ConEx Performance Evaluation and Application to Video Streaming

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

André-Luc Beylot
INP-ENSEEIHT, IRIT Laboratory,
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, allowing the network operator to improve the fairness between users. ConEx is designed to limit the added complexity, leveraging already existing mechanisms such as Random Early Detection (RED) and Explicit Congestion Notification (ECN), plus a number of modifications to the senders and receivers to be fully operational. Nonetheless, ConEx can also be deployed in a simplified mode of operation, relying only on loss information in DropTail queues to estimate congestion. The objective of this paper is to provide an in depth evaluation of ConEx mechanism. Firstly, we investigate how the setting of ConEx parameters (e.g., congestion policer) and the network configuration (e.g., router queuing, network delay, etc.,) impact the behavior of ConEx and influence its ability to improve fairness between users. Secondly, we compare the level of performance, in terms of fairness improvement, provided by different variants of ConEx of increasing complexity, i.e., from a simple implementation with modifications limited to the sender to a "full" ConEx approach implementing all proposed features. We show that, despite a reduced accuracy in congestion estimation, a simple variant of ConEx is already able to provide a good fairness improvement between users. This is particularly interesting in the context of an initial deployment scenario, allowing an incremental deployment of ConEx. Thirdly, we investigate and discuss the limitations and weaknesses presented by ConEx with regard to short-lived flows. Finally, based on a YouTube traffic model, we illustrate how ConEx can help to enhance the Quality of Experience (QoE) of video streaming users during congestion periods, significantly reducing the number and duration of stalling events.**

Keywords-ConEx; Performance; ECN; Congestion; Policing; YouTube; LEDBAT.

I. INTRODUCTION

This paper complements the investigation of ConEx presented in [1], adding new simulation results and more detailed discussions.

During the network's busy hours, an amount of traffic greater 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, often referred as "heavy" users. For example, in Orange's Fiber To The Home (FTTH) access networks, 80% of downstream traffic is generated by 15% of the customers [2]. In order to improve the user's network experience, while restraining the network costs, 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 by network operators, like rate-limiting traffic 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, i.e., if it is congested or not. A heavy user can be rate-limited even when he does not hamper the experience of the others, or when there are plenty network resources available, which would allow his traffic to be far much faster. Similarly, a heavy user might consume his allowed Data-Volume even when the network is not in a congestion phase, which can be perceived as largely unjustified. 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 is generally available to the end-to-end flow control algorithms, for example, it can be exchanged between the users at the transport layer (e.g., through TCP acks), but it is transparent for the network layer. As the network elements operate at the network layer, they cannot have access to congestion information.

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 [3]. The amount of lost and congestion 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.

In order to minimize the implementation complexity, ConEx largely relies on existing mechanisms (e.g., RED, ECN capability on routers, TCP exchanges), and on new features added to both the sender and the receiver to be fully ConEx-capable. Considering the initial deployment of ConEx, we are interested in whether or not ConEx still presents good performance without the use of ECN in the network and relying only on minimal modifications to the user's end devices.

The additions to [1] are the following: firstly, the impact of the network configuration (e.g., router queueing, network delay) on ConEx mechanism is evaluated to determine how it may influence its ability to improve fairness between users. The sensitivity of ConEx to its environment is a key factor when considering its deployment in a real network. Secondly, the introduction of a new step of deployment in the analysis of the performance of ConEx variants with an increasing

implementation complexity, is also a valuable addition. It enhances the understanding of how all the ConEx components interact to achieve the goal of improving fairness between users.

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 main outcomes of the study, finally, Section VIII discusses the future work, still waiting to be covered.

II. Related Work

The IETF has set up since June 2010 a working group to develop experimental specifications of ConEx in IPv6 networks [3]. A Request For Comments (RFC) [4] discussing the concepts and use cases has been published, and other documents concerning the ConEx mechanism have also been produced and are waiting for final adoption: the use of a destination option in the IPv6 Header to carry the ConEx markings [5], a mobile communications use case for congestion exposure [6] and the necessary modifications to TCP [7].

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 [8]. 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. [9] 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. [10] evaluates mobility issues with congestion exposure and shows that mobility is not a major concern for Re-ECN. [11] evaluates Re-ECN applicability in LTE networks and found that it can bring a significant improvement for these networks unless they experience a 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 user's traffic.

A. ConEx mechanism

Figure 1 shows the whole ConEx process and all the elements involved with it, in case of TCP traffic, which is the primarily target for ConEx. 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 [12]) by the congested routers. This information about loss or marking will be detected by 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.

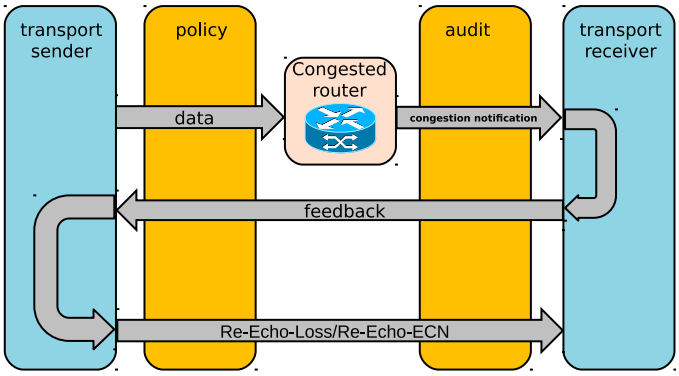

Figure 1. ConEx mechanism

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 might be used to ensure that the senders are exposing the right amount of congestion to the network. It helps as prevention from users understating the congestion their flows encounter, to preserve their congestion allowance and avoid policing. If the sources are trusted ones, for example, if the sources are controlled by the network operator or if there is an agreement between the sender and the network, the auditor is unnecessary and can be omitted. As reliable auditing is a complex task this greatly simplifies the deployment of ConEx.

B. Random Early Detection

Random Early Detection is an Active Queue Management technique, implemented on many routers, which was first introduced in [13]. It allows to randomly drop or ECN mark packets according to a probability that increases from 0 to the maximum probability p_{max} when the mean queue length increases from a minimum threshold to a maximum threshold (see Figure 2). Above the maximum threshold, all packets are either dropped or marked if ECN is used (the "gentle mode" was introduced later on to fix the problem caused by the discontinuity of the marking or drop probability when the queue length exceeds the max threshold).

C. Explicit Congestion Notification

Explicit Congestion Notification [12] is a way to indicate the occurrence of congestion in the network without having to drop packets. It uses two ECN bits [ECT,CE] of the IP header to signal congestion to the receiver.

