An IP-ERN architecture to enable hybrid E2E/ERN protocol and application to satellite networking

Dino Martin Lopez Pacheco [*,a] Tuan Tran Thai [c]
Emmanuel Lochin [b,c] Fabrice Arnal [d]

[a] *I3S Lab - CNRS UMR 6070 University of Nice, Sophia-Antipolis, France*

[b] *CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse, France*

[c] *Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France*

[d] *Thales Alenia Space, Toulouse, France*

Abstract

We propose an architecture based on a hybrid E2E-ERN approach allowing ERN protocols to be inter-operable with current IP-based networks. Without introducing complex operations, the resulting E2E-ERN protocol provides inter and intra protocol fairness and benefits from all ERN advantages when possible. We detail the principle of this novel architecture, called IP-ERN, and show that this architecture is highly adaptive to the network dynamics and is compliant with every TCP feature, IPv4, IPv6 as well as IP-in-IP tunneling solutions. As a possible use case, we test this architecture as a potential candidate to replace Performance Enhancing Proxies (PEPs) commonly-used over satellite IP-based networks. Compared to splitting PEP, the IP-ERN architecture does not break the E2E connectivity, still achieves high satellite link utilization and fairness without needs of extra fault tolerant mechanisms.

Key words: IP-ERN, PEP-less architecture, XCP

* Part of these results have been presented at the IEEE ICC 2011 conference and PFLDNet 2010 workshop. Corresponding author. Address: Laboratoire d'Informatique, Signaux et Systmes de Sophia-Antipolis - UNSA - CNRS 2000, route des Lucioles - Les Algorithmes - bt. Euclide B - BP 121 - 06903 Sophia Antipolis Cedex - France

Email addresses: dino.lopez@eunice.fr (Dino Martin Lopez Pacheco), tuan.tran-thai@isae.fr (Tuan Tran Thai), emmanuel.lochin@isae.fr (Emmanuel Lochin), fabrice.arnal@thalesaleniaspace.com (Fabrice Arnal).

1 Introduction

TCP New Reno (denoted standard TCP or simply TCP in the rest of the paper) was the dominant protocol in charge of providing congestion control, fair share and full utilization of the network resources. The very large deployment of Internet cannot be explained without the wide utilization of TCP. However, as the link capacity and propagation delay grow, the performance of TCP decreases. Indeed, TCP is known to obtain poor performance in large bandwidth×delay product (LBDP) networks [11].

Following the pervasive deployment of gigabit links or satellite links, LBDP networks are now common in the Internet. To solve the problem of standard TCP over LBDP networks, high speed TCP variants have been proposed such as CUBIC [27] (currently enabled by default in GNU/Linux systems) [1] , Compound TCP (deployed in recent Windows systems) [30], High Speed TCP [11] or more specialized TCP version for LBDP networks that are characterized by long delay such as Hybla TCP [6], specially designed to fit the needs of satellite networks. However, it has been shown in [17,12] that the aggressiveness of high speed TCP variants can lead to congestion events, and their convergence time can be potentially large, increasing the intra/inter-protocol unfairness [2] . Other high speed TCP variants, known as delay-based protocols such as FAST TCP [7], monitor the round-trip time (RTT) at the sender side. Such protocol considers an increase of the RTT as a congestion indicator. Thus, delay-based protocols seek to prevent and react before a congestion event occurs. However, delay-based protocols do not solve the problem of intra/inter-fairness [18,12].

Estimating the bottleneck capacity and updating the Slow-Start threshold to that value is another possible solution to improve the performance of satellite-based networks. Hence, when a loss occur, the congestion window is still able to grab all the network resources. On one hand, for instance, TCP Westwood [28] uses the inter-arrival time of duplicate acknowledgments (ACKs) to estimate the bandwidth at the bottleneck. After the execution of Fast Retransmit/Fast Recovery, depending on the accuracy of the estimation, the congestion window can correctly grab all the network capacity. However, when the network capacity increases because the number of flows that share the bottleneck decreases, TCP Westwood is unable to quickly grab the remaining bandwidth in presence of large RTTs [2]. On the other hand, TCP Peach [13] also estimates the available bandwidth by mean of Dummy Segments (which are low-priority duplicated data packets especially marked). In order to reach as

[1] Although the right name of this protocol is CUBIC we denote it "CUBIC TCP" in the following to prevent misunderstanding with another notation: CUBIC-XCP used in this paper.

