP tends to use all the bandwidth available on a transmission path, eventually congesting the network and preventing TCP transfers to live normally.

This paper presents the work accomplished in order to investigate the fact itself that a relationship between UDP and TCP transfers exists, and if the answer is affirmative, under which conditions can this effect be demonstrated, and is organised as follows. Section 2 will present an overview of the related research in this domain, trying to give some solutions to the problem of aggregated traffic using adequate queue management, developing new application layer protocols or, more drastically, developing new transport solutions that would be deployed alongside TCP and UDP. Section 3 will describe the network simulation developed to investigate in a controlled environment the apparition of the TCP-UDP aggregate traffic problem and Section 4 will present the methodology and results from the analysis of real traffic trace data to detect if such a problem can be observed on different live networks.

2. Related work

Due to the rising data rates employed by multimedia applications, network nodes responsible for the routing of the traffic like routers have to be able to forward packets faster and better, possibly adapting their queues management has to adapt also to this evolution, the standard management scheme operated on a Round Robin (RR) fashion becoming more difficult to meet, resulting in overgrowing queues. Evolutions of the RR queue management like the Weighted Round Robin algorithm (WRR), attributing a weight to each queue, has been tested in (F de Castro et al 2003) and present better solution, but are not adaptive schemes as they do not depend on the traffic passing through the router interfaces but on the interfaces themselves. The evolutions of active queue management with Random Early Detection (RED) (Floyd and Jacobson 1993), Flow based RED (FRED) (Lin and Morris 1997) allow a better packet dropping scheme more fair towards TCP transfers., as they are based on an algorithm trying to predict that a queue would overgrow and start to drop packets from it before the congestions starts. While RED monitors the queues lengths and discards randomly packets from a queue, it is not able to restrict unresponsive flows such as UDP. FRED however implements preferences in the way it discards packets, trying to drop more packets from unresponsive flows. The implementation of RED-DT is a trial to improve this behaviour by adapting the dropping probability of each queue upon the arrival of each packet (Vukadinović and Trajković 2004).

The easiest way to implement fairness in aggregate transport would probably be to evaluate the impairments TCP transmissions are confronted to while transmitting UDP streams. Several researches in such a direction have been started and a few protocols have been developed for specific purposes. SABUL and later redefined as UDT (Gu *et al* 2003) has been developed on a rate based congestion control basis for high data rates traffic, using UDP to transfer data and TCP feedback messages to provide the application with information to change the UDP sending rate. A similar idea has been developed with TCP Rate Probing Adaptation (TPBA) (Tobe 1999), switching from UDP to TCP in the same transmission to regularly probing the network status while transmitting to adapt the UDP sending rate accordingly.

An alternative, though much heavier would be to develop a new transport protocol. Instead of relying on the existing transport architecture, some researches have been focusing on creating side protocols to TCP and UDP. Among them are the Reliable UDP (RUDP) based on RDP (Partridge and Hinden 1990) and the Datagram Congestion Control Protocol (DCCP) (Kohler *et al* 2006). DCCP implements an unreliable flow of datagrams with acknowledgements, a connection handshake and teardown. The most important feature of DCCP is its evolution capacities, as its congestion control and special features can be changed with the usage of different Congestion Control IDs (CCID). It currently implements two CCIDs, which respectively implements a TCP-like Additive Increase Multiplicative Decrease (AIMD) congestion control (Jacobson 1988), and a TCP friendly rate control (Floyd and Kohler 2006; Floyd *et al* 2006).

3. Network Simulation

In this section we present the simulation results obtained with OPNET MODELER. The simulation has been carried in 4 rounds, implementing two different network topologies under diverse TCP and UDP loads. The simulations started with simple topologies constituted of two single hosts linked to a switch and a server by 10 Base-T or 100 Base-T links, followed by a more complex topology linking sub-networks to a switch and the server by 10 Base-T links. Figures 1 and 2 present the general topologies used.

