DropTail Based ConEx Applied to Video Streaming

Ali Sanhaji, Philippe Niger, Philippe Cadro
Orange Labs
2, avenue Pierre Marzin,
22300 Lannion, France
Email: {ali.sanhaji, philippe.niger, philippe.cadro}@orange.com

André-Luc Beylot
Toulouse Institute of Computer Science Research (IRIT)
INP-ENSEEIHT, Laboratoire IRIT,
2, rue Charles Camichel,
31071 Toulouse Cedex 7, France
Email: andre-luc.beylot@enseeiht.fr

***Abstract*—With Internet traffic ever increasing, network congestion should occur more and more frequently. During congestion periods, some users contribute more than others to the congestion in the network. It might be interesting for a network operator to differentiate between users proportionally to the congestion they induce, but the necessary information for this purpose is not available at the network layer, and is exchanged at the transport layer (e.g., Transmission Control Protocol (TCP) acks). This led the Internet Engineering Task Force (IETF) to design Congestion Exposure (ConEx), a new mechanism to expose to the network the amount of congestion a user is responsible for. However, ConEx needs other mechanisms such as Random Early Detection (RED), Explicit Congestion Notification (ECN) and a number of modifications to the senders and receivers to be fully operational. Nonetheless, it is deployable with few modifications by relying only on loss information in DropTail queues to improve the fairness between users. The aim of this paper is to provide a comparison between the performance in terms of fairness improvement provided by ConEx with few modifications and by ConEx with complete modifications. Firstly, we will see that despite the limited accuracy due to the few changes, ConEx still provides good fairness improvement between users. Secondly, we will discuss the weaknesses ConEx presents with regard to short-lived flows. Finally, we will show how ConEx can help during congestion periods to enhance the Quality of Experience (QoE) of video streaming users (based on a YouTube traffic model).**

***Keywords*-ConEx; ECN; Congestion; policing; YouTube; LEDBAT.**

I. Introduction

During the network's busy hours, a greater amount of traffic than what the network can handle leads to congestion, affecting the quality of experience of many users. Yet, this great amount of traffic is mainly caused by a small percentage of users. For example, in Orange's Fiber To The Home (FTTH) access networks, 80% of downstream traffic is generated by 15% of the customers [1]. The aim is to convince these heavy users to yield network resources during congestion periods for the benefit of everybody.

Some traffic management approaches are already implemented like rate-limiting or defining Data-Volume caps above which the users are slowed down or stopped. However, these solutions show limited efficiency because they do not consider the network state, if it is congested or not, if the rate-limited user has seriously hampered the experience of the others, if this "heavy user" yielded the network resources when encountering congestion. A heavy user might consume his allowed Data-Volume even when the network is not in a congestion phase. It would be fairer to limit the users according to how much congestion they induced. For this, we would need the information about the congestion encountered by the users. This valuable congestion information can be exchanged between the users at the transport layer (e.g., through TCP acks) but it is transparent for the network layer.

To counter this lack of information at the network layer, the IETF designed ConEx, which is a mechanism that allows the sender to inform the network about the congestion encountered [2]. The amount of lost and ECN-marked packets exposed by a user defines a new metric called the **Congestion-Volume**, which is a more useful metric than **Data-Volume** because it reports directly the congestion in the network.

The implementation of ConEx relies on existing mechanisms like RED, ECN capability on routers and new features to both the sender and the receiver to be fully ConEx-capable. We are interested in whether or not ConEx still presents a good performance without the use of ECN and relying only on minimal modifications to the users. In this paper, we will first present in Section II the related work on ConEx. Section III will describe the ConEx principle and the mechanisms on which it relies. The performance evaluation of ConEx with and without ECN using long-lived flows is presented in Section IV while the short-lived flows issue will be discussed in Section V. Our interest will be focused, in Section VI, on how ConEx can be useful in the case of video streaming traffic to enhance the users' QoE, with scenarios using a YouTube traffic model, and how heavy users can take advantage in using a congestion control algorithm like Low Extra Delay Background Transport (LEDBAT). Section VII summarises the contributions, finally, Section VIII discusses the future work, still waiting to be covered.

II. Related Work

The IETF launched a working group to develop experimental specifications of ConEx in IPv6 networks [2]. An Request For Comments (RFC) [3] discussing the concepts and use cases has been published, and other drafts concerning the ConEx mechanism are currently available: the use of a destination option in the IPv6 Header to carry the ConEx markings and the necessary modifications to TCP [4].

Re-ECN is a "pre-ConEx" implementation solution to allow congestion exposure for IPv4 networks. A thorough description and analysis of the Re-ECN mechanism has been done under the Trilogy project [5]. This work had a great influence for the emergence of the ConEx working group.

Some papers focused on the performance evaluation of the congestion exposure mechanism through the evaluation of Re-ECN in multiple scenarios. [6] developed a Linux implementation of Re-ECN and performed several simulations showing the great dependency of the Re-ECN information to the flow size, the Round Trip Time (RTT) and the Active Queue Management (AQM) parameters. [7] evaluates mobility issues with congestion exposure and shows that mobility is not a major concern for Re-ECN. [8] evaluates Re-ECN applicability in LTE networks and found that it can bring a significant improvement for these networks unless they are under severe packet loss rate. All these papers rely on the use of ECN to signal congestion; to our knowledge, no performance evaluation of ConEx has been made solely based on loss exposure.