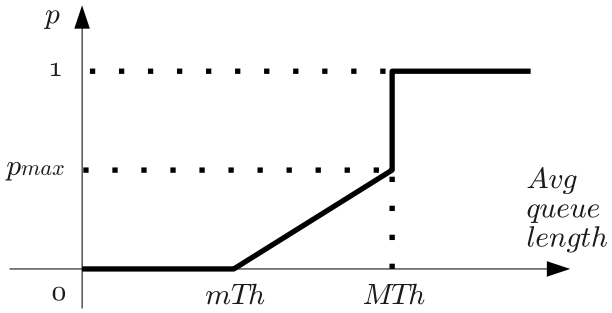

Figure 2. Random Early Detection dropping/ marking function

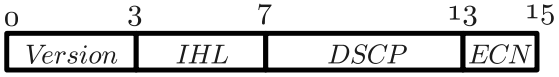

Figure 3. Bytes 1 and 2 of IPv4 header

D. Re-ECN

Re-ECN is a candidate implementation of ConEx for IPv4 [8]. 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

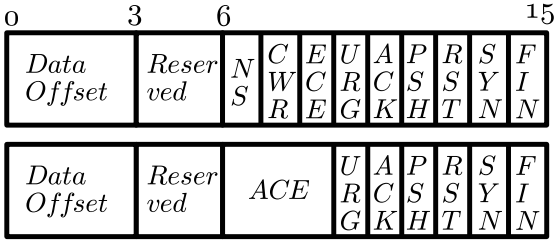

Figure 4. Bytes 13 and 14 of TCP header

The classic ECN mechanism as described in [12] 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 [14], modifications to TCP are needed to provide more than one feedback per RTT. [15] and [16] propose a solution to achieve such a goal. They suggest overloading the three TCP flags ECE, Congestion Window Reduced (CWR) and Nonce Sum (NS) to form a 3-bit field, the ACE field as shown in Figure 4. This field would act as a counter for the number of CE marks seen by the receiver, which can feedback it to the sender. The sender is then able to accurately follow the evolution of ECN markings and report the right amount of Re-Echo-ECN signals. The use of accurate ECN feedback is negotiated during the TCP three-way handshake.

F. Congestion policer

The great advantage brought by ConEx is the possibility for the network operator to police the users proportionally to their contribution to congestion, thus to the impact they have on other users. Based on the ConEx signal the policing can be applied at the ingress of the network, which is far more efficient than a policing at the egress by the auditor as it prevents the heavy users from overloading the network.

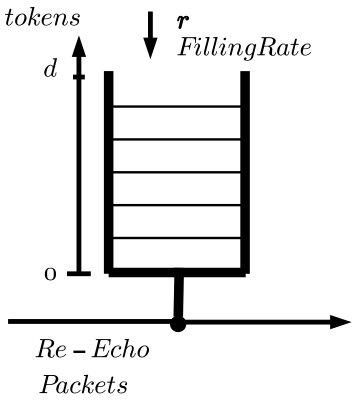

Figure 5. The congestion policer as a token bucket

The congestion policer can be implemented as a token bucket, as in Figure 5, with a filling rate r (the allowed Congestion-Rate) and a depth d (the allowed Congestion-Burst). In a byte-based mode of operation, 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. A packet-based policer, which does not consider the size of packets, can also be used, resulting in a simpler but potentially less accurate traffic control in case of heterogeneous packet sizes.

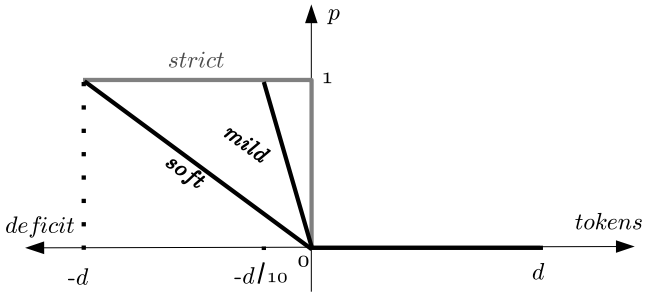

Figure 6. Drop function of the congestion policer

As shown in Figure 6, three levels of policing are used for performance evaluation characterized by their drop function: the strict policer discards all packets when the bucket is empty, the mild policer discards packets with a probability linearly

increasing from 0 to 1 when the bucket depth decreases from 0 to $-d/10$ and the soft policer discards packets with a probability linearly 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) [17] 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 7. 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 an 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.

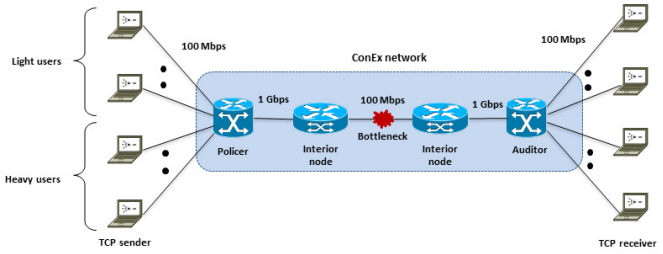

Figure 7. Simulated network topology

Unless specifically mentioned, the following configuration is used. All users have a minimum Round Trip Time of 100ms (due to the propagation time on the links) 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 (or 45 packets), 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 traffic. 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 that is deactivated because we use trusted sources, i.e., sources that are fully compliant to the behavior specified for ConEx.

Each simulation has a duration of 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 (i.e., sending at their maximum possible rate) and each flow starts randomly and uniformly 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, which is useful for a network operator, as providing fairness between its customers in their use of the network is necessary. So, monitoring the impact of the mechanism on the fairness between the users is valuable. Therefore, we will be monitoring a metric defined in [18]:

$$unfairness = \frac{throughput\ of\ a\ heavy\ user}{throughput\ of\ a\ light\ user} \quad (1)$$

In the following sections, we will firstly evaluate the impact of the internal and external parameters of ConEx on its performance. The internal parameters are the ones that come with the implementation of ConEx i.e., the congestion policer configuration (filling rate, harshness of the policer and bucket depth). The external parameters are the ones that come from the environment in which ConEx operates: the main network parameters (i.e., queuing strategy and delay) and the TCP congestion control algorithm (i.e., cubic and compound). Afterwards, we will compare the performance of ConEx for a set of implementation variants of increasing complexity, corresponding to increasing steps of deployment, a very essential consideration for a network operator.