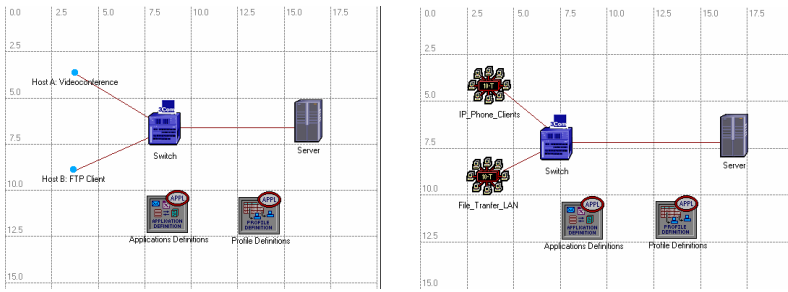

Figure 1: OPNET Topology #1 (left) and topology #2 (right)

The load imposed on the network comprises for all the simulated scenarios a constant FTP transfer as a source of TCP traffic and the UDP traffic was either generated with videoconference or Voice over IP (VoIP) clients following the definitions presented in Table III-1. The projects #1 and #2 are simulated using the topology #1 and the projects #3 and #4 use the topology #2. Project #2 is divided into two sub-projects as at this time in the research, it was easier to change the maximum speed of the links than to modify the default parameters of the applications defined by OPNET to obtain a different scenario exhibiting a load inferior to the bottleneck bandwidth of the network. Project #2-1 uses 10 Base-T links while project #2-2 uses 100 Base-T links.

OPNET Project	Bottleneck bandwidth	TCP Traffic					UDP Traffic				
		Hosts	*Applic°*	*Traffic KB/s*	*Start*	*Durat°*	*Hosts*	*Applic°*	*Traffic*	*Start*	*Durat°*
# 1	10 MB/s	1	FTP	250	0s	300s	1	Video-conference	7.6 MB/s	Every 20s	20s
# 2 - 1	10 MB/s	1	FTP	250	0s	900s	1	Video-conference	7.6 MB/s	20s	60s
# 2 – 2	100 MB/s	1	FTP	250	0s	900s	1	Video-conference	7.6 MB/s	20s	60s
# 3	10 MB/s	10	FTP	575	0s	900s	10	VoIP	16 KB/s	100s	300s
# 4	10 MB/s	10	FTP	575	0s	900s	75	VoIP	990 KB/s	100s	300s

The results exhibited by the simulations vary from one project to the other, but the projects #1, #2 – 1 and #4 exhibited some interaction from UDP transfers on the FTP traffic, whereas the projects #2 – 2 and #3 didn't. With the simple topology used for the project #1, it was already possible to observe the problem of the aggregated UDP – TCP transport as exposed by the traffic patterns in figure 3.

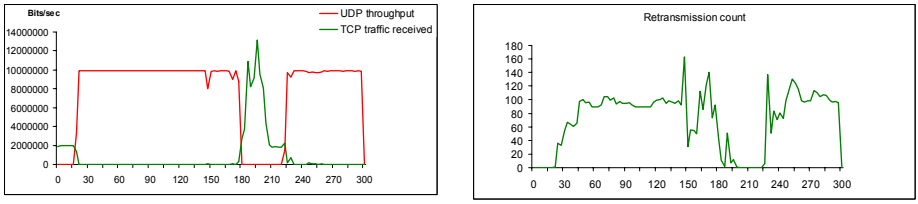

Figure 2: Project #1 traffic patterns (left) and retransmissions (right)

In this simulation round, the TCP traffic is established at 2Mb/s (or 250 KB/s) until when the UDP traffic generated by the videoconference client is triggered at *t=20s*. From this moment until the videoconference stops at *t=180s*, as well as during the second UDP burst after *t=220s*, TCP transfers stop completely and are the subject of a high level of retransmissions as figure 4 shows.