III. Congestion Exposure

In this section, we will describe ConEx, how it operates to expose congestion, along with the other mechanisms used to collect congestion information and control the users' traffic.

A. *ConEx mechanism*

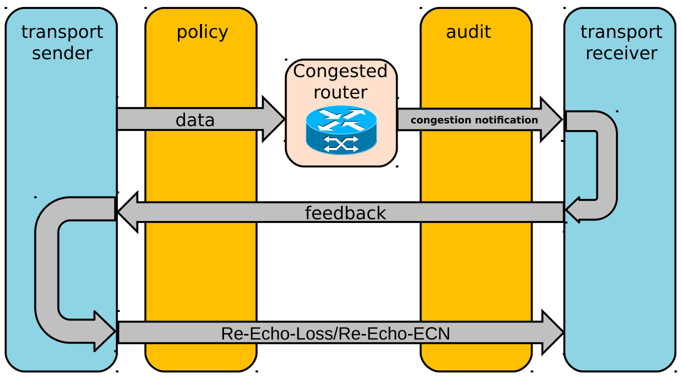

Figure 1. ConEx mechanism

Figure 1 shows the whole ConEx process and all the elements involved with it. The ConEx mechanism works as follows: A transport sender starts by sending a data packet in the network, this packet might encounter one or several congested routers along its path. The packet will either be lost or ECN marked (by setting the Congestion Experienced (CE) codepoint in the IP header [10]) by the congested routers. This information about loss or marking will reach the transport receiver, and through the TCP acknowledgments, the receiver will feedback this information to the sender. With the use of ConEx, the sender will reinject this feedback to the network in the IP packet headers (e.g., use of the RE bit in section III-D), which will hold the Re-Echo signals. Detecting a loss will generate a Re-Echo-Loss signal from the sender, while an ECN marked packet will generate a Re-Echo-ECN signal.

The information provided by ConEx can then be used by the network operator for traffic management through a congestion policer for example. At the ingress of the network, a congestion policer counts the congested packets and takes traffic control policy decisions (e.g., discard, deprioritize packets using Differentiated Services (DiffServ)) if the user has consumed the congestion-volume he was allowed. At the egress of the network, an auditor makes sure that the senders are exposing the right amount of congestion in the network. It helps as a prevention from the users understating the congestion their flows encounter, but if the sources are trusted ones, the auditor is unnecessary.

B. *Random Early Detection*

Random Early Detection is an Active Queue Management technique, implemented on many routers, which was first introduced in [9]. It allows to randomly drop or ECN mark packets according to a probability which increases from 0 to the maximum probability p_{max} when the mean queue length increases from a minimum threshold to a maximum threshold. Above the maximum threshold, all packets are either dropped or marked if ECN is used.

C. *Explicit Congestion Notification*

Explicit Congestion Notification [10] is a way to indicate the occurrence of congestion in the network without having to drop packets. It uses two bits [ECT,CE] of the IP header to signal congestion to the receiver.

D. *Re-ECN*

Re-ECN is a candidate implementation of ConEx for IPv4 [5]. It uses the bit 48 (RE bit) of the IPv4 header to extend the ECN field to a 3-bit field, allowing 8 codepoints. These codepoints identify the ConEx signals as described in Table I.

TABLE I. ConEx signals with Re-ECN encoding

ECN field	RE bit	ConEx signal
00	1	Credit (Used with the auditor)
01	1	ConEx-Not-Marked (ConEx-Capable)
01	0	Re-Echo-ECN or Re-Echo-Loss
11	1	ECN marked packet
11	0	Re-Echo packet and ECN-marked
10	0	ECN legacy (Not-ConEx)
00	0	Not-ECN (Not-ConEx)
10	1	Unused

E. *TCP modifications*

The classic ECN mechanism as described in [10] allows the receiver to feedback only one CE mark per RTT. Indeed, even if several packets of the same flow get CE marked during one RTT, the receiver has only one bit (ECN-Echo (ECE) flag in the TCP header) to feedback all the marks. The information about how many packets have been marked is valuable for ConEx but also for other mechanisms like DCTCP [11]; modifications to TCP are needed to provide more than one feedback per RTT. [12] proposes a solution to achieve such a goal. It suggests to overload the three TCP flags ECE, Congestion Window Reduced (CWR) and Nonce Sum (NS) to form a 3-bit field. This field would act as a counter for the number of CE marks seen by the receiver which can feedback it to the sender, allowing the sender to follow the accurate evolution of ECN markings and report the right amount of Re-Echo-ECN signals.

F. *Congestion policer*

The great advantage with ConEx is to allow the network operator to police the users proportionally to their contribution to congestion, thus to the impact they have on other users. The policing can be applied at the ingress to prevent the heavy users from overloading the network. The congestion policer can be implemented as a token bucket with a filling rate r (the allowed Congestion-Rate) and a depth d (the allowed Congestion-Burst). The policer removes the same amount of tokens from the bucket as there are bytes in the Re-Echo-ECN/Re-Echo-Loss packets sent by a user. When the bucket empties, the policer proceeds to discard the packets of the user who exceeded his allowed Congestion-Volume. As shown in Figure 2, there are three levels of policing used in the

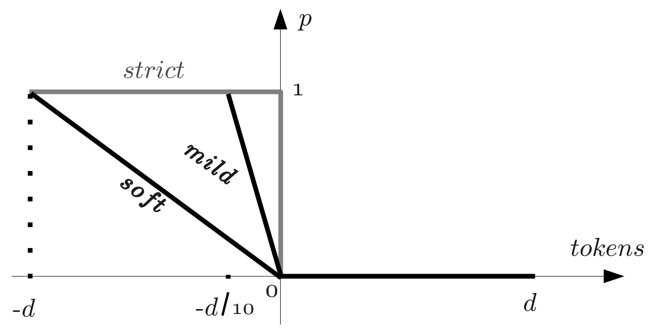

Figure 2. Drop function of the congestion policer

performance evaluation, the strict policer discards all packets when the bucket is empty, the mild policer discards packets with a probability increasing from 0 to 1 when the bucket depth decreases from 0 to $-d/10$ and the soft policer discards packets with a probability increasing from 0 to 1 when the bucket depth decreases from 0 to $-d$.