TABLE II. Parameter values summary

Parameter	Values
Policer Filling Rate r (packets/s)	$1-2-3-4-5$ $10-15-20-25-30$ $45-90-120-180$
Policer Depth d (packets)	$5-12-23-$ **45** -90
Policer Harshness	Soft−**Mild**−Strict
Fixed RTT (ms)	$20-50-$ **100** $-150-200$
TCP Congestion Control	**Cubic**−Compound
Queue Size q (ms)	$10-20-50-$ **100** -200
Queue MinThresh mTh (%q)	$2.5-5-$ **10** $-20-50$
Queue MaxThresh MTh (%q)	$20-40-60-80-$ **100**
Queue Maximum Mark/Drop Probability	$0.1-0.25-0.5-0.75-$ **1**
ConEx Complexity	DTConEx−REDConEx ECNConEx−**FullConEx**

Table II summarizes the evaluated parameters and their values in the simulations. The value in bold represents the default value of the parameter, used when no value is specifically mentioned in the text.

B. Policer harshness

Figure 8 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.

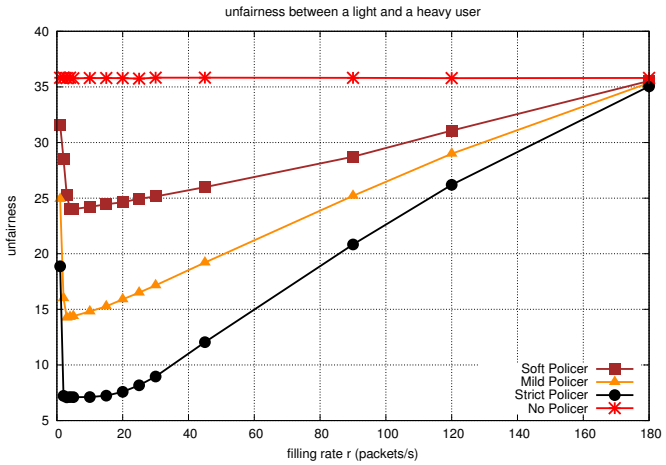

Figure 8. Unfairness between a heavy and a light user

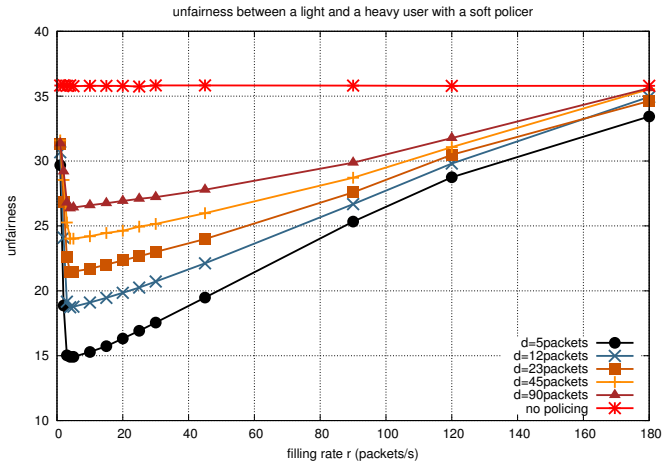

Figure 9. Unfairness with different depth values (in packets) with a soft policer

As the heavy users are forced to reduce their throughput, the light users occupy the freed bandwidth and the unfairness is reduced.

In Figure 8, 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 obtained with the optimal filling 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 8 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, the ConEx-enabled source 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, even when the network becomes uncongested. This potential artefact should be taken into account in the design of the policer's algorithm.

C. Token Bucket Depth

The depth of the token bucket corresponds to the burst of ConEx signals the network operator allows a user to send, it conditions the quickness of the policer to take action against a user inducing congestion. The token bucket depth is involved in the dropping function of the soft and mild policer (the smaller the bucket, the steeper the dropping function) as shown in Section III-F, thus the depth also represents the progressivity of the policing in relation to the congestion induced. Figure 9 and Figure 10 represent the unfairness between users versus the filling rate for different depth values in the case of a soft and a mild policer respectively. In both cases, as the token bucket depth decreases, policing becomes more reactive to congestion, and less permissive towards heavy users, so the unfairness decreases as the heavy users react to dropped packets by reducing their throughput.

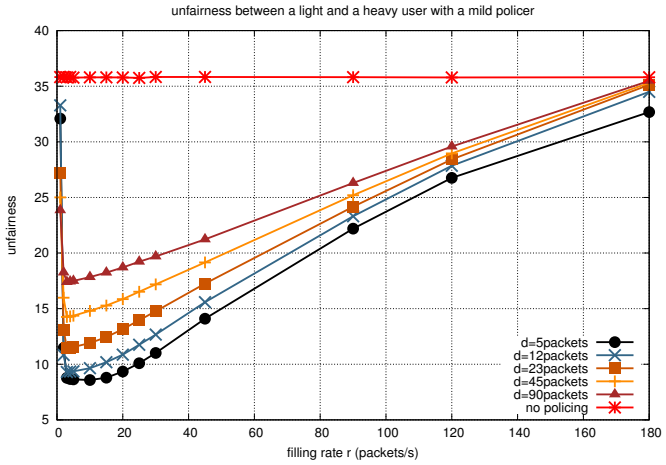

Figure 10. Unfairness with different depth values (in packets) with a mild policer

When the drop function is independent of the bucket length, like in the case of a strict policer (cf. Figure 11), only the reactivity of the policing is affected by the depth of the bucket, the harshness is not. As noticed in Figure 11, there is almost no difference in the decrease of the unfairness between the different depth values: long-lived flows can react to congestion by adapting their throughput, thus they induce a steady rate of congestion signals. If this congestion-rate is greater than the filling rate, it is only a matter of time for the bucket to be emptied completely, and for the policer to start taking action against the congestion inducing user, with the same harshness whatever the bucket depth. In the long run, the unfairness decrease would not be affected by the bucket depth

as much as by the filling rate and the dropping probability. So the reactivity of the policing, conditioned by the depth of the token bucket, is less significant than the harshness, conditioned by the filling rate and the dropping function steepness, in reducing the unfairness.

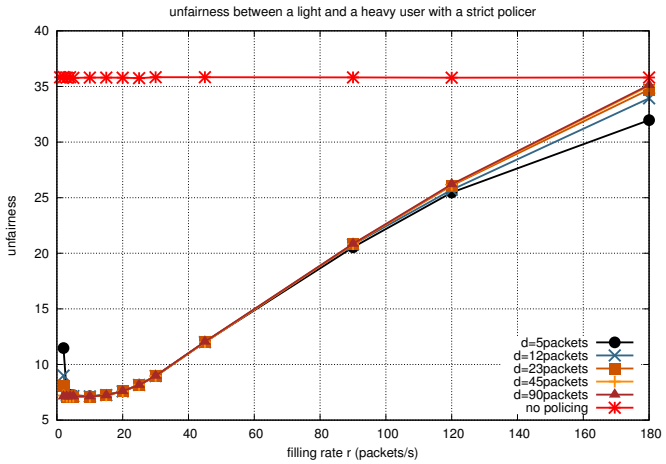

Figure 11. Unfairness with different depth values (in packets) with a strict policer

To proceed in the performance evaluation of ConEx, we use a mild policer and vary the values of the other parameters involved in the control of the mechanism or likely to influence its behavior. We will evaluate next the effect of the Round Trip Time.