It is important to notice that during this project #1, the bottleneck link is filled up to its maximum bandwidth of 10 Mb/s by the UDP traffic. Similar results have been observed in projects # 2 – 1 and #4, this later one giving a better idea of what can happen on a real network. In fact, the evolution of the project scenarios was elaborated in order to evaluate the possible impact of UDP on TCP starting from small topologies easily encountered on home networks, up to larger topologies closer to a small company network comprising more hosts. Figure 5 present the comparative TCP / UDP traffic of project #4 where it is possible to follow the evolution of the TCP traffic regarding the VoIP calls emitted between *t=100s* and *t=400s*.

Using the same topology, but with 10 VoIP hosts instead of 75, hence generating UDP traffic figures 7.5 times smaller than the 990KB/s generated during the Project #4, the FTP traffic going through the aggregation link of project #3 fails to present any influence of UDP on TCP, as shown by figure 6.

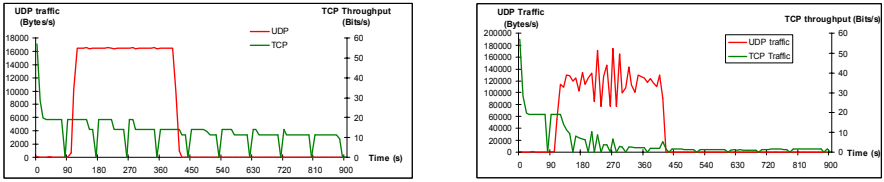

Figure 3: Project #3 (left) and project #4 (right) TCP / UDP traffic patterns

For both these simulation projects, the point-to-point throughput from the server to the aggregating node over the bottleneck link, were used close, or up to the bottleneck bandwidth during UDP transfers. We believe the interaction phenomenon observed between UDP and TCP traffics is due to the fact that for a given network, if the traffic generated by the hosts is not close or greater than the bottleneck bandwidth, the UDP traffic has no influence on the TCP transfers. Additionally, if this UDP traffic tends to be greater than the bottleneck bandwidth, not only the TCP traffic will be affected and will totally back off, but also the UDP transmission themselves will suffer. It can been observed in figure 7 that for the simulated scenarios presenting an influence of UDP over TCP, the overall delay for the real-

time application responsible for generating the UDP traffic was raising dramatically during the duration of the videoconferences or the VoIP calls sessions. For the remaining simulation projects, when no interaction between UDP and TCP was observed, the packet to packet delay during the UDP transmissions didn't vary in the same proportions, if any.

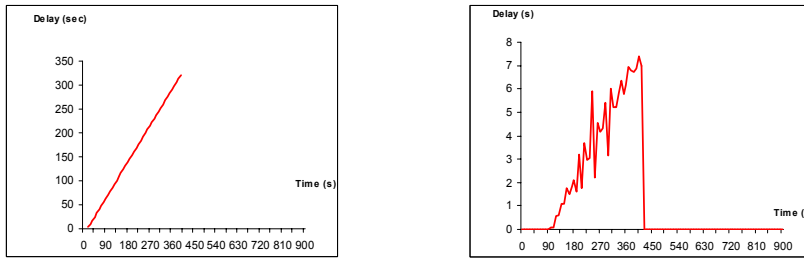

Figure 4: Project #2 Videoconference and Project #4 VoIP packet-to-packet delay

With the simulation of network scenarios implementing different application behaviours, it has been demonstrated in this section that for both small networks and larger topologies likely to be found in real life, the UDP traffic generated by multimedia applications, such as videoconference or VoIP, oppose a threat to the TCP traffic in the case the aggregated bandwidth required to carry both of them is close or greater than the bottleneck bandwidth of the network. Section IV presents the analysis of real traffic trace data aiming to match such behaviour with real traffic.

4. Real Traffic Trace Analysis

In this section, we setup an analysis methodology to monitor the evolution of TCP transmissions in regard to the UDP concurrent traffic present on the same links. While the simulations described in section III propose that the apparition of TCP back off due to unresponsive UDP transfers is triggered by a bandwidth usage close to the bottleneck bandwidth of the network, we investigate here the TCP behaviour in two real traffic data sets.