[2] The intra-protocol and inter-protocol fairness indicate, respectively, the fairness between flows using either the same or different protocols.

quickly as possible the bottleneck capacity, Dummy Segments are sent during both the Slow-Start phase (renamed Sudden Start) and Fast Recovery phase (renamed Rapid Recovery). The idea is that Dummy Segments will reach the destination in absence of congestion but never in case of congestion, since they are identified in the IP header as low-priority packets. Indeed, Dummy Segments are expected to be immediately dropped once there is no resource left. The main barrier of such approach is that Internet routers must correctly handle the TOS IP field, which is used by TCP Peach. Routers that do not correctly handle the TOS IP field will process Dummy Segments and normal packets with the same priority. Thus, Dummy Segments will take the place of legitimate data packets even during congestion periods. Thus, we believe a deep study about the impact of Dummy Segments in Internet is needed before deploying such a strategy. Like TCP Westwood, TCP Peach does not propose a way to quickly grab the available bandwidth freed by outgoing flows.

In the case of networks with very large delay such as GEO satellite-based networks, the use of splitting Performance Enhancing Proxies (PEPs) [5] is a commonly used solution that improves the performance of standard TCP. PEPs break the end-to-end (E2E) TCP connection in order to emulate a receiver that acknowledges TCP packets before sending them to the long delay link. The splitting PEP gateway is thus responsible for transmitting these acknowledged data over the long-delay link. When a PEP is implemented before the link with large delay, the throughput is increased. Furthermore, between the PEP and the receiver, other transport protocols, more adapted to long delay link than standard TCP, are often used. One of the main barrier of this architecture is that the split of the end-to-end connection prevents the use of security protocols. In the context of privacy protection such as IPSec, PEPs can not be used without introducing complex modifications that satellite network providers have to take in charge [5]. In addition, PEPs might require both high memory capacity to keep connection states and complex fault tolerant mechanisms.

The main challenge for a transport protocol performing over a LBDP network is to achieve the capacity of links and converge as fast as possible. So, a potential transport protocol solution to LBDP networks would be the use of an Explicit Rate Notification (ERN) protocol where the routers inform the sender of the optimal sending rate (for instance the eXplicit Control Protocol - XCP [15]). This kind of protocols demonstrates high performance and intra-protocol fairness in fully ERN-capable networks [15] (i.e. where all routers support ERN capabilities). The major problem to the deployment of such concept is that ERN protocols do not implement any mechanisms to deal with networks where non-ERN protocols (e.g., standard TCP) and non-ERN equipments (e.g., DropTail routers) are present. Indeed, it has been proved that ERN protocols can perform worse than every TCP variant in non-fully ERN-capable networks [19,20]. Therefore, ERN protocols cannot be used in heterogeneous

networks and can not be gradually deployed in the current Internet. Despite several efforts to enable an incremental deployment of ERN protocols over heterogeneous networks, we will see in Section 7 that the proposed solutions do not (or only partially) solve the problems related to the interaction between ERN protocols with non-ERN protocols and non-ERN devices.

Although ERN protocols cannot be gradually deployed, this approach and particularly the use of eXplicit Control Protocol (XCP) [15] received a particular attention by the satellite community. Indeed, a satellite topology can be seen as a bounded network where the edges are defined by the PEPs. As an illustration, the authors in [14] propose the use of splitting PEPs which map TCP flows to XCP flows thus targeting the use of XCP to provide a faster access to satellite links. Some efforts have also been done to assess the benefits and to improve the behavior of XCP in a satellite context. For instance, in [31] the authors propose a revisited version of XCP (named P-XCP) especially designed to enhance XCP performance over satellite links, and [25] studies the impact of Fast Retransmit/Fast Recovery over XCP in large propagation delay links.