D. Round Trip Time

We vary the minimum RTT (due to the propagation time in the links) of the users from 20ms to 200ms in the simulations. Having a short RTT allows a flow to quickly increase its congestion window, reaching a high throughput over a short time period. It also increases the probability for its packets to get marked or dropped: in a single second a flow have many "round trips" of traffic in the network, potentially resulting in many packets queued at the bottleneck and marked or dropped in case of congestion. This could drastically increase the number of Re-Echo-ECN and Re-Echo-Loss packets sent by a flow over a given time period. The user could then experience a high congestion-rate, rapidly consume the tokens in the token bucket when compared to the allocated filling rate, leading to a more severe policing. As shown in Figure 12, the unfairness significantly decreases with the RTT, and the difference observed between the curves is important, particularly for a RTT below 100ms. For the shortest round trip times (RTT below 100ms), even the highest filling rate ($r = 180 packets/s$) is not sufficient to allow the heavy users to deal with the congestion-rate they induce in the network, so the unfairness is still low even with a high filling rate. The results presented here clearly show that the round trip time is a very influential parameter for ConEx mechanism, especially in the design of the congestion policer algorithm.

E. TCP congestion control algorithm

In Figure 13, two popular TCP congestion control algorithms are compared, cubic and compound. Cubic is a more aggressive algorithm than compound that occupies more bandwidth, and can lead to more congestion on the bottleneck.

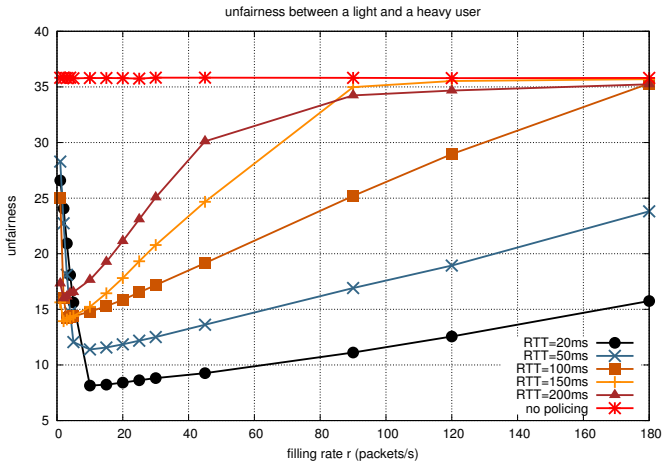

Figure 12. Unfairness with different RTTs

As a consequence, similarly to what was observed for a short RTT, it can be expected that cubic consumes more tokens, leads to more sanctions towards the heavy users hence results in a lower unfairness than compound TCP. On the contrary, the results show that compound TCP manages is more effective in reducing unfairness between users. In addition, the difference between cubic and compound remains almost constant when the filling rate varies, meaning that they do not have a significant impact on the behaviour of ConEx. The results show that the aggressiveness of cubic algorithm leads to more losses and markings during congestion, even for light users, forcing them to reduce their throughput more often and leading to a less effective unfairness reduction than with compound.

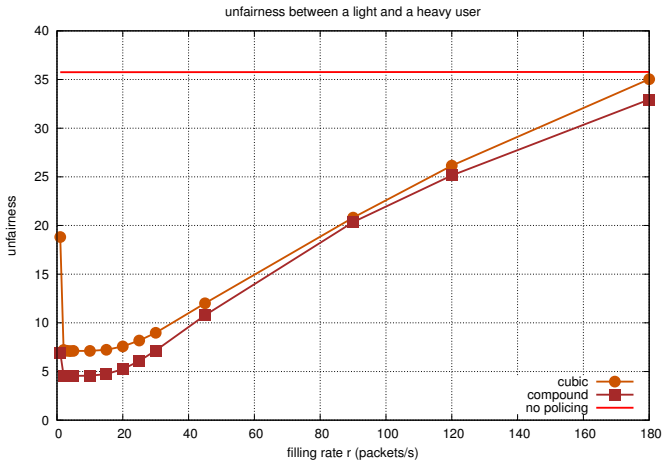

Figure 13. Unfairness with cubic and compound

Regarding more specifically ConEx, the obtained results show that both congestion control algorithms have a very similar behaviour in the presence of ConEx, resulting in roughly the same level of performance. This means that in the presence of ConEx, the users should be treated in the same way by the network, whatever their congestion control algorithm is, resulting in a relative independence regarding the type of device and/or the type of operating system.

F. Queue parameters

Many Active Queueing Management techniques and queueing algorithms can be used in a network depending on the objective the network operator aims at (reducing the queueing delay, the jitter, etc.), and these algorithms might have many parameters of their own, which will affect the behaviour of ConEx. In the case of the Random Early Detection algorithm, four parameters are involved: the queue size, the minimum threshold, the maximum threshold and the maximum marking probability. In this section, we will investigate the impact of each parameter on the behaviour of ConEx and try to quantify how it affects the effectiveness in reducing the unfairness.

In our evaluation the RED queue is used to mark packets, it only drops packets when it overflows. An ECN marked packets has two effects and provides two way to reduce the traffic load in the network. Firstly, it forces the traffic source to reduce its window size to lower its sending rate at a maximum rate of one time per RTT, similarly to what is done when detecting a packet loss. Secondly, it forces the traffic source to send a Re-Echo-ECN packet, consuming tokens at the policer and increasing the probability for the source to be policed.