4.1 Methodology

The analysis of real traffic trace data conducted during this research used off line analysis of trace data captured at aggregation nodes such as a router or a firewall. The methodology required the deployment of a complete process to extract and compute the important parameters from the trace files, including the usage of special tools to protect the identity and the security of the network on which the trace data was captured.

De facto, after collecting trace data from a network node with T*cpdump*, these privacy requirements imply the usage of a program such as *Tcpurify* to scramble the IP addresses found in the collected data and it has been proposed that due to the complexity of some networks and their range of IP addresses, it was better to scramble not only some of the subnets presented by the trace file, but all the IP

addresses. A special encoding scheme has been added to *Tcpurify* in order to renumber all the trace file IP addresses to 0.0.0.0. Doing so might, at first sight, remove the possibility to detect individual flows in the data collected but as the new encoding scheme doesn't scramble the port numbers this identification could still be carried on. In fact, as presented by (Gleason 2001), the random port numbers used by TCP connections to initiate their requests are unlikely to collide during the average duration of TCP connections, leaving the possibility to identify uniquely different TCP connections and their parameters.

The sanitised trace data obtained from the modified version of *Tcpurify* has been further processed by a series of scripts responsible for the generation of the trace statistics, comprising the evolution of the TCP throughput, retransmission count and reported losses. However, due to some software limitations, namely *Tcptrace*, responsible for generating TCP and UDP connections reports, it has not been possible to generate statistics reports with an update period less than a second. In fact, if the generation of the TCP connections timestamps is accurate up to the millisecond, as Tcptrace first focus is not UDP connections, the generation of the UDP parameters reports is subject to much less attentions, leaving the resolution of the timestamp to the interpretation of textual dates instead of the precise UNIX epoch format. This limitation induced a larger amount of work in order to be able to synchronise the TCP and UDP flows datasets extracted from the trace files.

The synchronisation of the two protocols flows information is the critical step in the trace analysis. In fact, as the TCP flows and UDP flows are used by different applications, may have different sources and destinations and last for different amount of time, it is not possible to directly compare both protocols by associating a UDP flow to a TCP connection. Instead, an averaging process has been set up at regular intervals in order to compare the evolution of the previously mentioned TCP parameters regarding the overall UDP throughput encountered in the path. As defined by the lack of capability of *Tcptrace* to present UDP timestamps in a precise fashion, this interval has been reduced to the minimum available after the *Tcptrace* reports generation, e.g. 1 s. The process of synchronisation of the TCP and UDP flows information is considered critical because if not done properly, some important comparison points might be lost by comparing parameters from different times.

4.2 Trace analysis

The data used during this research came from A. a small network constituted of a limited number of hosts at the France Telecom (FT) laboratory, with a bottleneck bandwidth in direction of Internet of. 2Mb/s and B. a large network comprising hundreds of hosts at the University of Plymouth (UoP) using four 150Mb/s connections to connect its backbone its ISP.

To give credit to the methodology presented in section IV. A., it is important to know that this analysis is relevant only if the levels of TCP and UDP usage on the network are sufficient to assume that, would an evolution of TCP parameters be detected, it could be attributed to UDP transmissions and only them. In fact, if other protocols share the same amount of bandwidth than TCP or UDP transfers, it would not be relevant to compare the evolution of those ones only, but would require the

integration of the other protocols too. Fortunately, the distribution of the protocols usage shown in figure 8 and figure 9 is much in favour of TCP and UDP, whose aggregate usage represents at least 95.85% of the traffic encountered at the FT lab and 96.45% for the UoP dataset, allowing us to make the assumption that would an interaction be discovered between UDP and TCP, it would not be dependant on other protocols behaviours.