IV. Long-lived flows

A. Simulated Network

To perform the simulations, we used the Network Simulator 2 (NS2) [13] in which we implemented ConEx following the latest RFCs and drafts and we used the IPv4 proposal presented in Section III-D. The simulated network is depicted in Figure 3. There are 100 users on either side of the network, each single user on the right receiving traffic from a single user on the left. 90 of them are light users using only one File Transfer Protocol (FTP) flow each as a traffic source. The other 10 users are heavy users, they use 36 FTP flows each as a traffic source, they will thereby be responsible for 80% of the traffic on the bottleneck. The TCP senders use cubic as a congestion control algorithm with Selective Acknowledgments (SACK) and TimeStamps options. The TCP receivers can feedback ECN markings in a accurate count to the sender which in turn will send a Re-Echo-ECN/Re-Echo-Loss signal for every ECN-marked/lost packet. The TCP maximum window value is equal to 64KB while the packet size is equal to 1500 bytes.

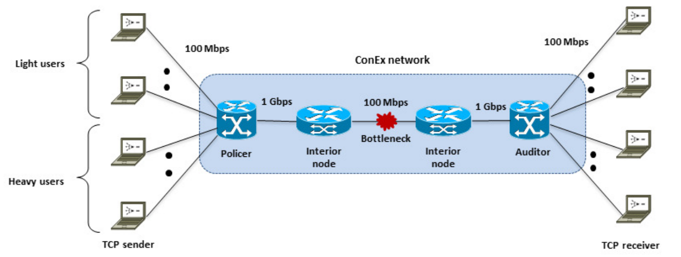

Figure 3. Simulated network topology

All users have 100ms Round Trip Time and share a 100Mbps bottleneck. At the ingress of the network, there is a per user congestion policer, which is implemented as described in Section III-F. The action taken by the policer is dropping the user's packets when the bucket, which has a depth of 64KB, is emptied. On the bottleneck's router, there is a RED queue with a length equal to the Bandwidth Delay Product (BDP) in order to hold 100ms of the bottleneck's packets. The probability of marking packets increases from 0 to $p_{max} = 1$ as the average queue length increases from 10% to 100% of the total queue length. At the egress of the network, there is an auditor which is deactivated because we use trusted sources.

A single simulation lasts 100s and is run 30 times to have proper 95% confidence intervals for each point. For greater visibility of the graphs, these intervals are not depicted when their value is around 1% of the metric's mean. The traffic sources are saturated and each flow starts randomly between 0 and 300ms.

TCP provides a flow-based fairness, meaning that a user can get more bandwidth share if he uses more flows. The per user congestion policer does not consider the user's flows individually but only the aggregate traffic of the user to monitor the amount of congestion induced in the network. The purpose of ConEx is to improve fairness between users, especially between the light user and the heavy user. Therefore, we will be monitoring a metric defined in [14]:

$$unfairness = \frac{throughput\ of\ a\ heavy\ user}{throughput\ of\ a\ light\ user} \tag{1}$$

Through the action of the policer, ConEx provides the ability to decrease the unfairness between users in a congested network. In the following sections, we will discuss the impact of the filling rate and the harshness of the congestion policing. Afterwards, we will compare the performance of ConEx when all the modifications are applied with the performance of ConEx when only the minimum modifications are applied.

B. Policer harshness

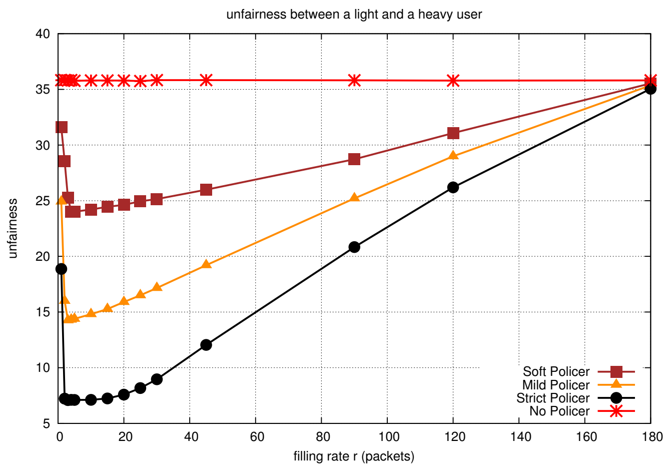

Figure 4. Unfairness between a heavy and a light user

Figure 4 represents the average unfairness versus the allowed filling rate of a user in the simulation. Each curve represents a level of harshness of the policer as explained in Section III-F. The straight red curve on top is the unfairness when no policing is applied (the policer is deactivated). Only TCP is performing congestion control and TCP induces fairness between flows; as a heavy user has 36 flows and a light user has only one, the unfairness is equal to 36 as expected. When the policer is activated (the three remaining curves), the heavy users are the ones that will be the most policed. As the heavy users are forced to reduce their throughput, the light users occupy the freed bandwidth and the unfairness is reduced.