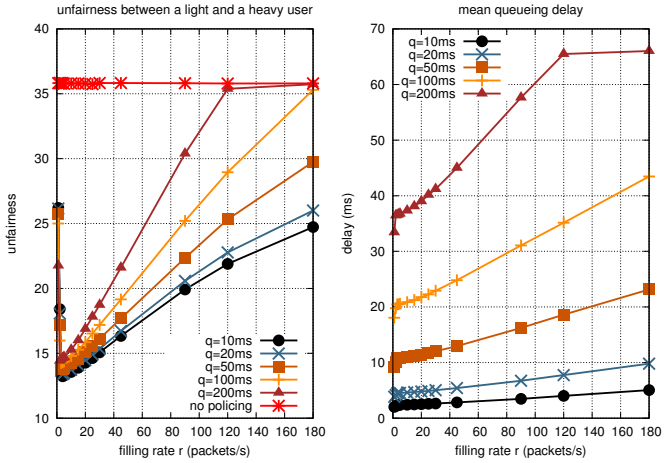

Figure 14. Unfairness with different queue sizes

1) Queue size: We vary the queue size to hold from 10ms to 200ms of the bottleneck traffic. A small queue quickly overloads, leading rapidly to an important number of packet drops. In the case of RED, as the marking probability increases with the mean queue length (cf. Section IV-A), the fraction of marked packets also increases rapidly for a small queue. Thus, the congestion induced by a user increases as the queue size decreases, with more and more Re-Echo-Loss and Re-Echo-ECN packets sent by the heavy users. These heavy users are policed and forced to reduce their throughput, and, as shown in Figure 14, the unfairness decreases significantly with the decrease of the queue length. It should be noticed that the decrease of the unfairness is also affected by the shorter RTT due to the shorter queue, as it is explained in Section IV-D, a short RTT leads to a reduced unfairness.

Policing the heavy users at the ingress of the network reduces the congestion in the network and allows the light users to have a greater share of the available bandwidth, but it also reduces the delay in the bottleneck queue. As the filling rate decreases, making the policing harsher, the mean queueing delay decreases, as shown in Figure 14. The value of the queueing delay without congestion policing is close to the value of the queueing delay when $r = 180 packets/s$, as with this filling rate, the heavy users are almost not policed.

For a given filling rate, the queueing delay exhibits important variations, which is natural as the queues varies in size, but compared to the impact on the unfairness, the impact of the queue size on the delay is much more significant. Thus, it can be considered that the impact of the queue size on the delay is much more a concern than its impact on ConEx.

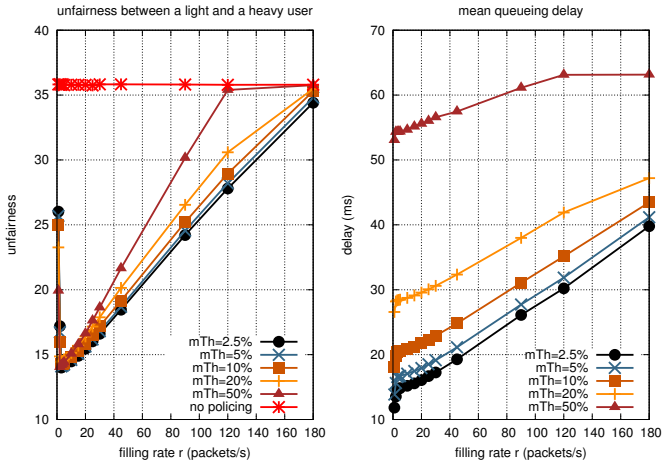

Figure 15. Unfairness with different minimum thresholds

2) Minimum Threshold: The minimum threshold determines when the queue will start marking packets. It can be expected that the sooner the marking will begin, the more Re-Echo-ECN packets will be sent by the users inducing congestion, consuming more tokens and being more severely policed. In Figure 15, the unfairness effectively decreases with the decrease of the minimum threshold, however, the unfairness is only slightly affected by the variation of the minimum threshold, particularly for the values of filling rate leading to the lower unfairness, making ConEx relatively insensitive to the minimum threshold setting. This can be explained by the fact that as the maximum threshold and the marking probability remained unchanged, a low minimum threshold results in a more reactive but less aggressive marking process.

On contrast the queueing delay is largely influenced by the variation of the minimum threshold, as already observed previously for the queue size variation case. This is quite natural as one of the main motivations for introducing RED was to control the queuing delay. Here again it can be concluded that setting the minimum threshold is primarily a question regarding the control of the queuing delay, with limited influence on the behavior of ConEx.

3) Maximum Threshold: When the mean queue length exceeds the maximum threshold, all packets are marked, so the lower the threshold the greater fraction of Re-Echo-ECN packets that will be sent by heavy users inducing congestion. Thus, the policer is more severe towards these heavy users and forces them to reduce their throughput. In Figure 16, the unfairness decreases with the decrease of the maximum threshold, because of the higher fraction of ECN marked packets in the queue, but as already observed for the minimum threshold, the unfairness is only slightly affected by the variation of the maximum threshold. The queueing delay also decreases with the maximum threshold, and here again, the queueing delay is

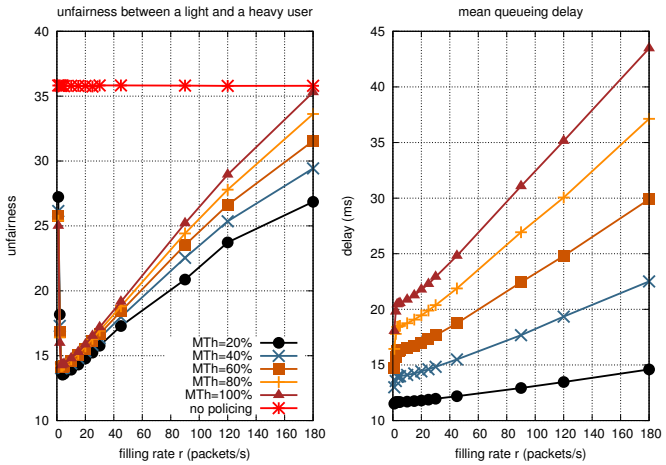

Figure 16. Unfairness with different maximum thresholds

widely impacted by this variation. From these results ConEx appears as also relatively insensitive to the maximum threshold setting. The drawback with a low maximum threshold is that the light users also experiment a high ratio of ECN marked packets in the queue. As they have to react to these marked packets, they are unable to increase their throughput when the heavy users are policed and the bottleneck freed.