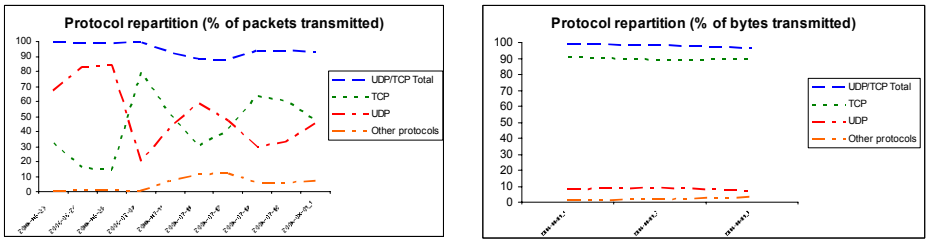

Figure 5: FT (left) and UoP (right) datasets protocol usage distribution

Due to the self adjusting behaviour of TCP, the evolution of TCP throughput in response to a high UDP level on the network would be expected to decrease as the UDP throughput raises. Unfortunately, the figures presented by both datasets for directions incoming to and outgoing from the capture node do not present such a pattern. Figures 10 and 11 present the repartition of the TCP throughput in regard to the UDP throughput, both averaged every second of the capture file as described in section IV. A. It might be noticed that for the FT dataset, the incoming traffic seems to present a reduced TCP throughput for higher UDP throughput values, but we believe this can be attributed simply to a lack of high throughput UDP transmissions rather than a direct influence of UDP on TCP connections.

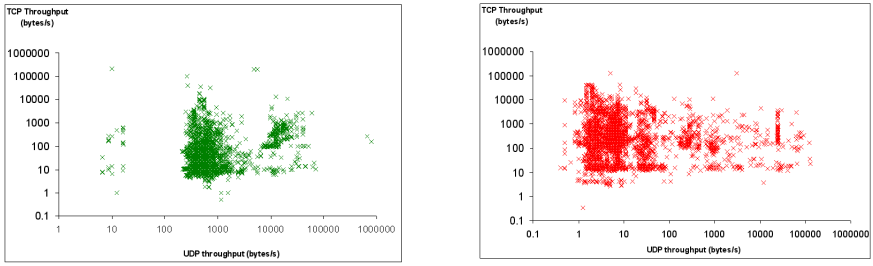

Figure 6: FT traces – Throughput distribution, outgoing (left) and incoming (right) traffic.

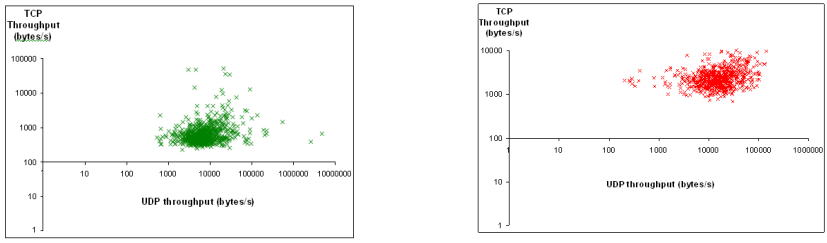

Figure 7: UoP traces – Throughput distribution, outgoing (left) and incoming (right) traffic.

For the UoP dataset, the values obtained appear to be more stable. This is probably due to the higher number of hosts generating some traffic compared to the FT lab, where a single host can have greater influence on the overall throughput figures. The UoP traces even present in the case of the incoming traffic a rising trend towards the higher UDP throughput values, which would be the opposite effect than the expected diminution. However, as indicated previously, the bottleneck bandwidth of the University of Plymouth backbone is much greater than the network load shown in the trace data captured. The rising pattern presented in figure 11 may only be the normal evolution of the traffic for such slow bandwidths. This might come as an element to confirm that for lower levels of usage of the bottleneck link capacity, no interaction of UDP on TCP transfers is observable.

Retransmissions are triggered when packets are not acknowledged, when the TCP retransmission timer expires, or when some packets are missing in the received sequence. The results obtained with both data sets present different patterns. For the FT traces, as for the throughput, it appears that many TCP connections present a higher amount of retransmissions when the UDP traffic is minimal. And despite the fact that 96% of the time, the loss rate for the outgoing traffic remains under 1% and under 0.5% for 90% of the time for the incoming transmissions, no tendency to increase towards the higher UDP throughput has been detected.