In Figure 4, the unfairness presents a minimum value suggesting an optimal filling rate. On the two sides of the optimum, the unfairness increases but for two different reasons. On the right side, as the filling rate increases, the heavy users undergo less policing. They get a higher throughput than with the optimal filling rate and the unfairness increases. When the filling rate is high enough, the heavy users avoid the policer's intervention, so the unfairness reaches the value obtained without policing ($unfairness = 36$). On the left side of the optimum, both the heavy users and the light users are policed because of the insufficient filling rate. The light users are forced to reduce their throughput and the unfairness increases compared to the unfairness with the optimal rate. Policing the light users is counter-productive if the purpose is to reduce unfairness between light and heavy users; one has to attribute filling rates which will avoid the light users from being policed while keeping the heavy users from overloading the network during busy hours.

To evaluate the impact of the harshness of the policer, a soft, a mild and a strict policer are used, which drop packets with increasing aggressiveness. Figure 4 shows that the three policers present the same optimal filling rate but are different in decreasing the unfairness. The harsher is the policing, the lower is the unfairness, because the heavy users will need to further reduce their throughput due to the policer's higher dropping probability. The difference between the policers is substantial because when the policer drops packets, ConEx will react by sending more Re-Echo-Loss packets which will eventually lead to more policing. With a severe policer, the risk is to have a user continually decreasing his throughput because of the policer's actions while the network is uncongested. This fact should be taken into account in the design of the policer's algorithm.

C. ConEx with increasing complexity

The deployability of ConEx is a major concern for a network operator, and ConEx allows incremental deployement by requiring only a few modifications to be operational. It can afterwards be upgraded to provide a more accurate feedback of congestion information.

TABLE II. ConEx with increasing complexity

Case	queue	sender	receiver
DTConEx	DropTail	No ECN	No ECN
REDConEx	RED	No ECN	No ECN
FullConEx	RED	Accurate ECN	Accurate ECN

The minimum modifications needed for ConEx are the modifications of the sender which will react to a loss detection by sending a Re-Echo-Loss signal. In this case, ECN support is needed neither on the sender nor on the receiver and the RED queue can be replaced by a simple DropTail queue, which will drop packets when it overflows. In the next paragraphs, this case is referred as the $DTConEx$ case. The next step of modifications is when a RED queue is used on the router to improve reactivity to congestion appearance. ECN is not used and ConEx will react only to dropped packets by the RED queue. This is referred as the $REDConEx$ case. The ultimate step of modifications is when ECN is used by both the sender and the receiver along with modifications to the receiver to allow accurate ECN feedback. This is referred as the $FullConEx$ case. The three cases are summurised in Table II.

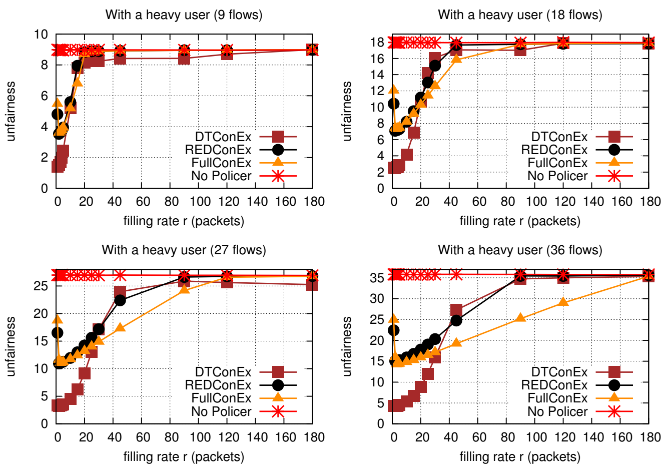

Figure 5. Unfairness with DTConEx, REDConEx and FullConEx

Figure 5 depicts the average unfairness versus the filling rate in four scenarios where we vary the number of flows per heavy user (9, 18, 27, 36), while the light user remains with a single flow. In each scenario, the red curve represents the unfairness without policing, while the three other curves represent the three cases explained above. As the number of flows of a heavy user increases, the number of Re-Echo packets sent increases, consuming more tokens, leading to more policing, so the range of filling rates allowing fairness improvement is widened.

In all scenarios, we see that $FullConEx$ decreases the unfairness more than $REDConEx$. The reason is that the former case provides both the information on ECN and on losses, which makes the policer more accurate in its actions.

The $DTConEx$ case provides even less congestion information than the two other cases but manages to decrease more the unfairness in all scenarios in a range of filling rates around the optimum. $DTConEx$ is effective because it does not make the light users reduce their throughput as early as the two other cases. Indeed, in both $REDConEx$ and $FullConEx$, the queue drops or marks packets when its mean length exceeds a minimum threshold forcing the users to reduce their throughput. The DropTail queue only drops packets when the entire queue is filled, which gives the opportunity for the light users to increase their throughput when heavy users are restrained by the policer.