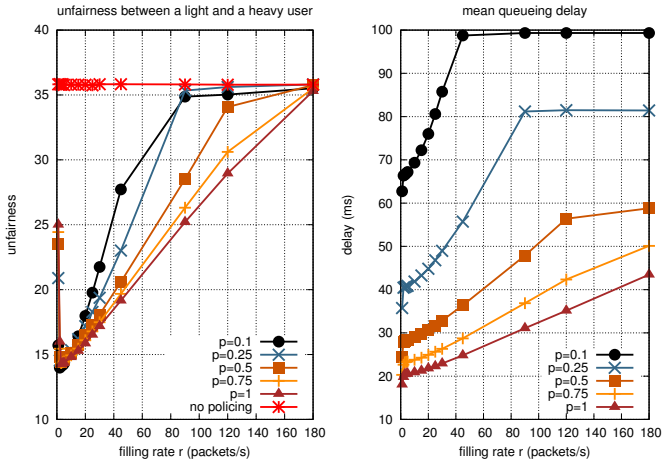

Figure 17. Unfairness with different maximum marking probabilities

4) Maximum marking probability: The maximum marking probability is reached when the mean queue length is equal to the maximum threshold, the higher the probability the more packets are marked so the more Re-Echo-ECN signals are sent by heavy users, making the policer harsher towards their traffic. As presented in Figure 17, a higher maximum marking probability decreases the unfairness, and it also decreases the queueing delay because of both the marking in the queue and the heavy users' policing at the ingress of the network.

The queueing delay is greatly affected by the variation of the marking probability. For a low marking probability the users experiment only small limitations and can largely (e.g., for a marking probability equal to 0.25), or completely (e.g., for a marking probability equal to 0.1), fill the queue, resulting in a high queuing delay. As the marking probability increases the user's limitation is more severe, resulting in a less loaded queue and a reduced delay. For low filling rate congestion policing is more severe and the queuing delay reduction is emphasized.

On contrast, unfairness exhibits a smaller sensitivity to the maximum marking probability. For a low filling rate (i.e., below 20 packets/s), the allowance of Re-Echo-ECN packets is so small that the marking probability has a limited impact on the unfairness reduction. For a high filling rate, on the opposite, the allowance is so important that here again the marking probability has a limited impact. It can even be observed that for a filling above 90 packets per second and a low maximum marking probability (i.e., below 0.25), the maximum rate of marked packets becomes close to the policer filling rate, resulting in heavy users being almost not policed.

From the investigation of the RED queue presented here, a trend can be observed: there is certainly an impact of the queue's parameters on the fairness improvement provided by ConEx, but it always goes with a significantly more important impact on the queuing delay. Thus, tuning of the RED parameters should focus much more on delay control, than on the influence of these parameters on ConEx. Additionally, whatever is the chosen tuning of the RED parameters, congestion policing can always provide a further reduction in the queueing delay, because of the traffic load reduction resulting from its action at the ingress of the network.

G. ConEx with increasing complexity

The deployability of ConEx is a major concern for both content providers and network operators. If a content provider can easily upgrade its servers, it does not control the traffic queuing mechanisms implemented on routers, nor the IP stack of receiving devices. On its part the network operator can modify the queuing strategy in its network but it does not control the IP stack on the senders and receivers. In that context the possibility of minimal modifications is a key factor for an introduction phase. ConEx allows incremental deployment by requiring only a few modifications to be operational. It can afterwards be upgraded, step by step, increasing the implementation complexity to provide a more and more accurate feedback of congestion information.

TABLE III. ConEx with increasing complexity

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

The minimum modifications needed for ConEx are the modifications to 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 to it 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 to it as the $REDConEx$ case. Another step of modifications is when ECN is activated on both the sender and the receiver, but the receiver does not provide an accurate account of the congestion signals it receives from the

network (cf. Section III-E), so only one congestion notification can be sent to the sender per RTT. This case is referred to it as the $ECNConEx$ case. The ultimate step of modifications is when ECN is used by both the sender and the receiver along with the modifications to the receiver to allow accurate ECN feedback (cf. Section III-E). This is referred to it as the $FullConEx$ case. The four cases are summurised in Table III.

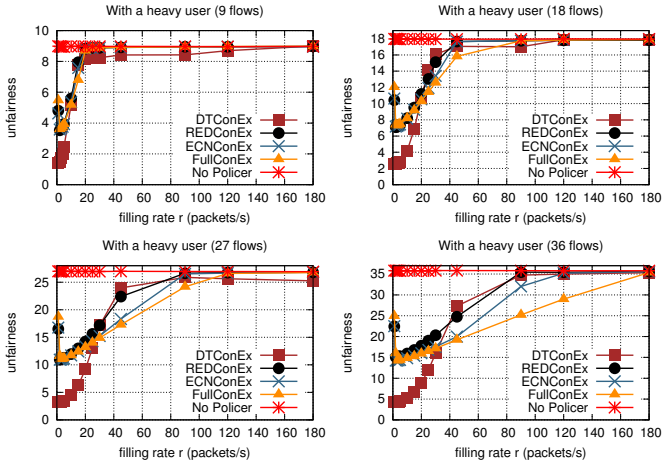

Figure 18. Unfairness with DTConEx, REDConEx, ECNConEx and FullConEx

Figure 18 depicts the average unfairness versus the filling rate in four scenarios where we vary the number of flows per heavy user (9, 18, 27, and 36), while the light user remains with a single flow. In each scenario, the red curve represents the unfairness without policing, while the four other curves represent the four cases explained above. As the number of flows of a heavy user increases, its contribution to congestion also increases. The user has to send more Re-Echo packets, he consumes more tokens and is more severely policed. As a consequence the range of filling rates allowing fairness improvement is widened.

In all scenarios, we see that $FullConEx$ and $ECNConEx$ have a similar behavior and decrease the unfairness more than $REDConEx$. The reason is that the two former cases provide the congestion information via both ECN and losses, which makes the policer more accurate than with $REDConEx$, which only provides the information on lost packets. In the same way, $FullConEx$ is slightly more effective than $ECNConEx$ in decreasing the unfairness, because it provides a more accurate congestion signal, particularly when the level of congestion increase (27 and 36 flows per user), allowing the congestion policer to more accurately restrain the heavy users.

The $DTConEx$ case provides even less congestion information than the other cases (i.e., only when the queue overflows), 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 force the light users to reduce their throughput as early as for the other cases. Indeed, for $REDConEx$, $ECNConEx$ and $FullConEx$, the queue starts dropping or marking packets when its mean length exceeds a minimum threshold, forcing the heavy users, and in a smaller proportion the light user, to reduce their throughput. On opposite 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.