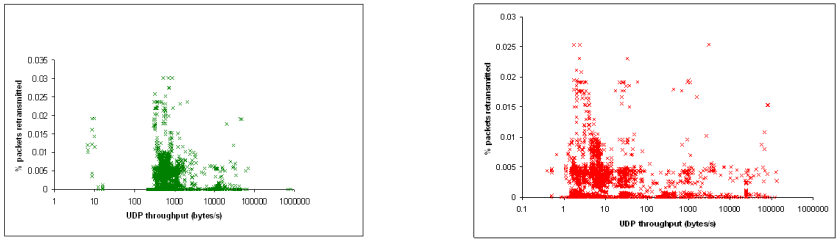

Figure 8: FT traces – TCP Retransmission rate vs. UDP throughput outgoing (left) and incoming (right) traffic.

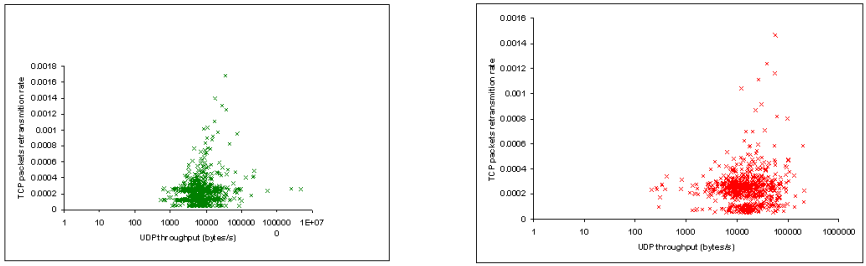

Figure 9: UoP traces – TCP Retransmission rate vs. UDP throughput outgoing (left) and incoming (right) traffic.

However, for the UoP traces, it seems that, for some TCP flows transmitting when UDP throughput is higher than during the rest of the trace data, more retransmissions can be observed in both directions. This may constitute an example in favour of the expected behaviour for the aggregate traffic, but these values represent only 5% of the dataset, 82% of the entries presenting a retransmission rate under 0.03%, even for

the higher UDP bandwidth observed in the trace data, and its impossible to conclude from them.

As for the retransmissions, the expected behaviour of packet loss was an increase as UDP throughput raises. Losses are here reported from Tcptrace statistics which indicate them as post loss acknowledgements (ACKs), e.g. the total number of ACK packets received after losses were detected and a recovered from. If the losses distribution for the FT dataset presents a greater amount of losses when the UDP throughput is lower, the UoP traces show an opposite distribution. The general figures for the outstanding data are again quite low as only 0.4% of the outgoing traffic in the FT dataset are greater than 0.1% and 4% of the incoming traffic present more than 0.5% of losses. For the UoP dataset, 7.6% of the outgoing traffic presents a loss rate inferior to 0.3% while 2% only of the incoming traffic loss is greater than 0.04%.

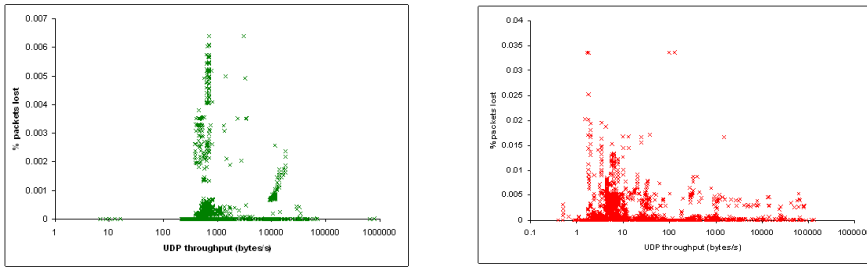

Figure 10: FT traces – TCP Loss rate vs. UDP throughput outgoing (left) and incoming (right) traffic.