Figure 6 represents the mean queueing delay and the loss rate that a light user encounters as a function of the filling rate (scenario with 36 flows per heavy user). A DropTail queue does not allow, unlike RED, to reduce the queueing delay observed by the users as we can see in the $DTConEx$ case. As the DropTail queue is entirely filled, the users experience the highest delay equal to 100ms. In $REDConEx$ and $FullConEx$, the queueing delay is reduced by the action of the RED queue. $REDConEx$ is reducing the queueing delay more than $FullConEx$ because the RED queue drops packets in the former case while it marks them in the latter. The congestion policer also contributes to reduce the queueing

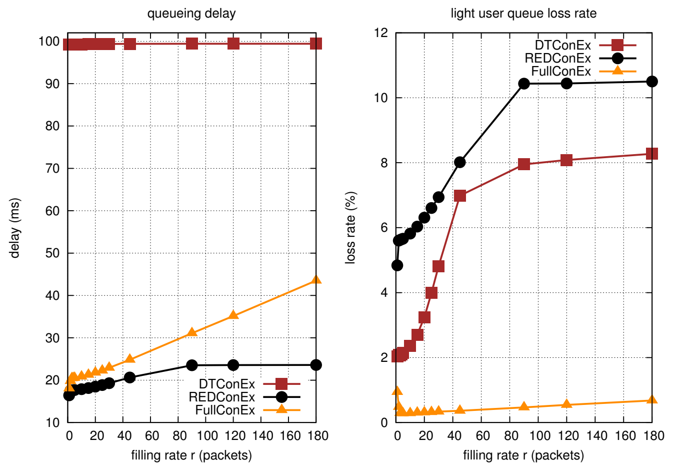

Figure 6. mean queueing delay and queue loss rate of a light user

delay which is further decreased as the filling rate decreases due to heavy users' policing.

By reducing traffic pressure on the bottleneck, the congestion policing also reduces the loss rate light users encounter, especially in $DTConEx$ and $REDConEx$, which are based only on losses in order to notify congestion. In both cases, the light users' loss rate decreases as the filling rate decreases. For all filling rates, $REDConEx$ shows a highest loss rate than $DTConEx$ because in the former case, the RED queue begins dropping packets earlier than the DropTail queue in the latter case. Finally, $FullConEx$, in which packets are ECN-marked rather dropped, shows a significantly lower loss rate for light users than the two other cases.

TABLE III. Performance summary

Case	Fairness	Loss rate	Delay	Deployability
DTConEx	***	**	*	***
REDConEx	*	*	***	**
FullConEx	**	***	**	*

Table III summarises the advantages and drawbacks of each case in terms of fairness improvement, loss rate, queueing delay and deployability.

V. Short-Lived flows

Short-lived flows represent a great number of flows that cross the Internet (Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP), Web objects, etc.). These flows are just a few packets long, they finish during the slow-start phase (in few RTTs) before reaching their fair-share rate [15]. This section aims to see how ConEx, which is a closed-loop mechanism requiring a number of RTTs to gather congestion information, behaves with short-lived flows and if it does bring an improvement to the completion time of these flows.

For performance evaluation, we use the same topology as in Section IV but modify the traffic sources from saturated long-lived flows to short-lived flows lasting only 10 packets. We use the aggregated traffic model described in [16], which uses a gamma distribution for the flow inter-arrival time, with a newly generated flow every 6ms on average. The 10 heavy users will generate 80% of the flows while the light users will generate the remaining 20%. In order to experience congestion in the network, a Not-ConEx cross traffic of 90Mbps over the 100Mbps bottleneck is generated. A strict policer is used as described in Section III-F. We monitor the flow completion time as a performance metric.

Each simulation lasts 600s, 30 simulations are performed to obtain a single point with a 95% confidence interval which is depicted on the graphs.

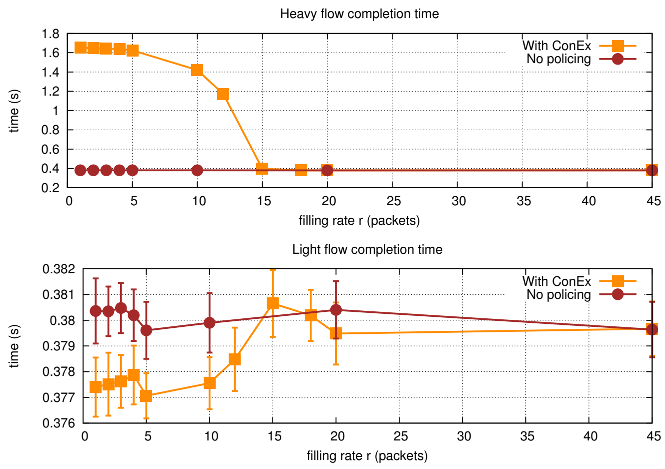

Figure 7. Completion time of a light and a heavy user's flow

Figure 7 represents the average completion time of a heavy user's flow and a light user's flow with and without the use of ConEx. The flows have an initial window of 2 segments and can be completed in 3 RTTs (300ms) which does not allow them to provide much congestion information for ConEx. Nevertheless, a heavy user can be policed when the filling rate is low enough ($r < 15packets$), increasing greatly the completion time of his flows. The completion time of a 10-segment flow ranged from 380ms without policing up to 1.64s when policed. This is supposed to free the bottleneck for the light users' flows. Indeed, we see that when the heavy user is delayed, the light users benefit from a reduced completion time, but the decrease is only a few milliseconds, which is hardly a significant improvement.