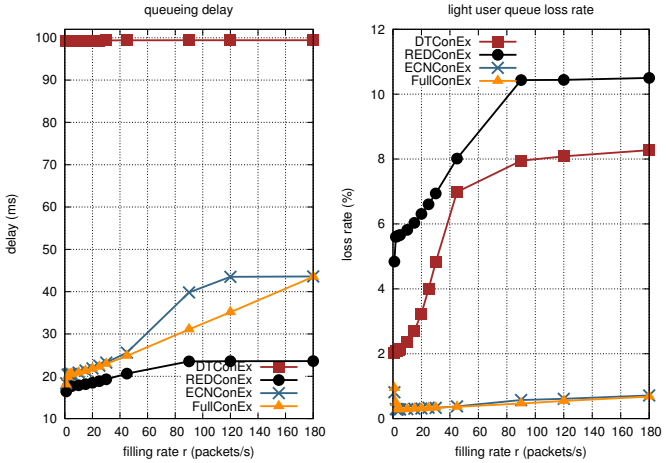

Figure 19. mean queueing delay and queue loss rate of a light user

Figure 19 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). Unlike RED, a DropTail queue does not allow reducing the queueing delay observed by the users, as we can see for the $DTConEx$ case. The DropTail queue is entirely filled when it starts dropping packets, so the users experience the highest delay equal to 100ms. For $REDConEx$, $ECNConEx$ and $FullConEx$, the queueing delay is reduced by the action of the RED queue. $REDConEx$ reduces the queueing delay more than $ECNConEx$ and $FullConEx$, because the RED queue drops packets while the two others only mark packets, leaving them in the queue. The congestion policer also contributes to the reduction of the queueing delay by limiting the amount of traffic entering the network. This effect is more visible as the filling rate decreases.

By reducing the traffic pressure on the bottleneck, the congestion policer also reduces the loss rate encountered by light users, especially in $DTConEx$ and $REDConEx$, which are based only on losses in order to notify congestion. In both cases, the light user's loss rate drastically decreases as the filling rate decreases. For all filling rates, $REDConEx$ results in a higher loss rate than $DTConEx$ because the RED queue begins dropping packets earlier than the DropTail queue. Finally, $ECNConEx$ and $FullConEx$, in which packets are ECN-marked rather than dropped, results in a similar and significantly lower loss rate for light users than the two other cases.

TABLE IV. Performance summary

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

Table IV summarises the advantages and drawbacks of each implementation variant 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 (e.g., Domain Name System (DNS), Web objects). These flows are just a few packets long, they finish during the slow-start phase (in few RTTs) before reaching their fair-share rate [19]. 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 [20], 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 that is depicted on the graphs.

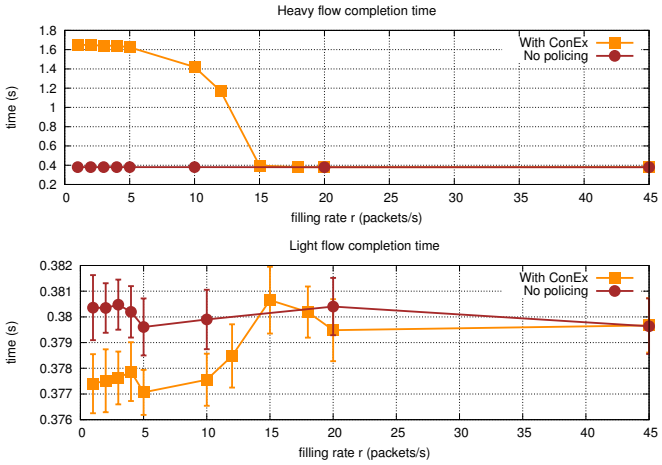

Figure 20. Completion time of a light and a heavy user's flow

Figure 20 represents the average completion time of a heavy user's flow and a light user's flow with and without the use of ConEx (FullConEx implementation in this simulation). 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 ($r < 15 packets$), 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. In addition, these flows also finish before they can react to policing. When short flows lose packets, they can see their completion time increases 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 [19] 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 [2]). 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, [21] [22] 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 [23] 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 [24] 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 equally share the bottleneck. During

the next 100s, the light users begin requesting, randomly and uniformly over the 100s, a video from the servers. This video has a 300s duration 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 [25]). 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.

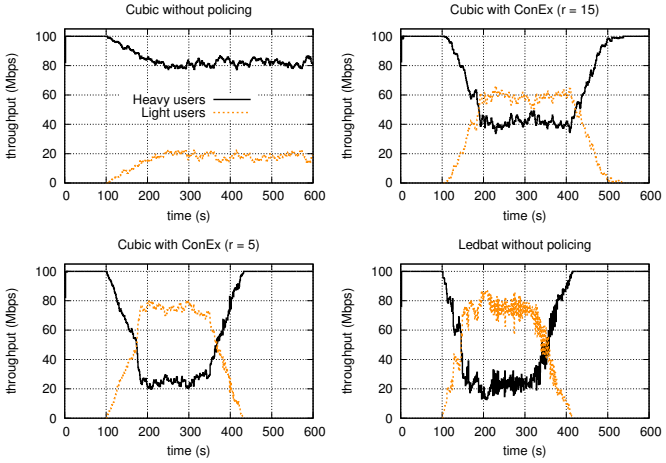

Figure 21. Throughput of heavy and light users versus time

Figure 21 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 the presumed departure of the light users if they watched the videos smoothly (500s-600s).

Figure 22 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.

1) 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, close to 10s, as shown in Figure 22, resulting in a $QoE = 1.5$, which is the lowest obtainable value with equation (2). It can be anticipated that in real life the users with such a low QoE would have stopped watching the video when the first stalling events occurred.

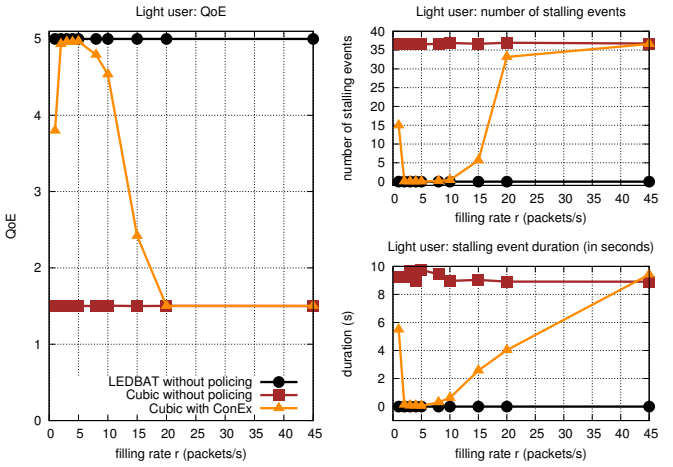

Figure 22. QoE of light users, the number of stalling events and the duration of a single stalling event

2) Cubic with ConEx: ConEx is activated in order to restrain the heavy users and improve the QoE of video users. Figure 22 shows that ConEx perfectly achieves this objective: as the filling rate decreases, the light users' QoE significantly increases due to a drastic reduction of the number of stalling events. For a filling rate below 15 packets/s video users benefit from a very good values ($QoE > 4$). The gain in QoE for the light users results from the heavy users reducing their throughput, in response to congestion policing, during the second period as represented in Figure 21. As represented for a filling rate of 15 and 5 packets/s, tuning the filling rate allows to finely control the bandwidth reparation between the heavy and light users. 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.