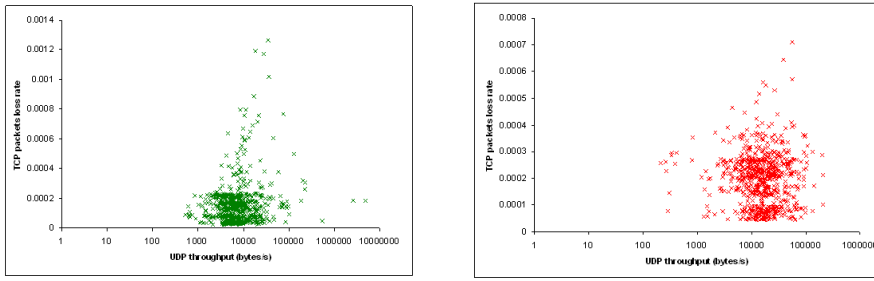

Figure 11: UoP traces – TCP packet loss rate vs. UDP throughput outgoing (left) and incoming (right) traffic.

The really low loss figures presented the UoP trace data, associated to the averaging methodology used to obtain those figures, do not allow us to emit any conclusion in favour of the UDP influence on TCP transmissions, and further investigation, especially for higher values of UDP throughput, need to be conducted.

5. Summary, Limitations and Future Work

In this paper, we have presented the research conducted both with simulation scenarios and the analysis of real trace data in order to detect in which conditions an influence of UDP unresponsive transfers might impact on the TCP traffic. The

simulation scenarios presented that, for a given network, if the aggregated traffic throughput is not close enough or does not exceed the bottleneck capacity of the network, no interference can be detected and both types of traffics can flow without suffering any impairments. However, it has been proposed that in opposite scenarios, TCP connections would start to back off, and eventually starve. In extreme cases, not only TCP transfers would stop, but a dramatically increase of packet to packet delay would be observed in UDP transmissions. However, if this phenomenon has been observed during the simulation rounds, the analysis of two real traffic trace datasets didn't provide any explicit demonstration of this phenomenon. These results have to be tempered by the methodology used for the analysis: in order to obtain data suitable for a cross analysis between the two protocols parameters, an averaging process has been developed, which might hide important information about the connections. As the figures obtained from the trace files used did not present an extensive usage of the network, to validate the hypothesis that effects of UDP can be observed on TCP only if the link usage gets close to the bottleneck bandwidth, the analysis methodology would gain to be revised, possibly implementing live analysis instead of off-line analysis, allowing longer runs and more accurate information to be extracted.

6. References

F. de Castro M., Gaïti D., M'hamed A., Oliveira M., "Comparing Application Performance on Distinct IP Packet Scheduling Configurations",http://www.cefetce.br/Ensino/Professores/mauro/Public/sBC-SBRC_Comparing%20Application_Performance_on_Distinct_IP_Packet_Scheduling_Configurations.pdf, 2003

Floyd S., Jacobson V., "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, 1993

Floyd S., Kohler E., "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control", RFC 4341, March 2006

Floyd S., Kohler E., Padhye J., " Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC)", RFC 4342, March 2006

Gleason M., "The ephemeral port range", NCFTP website, http://ncftp.com/ncftpd/doc/misc/ephemeral_ports.html, 2001

Gu Y., Hong X., Mazzucco M., Grossman R., " SABUL: A High Performance Data Transfer Protocol", submitted for publication, http://www.dataspaceweb.net/papers/sabul-hpdt-03.pdf, 2003

Jacobson V., "Congestion avoidance and control", Proceedings of SIGCOMM '88, ACM, Stanford, CA, August 1988

Kohler E., Handley M., Floyd S., Datagram Congestion Control Protocol (DCCP), RFC 4340, March 2006

Ling D., Morris R., "Dynamics of Random Early Detection", Proceedings of SIGCOMM '97, 1997

Partridge C., Hinden R., "Version 2 of the Reliable Data Protocol (RDP)", RFC 1151, April 1990

Tobe, Y., "A Host Architecture of QoS Control for Continuous Media Stream Communications", PhD thesis, http://www.unl.im.dendai.ac.jp/~yoshito/yoshito-thesis.pdf, 1999

Vukadinović V., Trajković L., "RED with dynamic thresholds for improved fairness", Proceedings of the 2004 ACM symposium on Applied computing, March 2004