Neither ConEx benefits from the use of short flows nor short flows benefit from ConEx. Short flows are not suited to retrieve congestion information for ConEx as they finish in few RTTs. These flows finish before they can react to policing. When short flows lose packets, they can see their completion time increase dramatically from a few milliseconds to several seconds because they might need to wait for an RTO to perform retransmissions and complete. As expected, ConEx behaves poorly in presence of short flows, and it should be even less interesting if, as [15] suggests, the initial window is increased to 10 segments, which represents a less favorable scenario than the simulated one. However, the poor behaviour of ConEx observed with short flows does not lessen the interest of the mechanism considering that long flows are the main source of congestion. If a per user congestion policer is used, it should be more profitable to focus on long flows, which can retrieve congestion information and can efficiently react to policing.

VI. Video Streaming Traffic: YouTube use case

We have observed over the last years an impressive growth of the video streaming traffic in both Orange's fixed and mobile networks (36% for FTTH, 26% for Asymmetric Digital Subscriber Line (ADSL) and 39% for mobile downstream [1]). This led us to analyse how ConEx can alleviate the pressure caused by video streaming traffic and we chose as a use case the very popular YouTube plateform.

A. YouTube server model

Many papers analysed the YouTube traffic generation. Among them, [17] [18] propose an algorithm to reproduce the behaviour of a YouTube server, which we implemented in NS2.

A server sends a video in two phases: the first phase is called the Initial Burst where 40s of video data is sent at maximum rate to provide sufficient buffering to the player. The second phase is called the Throttling phase where the server sends the rest of the video data in chunks with a $sending\ rate = 1.25 \times encoding\ rate$ of the video. The chunk size is 64KB and the chunks are sent over a TCP socket with a 2MB sending buffer.

B. YouTube player model

We used the most precise monitoring approach proposed by [19] to implement a YouTube player in NS2. It is based on the status of the video buffer on the client player. The player starts playing the video when the buffered length exceeds a first threshold $\theta_0 = 2.2s$. If the buffer is depleted and the buffered length goes below a second threshold $\theta_1 = 0.4s$, the video stalls until the buffered length exceeds θ_0, then the video can start anew. We retrieve from the video player the number of stalling events N and their average length L to compute the QoE following a model suggested by [20] with the following equation:

$$QoE(L, N) = 3.50 \exp^{-(0.15L+0.19).N} + 1.50 \qquad (2)$$

C. YouTube results

The same topology as in Section IV is used to perform the simulations with 10 heavy users and 50 light users. The simulated scenario is the following: in the first 100s of the simulation, the heavy users have 20 FTP flows downloading at the maximum rate they can reach. No light user is present yet, the 10 heavy users can equaly share the bottleneck. In the next 100s, the light users begin requesting, randomly and uniformly over the 100s, a video from the servers. This video has a 300s length and a bitrate of 1128kbps, which corresponds to the recommended bitrate for uploading 360p videos to YouTube (1000kbps for the video bitrate and 128kbps for the stereo audio bitrate [21]). The heavy users, which are responsible for 80% of the traffic, now have to share the network with the newcomers. At $t = 500s$, all light users should have finished watching their 300s video if no stalling events hampered the viewing, and the heavy users should be able to continue using the bottleneck until the end of the simulation 100s later. The mean QoE of the light users is computed at the end of each simulation.

A simple DropTail queue is used at the bottleneck. The policer is a strict policer as described in Section III-F and all users use cubic as a congestion control algorithm.

Figure 8 shows the throughput of the heavy and the light users versus time. The three time periods of the simulated scenario are shown: before the arrival of the light users (0s-100s), during the light users' presence (100s-500s), and after

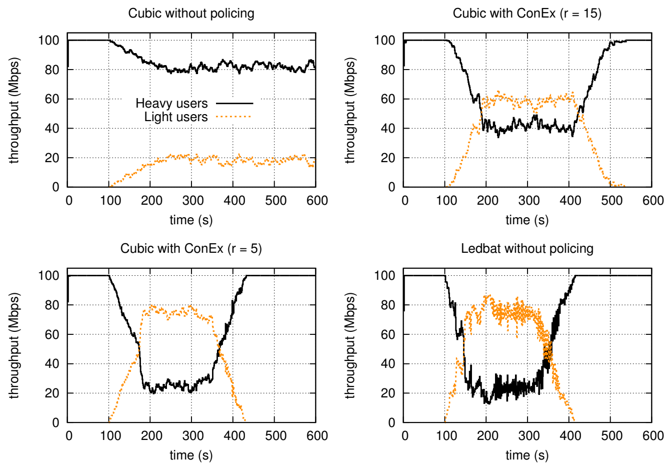

Figure 8. Throughput of heavy and light users versus time

the presumed departure of the light users if they watched the videos smoothly (500s-600s).

Figure 9 represents the computed QoE, the number of stalling events and the duration of a single stalling event for a light user in the following three cases: using cubic as a congestion control algorithm for heavy users without policing, using cubic for heavy users with ConEx policing and using LEDBAT as a congestion control algorithm for heavy users without policing.

a) Cubic without policing: When no policer is used, TCP with cubic will share the bottleneck equally between flows. The heavy users get 80% of the bottleneck and the light users will not be able to watch the video before the end of the second period. The light users will still be active during the third period, reducing the throughput of the heavy users when compared to the first period. The light users see their video stall many times and for a long duration as shown in Figure 9, resulting in a $QoE = 1.5$, which is the lowest obtainable value with equation (2). Users with this low QoE would have stopped watching the video when the first stalling events occured.