3) 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 resources when encountering congestion. LEDBAT [26] is such a congestion control algorithm. It is designed to use the available bandwidth in a bottleneck and yields rapidly in presence of standard TCP. When LEDBAT is used (implementated in NS2 by [27]) instead of cubic for the heavy users, results in Figure 21 show that, without requiring any policing, the heavy users rapidly decrease their throughput when the video users become active. The light users are then able to watch their video with a very good QoE (Figure 22), similar to the results obtained by using cubic and ConEx policing ($r = 5$). In the same way when the light users' videos finish, LEDBAT is able to use the freed resources in the bottleneck.

The behavior observed with LEDBAT may raise some questions: what is the usefulness of ConEx? Why not jumping directly to LEDBAT? In fact, the use of a congestion control mechanism like LEDBAT for bulk data transfers could be seen as a target, but to favor its adoption it is necessary for the network to find a way to acknowledge the users who adopt a TCP-friendly behavior. As suggested in the ConEx charter [3], ConEx can be deployed in order to incentivize the heavy users to migrate a LEDBAT-like congestion control mecha-

nism. 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.

In this study, we have decided to use a progressive download mode of video delivery as it allows accurately quantifying the benefits of ConEx and LEDBAT. If the video delivery relies on HTTP-adaptive streaming, similar behavior can be expected: 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. Many parameters (congestion policer parameters, RTT, TCP congestion control algorithm, RED queue) are involved in ConEx mechanism, and can influence its ability to improve fairness between users.

ConEx might be very sensitive with the harshness of the policing because of its interaction with the sent Re-Echo-Loss packets, as explained in Section IV-B. Tuning the policer harshness allows to precisely control the unfairness reduction for all level of congestion allowance (i.e., for all level of filling rate). Setting the token bucket depth allows to control the policer aggressivity without significantly impacting the level of performance. As for all mechanism operating in a close-loop mode between the sender and the receiver, the quickness of the congestion information retrieval, through a short RTT, also have a great impact on the behavior of ConEx and on its ability to improve the fairness between users. The results obtained with cubic and compound shows that ConEx is relatively insensitive to the TCP congestion control algorithms implemented by the end devices, meaning that users should be almost equally treated, whatever their congestion control is, resulting in a relative independence of ConEx regarding the type of end devices.

Finally, the investigation of the queuing strategy implemented by routers (i.e., RED queue) shows that the queue parameters have a limited impact on ConEx behavior, particularly on the fairness improvement through congestion policing, compared to their greater impact on other characteristics of the traffic like delay. The RED queue setting can then be optimized to control the queuing delay, with a limited impact on ConEx performance. We have also shown that ConEx can still improve fairness even with minimal 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 [4], before considering the deployment 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 IV.

In Section V, we illustrate the poor behavior of ConEx in presence of short-lived flows. 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 then 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. The results show the improvement that can be obtained by using ConEx alone, but also the benefits that can be expected from the combined used of ConEx and LEDBAT. 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.

To conclude, ConEx could be considered as a credible approach to improve user's fairness in case of high network loads, while being transparent otherwise. If ConEx operation is influenced by its environment (e.g., parameters setting, network topology, traffic demand, etc.), in all tested configurations its behavior remains robust, suggesting that reasonable margins exist for a network operator to deploy and provision ConEx in a real network. Considering its poor behavior when applied on short flows and the fact that long flows are the main cause of congestion, ConEx should be focused on long flows. In order to minimize the introduction cost, ConEx should be deployed first in a simple implementation mode, i.e., using only packet losses to estimate congestion. In that mode only the traffic sources (i.e., servers) have to be modified to be able to generate the ConEx signal. In a second phase, if a better level of performance is required, in particular regarding delay, upgrading ConEx could be envisaged, preferably for a full ConEx mode.

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 deployment 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 [8].

References

[1] A. Sanhaji, P. Niger, P. Cadro, and A.-L. Beylot, "DropTail Based ConEx Applied to Video Streaming," ICNS 2015, 2015, pp. 3–10.

[2] 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.

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

[4] B. Briscoe and al., "Congestion exposure (ConEx) concepts and use cases," December 2012. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6789.txt [retrieved: November, 2015]

[5] S. Krishnan and M. Kuehlewind, "IPv6 destination option for congestion exposure," October 2015. [Online]. Available: https://www.ietf.org/id/draft-ietf-conex-destopt-10.txt [retrieved: November, 2015]

[6] D. Kutscher, F. Mir, S. Krishnan, Y. Zhang, and C. Bernardos, "Mobile communication congestion exposure scenario," October 2015. [Online]. Available: https://www.ietf.org/id/draft-ietf-conex-mobile-06.txt [retrieved: November, 2015]

[7] M. Kuehlewind and R. Scheffenegger, "TCP modifications for congestion exposure," October 2015. [Online]. Available: https://www.ietf.org/id/draft-ietf-conex-tcp-modifications-10.txt [retrieved: November, 2015]

[8] B. Briscoe and 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: November, 2015]

[9] 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

[10] 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.

[11] 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.

[12] 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: November, 2015]

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

[14] 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, Aug. 2010, pp. –. [Online]. Available: http://dl.acm.org/citation.cfm?id=2043164.1851192

[15] M. Kuhlewind 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.

[16] B. Briscoe, R. Scheffenegger, and M. Kuehlewind, "More accurate ECN feedback in TCP," October 2015. [Online]. Available: https://www.ietf.org/id/draft-kuehlewind-tcpm-accurate-ecn-05.txt [retrieved: November, 2015]

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

[18] 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: November, 2015]

[19] N. Dukkipati, T. Refice, Y. Cheng, J. Chu, T. Herbert, A. Agarwal, A. Jain, and N. Sutin, "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

[20] 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

[21] 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, 2012, pp. 360–377. [Online]. Available: http://dx.doi.org/10.1002/ett.2546

[22] 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.

[23] 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.

[24] 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

[25] Google, "Advanced encoding settings." [Online]. Available: https://support.google.com/youtube/answer/1722171 [retrieved: November, 2015]

[26] S. Shalunov and al., "Low extra delay background transport (LEDBAT)," December 2012. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6817.txt [retrieved: November, 2015]

[27] 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.