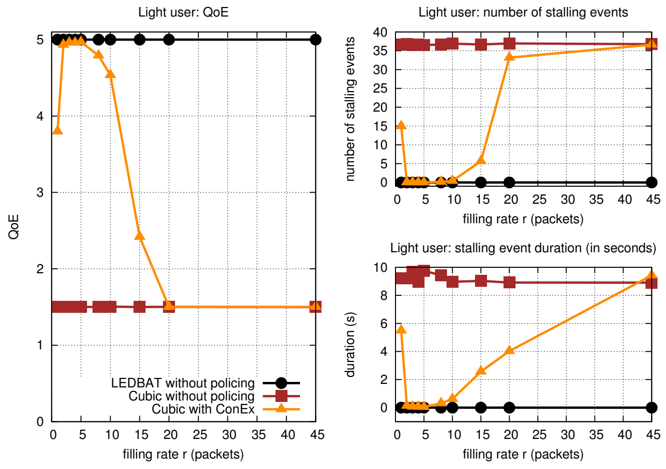

Figure 9. QoE of light users, the number of stalling events and the duration of a single stalling event

b) Cubic with ConEx: ConEx is activated in order to police the heavy users. Figure 9 shows that as the filling rate decreases, the light users' QoE increases to very good values

(QoE > 4) due to the reduced number of stalling events. The gain in QoE for the light users results from the heavy users decreasing their throughput, due to congestion policing, during the second period as represented in Figure 8. The light users are then able to finish viewing their video before the end of the second period. As the light users leave the bottleneck, the heavy users can increase their throughput during the third period.

c) LEDBAT without policing: The heavy users could avoid policing by being less aggressive towards video traffic. They could either postpone their activities until a less congested period, or they could use a less aggressive congestion control algorithm which yields the network ressources when encountering congestion. LEDBAT [22] is such a congestion control algorithm. It uses the available bandwidth in a bottleneck and yields in presence of standard TCP. When LEDBAT is used (implementated in NS2 by [23]) instead of cubic for the heavy users, results in Figure 8 show that, without requiring any policing, the heavy users decrease their throughput and the light users are able to watch their video with a very good QoE (Figure 9), similar to the results obtained by using cubic and ConEx policing ($r = 5$). When the light users' videos finish, LEDBAT is able to use the freed resources in the bottleneck.

As suggested in the ConEx charter [2], ConEx can be deployed in order to incentivize the heavy users to use a LEDBAT-like congestion control mechanism. The use of LEDBAT prevented the heavy users from consuming tokens for applications like file transfer, preserving their congestion allowance for more critical applications, while allowing the light users to have a good quality of experience. If the video delivery relies on HTTP-adaptive streaming, the light users would decrease the resolution of their video when encountering congestion, but after the heavy users have reduced their throughput using LEDBAT or in response to ConEx policing, the light users could increase the resolution of their video and benefit from a higher video quality.

VII. Summary and conclusion

ConEx is a new mechanism that allows a user to inform the network of the amount of congestion encountered. This allows the network operator to implement congestion-based policies proportionally to the amount of congestion a user has contributed to.

In Section IV, we have seen that ConEx allows us to differentiate between a light and a heavy user to improve the fairness between users. We have shown that ConEx can still improve fairness even with the minimum modifications (the ability to react to lost packets by sending a Re-Echo-Loss signal) and the use of simple DropTail queues. So, an efficient initial deployment is possible, as suggests [3], before considering the deployement of a more accurate ConEx relying on ECN, which requires modifications to both the senders and the receivers and the use of RED queues. The advantages and drawbacks of each step of modifications are summarized in Table III.

In Section V, we argued that neither ConEx benefits from the use of short flows nor short flows benefit from ConEx. Indeed, the short flows do not provide enough congestion information to ConEx, and policing them is not beneficial for their completion time. It is more profitable to focus on policing long and responsive flows.

In Section VI, we have seen how video streaming like YouTube can benefit from ConEx. ConEx can be used to restrain the heavy users who do not yield voluntarily under congestion, while leaving unpoliced those who do through a congestion control mechanism like LEDBAT. This should provide incentives for the heavy users to be more cooperative during congestion periods. The use of LEDBAT can protect the heavy users from being policed through ConEx while allowing the light users to have a great QoE.

VIII. Future Work

Implementing a per user congestion policer requires the determination of the policer's parameters, the filling rate (the allowed Congestion-Rate) and the bucket depth (the allowed Congestion-Burst). Different kinds of flows with different behaviours need to be policed with the same allowance rate which makes the determination of these parameters challenging. Further studies are required on this subject.

The congestion policing function is the key to improve fairness between users and to enforce some users to yield if they do not voluntarily. Designing a policer algorithm that achieves the goals we set is a crucial point in the deployement and is one of the main objectives of our future work.

Finally, the auditor can be necessary if there is a risk that the sources do not report the right Congestion-Volume they encounter. If auditing is relatively easy when ECN is used, ConEx on loss is more challenging as it requires detecting lost packets in the auditor. To address these issues, we can harness the substantial work concerning the auditor that has been done under the Trilogy project [5].

References

[1] M. Feknous, T. Houdoin, B. Le Guyader, J. De Biasio, A. Gravey, and J. Torrijos Gijon, "Internet traffic analysis: A case study from two major european operators," in Computers and Communication (ISCC), 2014 IEEE Symposium on, June 2014, pp. 1–7.

[2] ConEx Working Group Charter. [Online]. Available: http://datatracker.ietf.org/wg/conex/charter/ [retrieved: March, 2015]

[3] B. Briscoe, R. Woundy, and A. Cooper, "Congestion exposure (conex) concepts and use cases," December 2012. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6789.txt [retrieved: March, 2015]

[4] M. Kühlewind and R. Scheffenegger, "Tcp modifications for congestion exposure," November 2014. [Online]. Available: http://www.ietf.org/id/draft-ietf-conex-tcp-modifications-07.txt [retrieved: March, 2015]

[5] B. Briscoe et al., "Final report on resource control, including implementation report on prototype and evaluation of algorithms," December 2010. [Online]. Available: http://www.trilogy-project.org/fileadmin/publications/Deliverables/D13_-_Final_report_on_resource_control__including_implementation_report_on_prototype_and_evaluation_of_algorithms.pdf [retrieved: March, 2015]

[6] M. Kühlewind and M. Scharf, "Implementation and performance evaluation of the re-ecn protocol," in Incentives, Overlays, and Economic Traffic Control, ser. Lecture Notes in Computer Science, B. Stiller, T. Hoßfeld, and G. Stamoulis, Eds. Springer Berlin Heidelberg, 2010, vol. 6236, pp. 39–50. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-15485-0_5 [retrieved: March, 2015]

[7] F. Mir, D. Kutscher, and M. Brunner, "Congestion exposure in mobility scenarios," in Next Generation Internet (NGI), 2011 7th EURO-NGI Conference on, June 2011, pp. 1–8.

[8] Y. Zhang, I. Johansson, H. Green, and M. Tatipamula, "Metering re-ecn: Performance evaluation and its applicability in cellular networks," in Teletraffic Congress (ITC), 2011 23rd International, Sept 2011, pp. 246–253.

[9] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” IEEE/ACM Trans. Netw., vol. 1, no. 4, pp. 397–413, Aug. 1993. [Online]. Available: http://dx.doi.org/10.1109/90.251892 [retrieved: March, 2015]

[10] K. Ramakrishnan, S. Floyd, and D. Black, “The addition of explicit congestion notification (ecn) to ip,” September 2001. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3168.txt [retrieved: March, 2015]

[11] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, “Data center tcp (dctcp),” SIGCOMM Comput. Commun. Rev., vol. 41, no. 4, pp. –, Aug. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=2043164.1851192 [retrieved: March, 2015]

[12] M. Kühlewind and R. Scheffenegger, “Design and evaluation of schemes for more accurate ecn feedback,” in Communications (ICC), 2012 IEEE International Conference on, June 2012, pp. 6937–6941.

[13] The Network Simulator - ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/ [retrieved: March, 2015]

[14] A. Martin and M. Menth, “Conex-based congestion policing – first performance results,” March 2012. [Online]. Available: http://www.ietf.org/proceedings/83/slides/slides-83-conex-5.pdf [retrieved: March, 2015]

[15] N. Dukkipati et al., “An argument for increasing tcp’s initial congestion window,” SIGCOMM Comput. Commun. Rev., vol. 40, no. 3, June, 2010, pp. 26–33. [Online]. Available: http://doi.acm.org/10.1145/1823844.1823848 [retrieved: March, 2015]

[16] S. Gebert, R. Pries, D. Schlosser, and K. Heck, “Internet access traffic measurement and analysis,” in Proceedings of the 4th International Conference on Traffic Monitoring and Analysis, ser. TMA’12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 29–42. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-28534-9_3 [retrieved: Available: http://dx.doi.org/10.1007/978-3-642-28534-9_3 [retrieved: March, 2015]

[17] P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz, and J. Lopez-Soler, “Analysis and modelling of youtube traffic,” Transactions on Emerging Telecommunications Technologies, vol. 23, no. 4, June, 2012, pp. 360–377. [Online]. Available: http://dx.doi.org/10.1002/ett.2546 [retrieved: March, 2015]

[18] J. Ramos-munoz, J. Prados-Garzon, P. Ameigeiras, J. Navarro-Ortiz, and J. Lopez-soler, “Characteristics of mobile youtube traffic,” Wireless Communications, IEEE, vol. 21, no. 1, February, 2014, pp. 18–25.

[19] R. Schatz, T. Hossfeld, and P. Casas, “Passive youtube qoe monitoring for isps,” in Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on, July 2012, pp. 358–364.

[20] T. Hoßfeld, R. Schatz, E. Biersack, and L. Plissonneau, “Internet video delivery in youtube: From traffic measurements to quality of experience,” in Data Traffic Monitoring and Analysis, ser. Lecture Notes in Computer Science, E. Biersack, C. Callegari, and M. Matijasevic, Eds. Springer Berlin Heidelberg, 2013, vol. 7754, pp. 264–301. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36784-7_11 [retrieved: March, 2015]

[21] Google, “Advanced encoding settings.” [Online]. Available: https://support.google.com/youtube/answer/1722171 [retrieved: March, 2015]

[22] S. Shalunov, G. Hazel, J. Iyengar, and M. Kühlewind, “Low extra delay background transport (LEDBAT),” December 2012. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6817.txt [retrieved: March, 2015]

[23] D. Rossi, C. Testa, S. Valenti, and L. Muscariello, “Ledbat: The new bittorrent congestion control protocol,” in Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on, Aug 2010, pp. 1–6.