In this paper, we propose a novel architecture allowing an incremental deployment of ERN protocols over heterogeneous networks that we call IP-ERN. In this architecture, a sender uses an hybrid E2E-ERN protocol which is able to adapt its emitting rate as a function of a DropTail or an ERN-capable bottleneck. Indeed, when the sender receives an ERN feedback, it uses the minimum between the ERN and E2E congestion windows. Thus, IP-ERN correctly handle congestion events whether the bottleneck is ERN-capable or not.

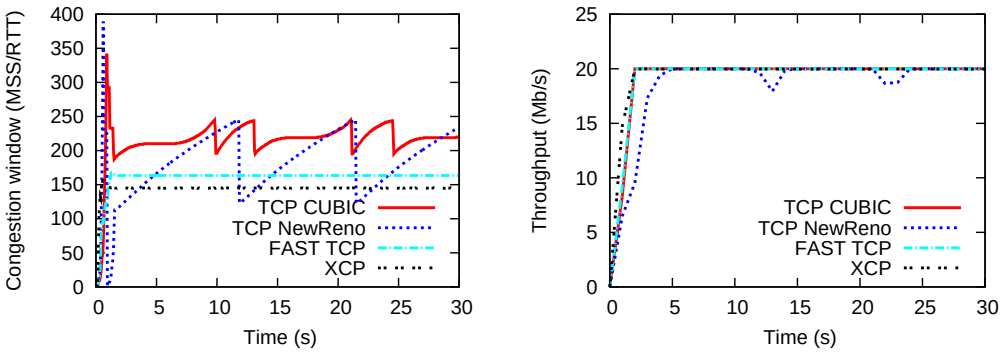
The IP-ERN architecture is presented in Section 2. We explain why our solution is compatible with any TCP variant and TCP add-ons; most of proposed ERN protocols; with IPv4, IPv6 and IP-in-IP tunneling mechanisms; and detail in which context the proposed architecture brings out benefits. Then, we propose a use-case in Section 4, where we illustrate how this architecture can replace efficiently a splitting PEP in the context of a satellite link. We present simulations in order to validate our IP-ERN architecture (Section 5) and show that our solution achieves high performance in terms of link utilization and fairness in satellite IP-based networks. Finally, we present the strengths and weaknesses of other current existing solutions compare to our proposal in Section 7 and conclude this work in Section 8.

2 Presentation of the IP-ERN architecture

2.1 Rationale of IP-ERN architecture

TCP-based protocols frequently probe the network capacity by increasing their emitting rate, hence leading to congestion events. Additionally, high speed TCP variants potentially increase the unfairness. While delay-based protocols attempt to prevent congestion states, they also prevent under-utilization of bandwidth by keeping a certain number of packets in the buffer of the bottleneck. Unfortunately, delay-based protocols do not solve the problems of unfairness [12]. On the contrary, thanks to the assistance provided by routers, ERN protocols provide both high link usage and intra-fairness while minimizing the buffer occupancy. As the buffer occupancy decreases the probability of loosing packets also decreases.

For the sake of giving the intuition of our idea, we illustrate the general behavior of one ERN protocol (XCP) and different E2E protocols (TCP New Reno, CUBIC, FAST TCP) over a network with a bottleneck capacity of 20Mb/s and a base RTT of 30ms in Figure 1. The router queue is XCP in case of XCP and DropTail for other cases. The results are obtained with the *ns-2* simulator [1]. Figure 1(b) shows that all protocols achieve the bottleneck capacity. Moreover, XCP outperforms all TCP variants since XCP uses the smallest congestion window (Figure 1(a)). This means that XCP maximizes the link utilization while minimizing the buffer occupancy. Indeed, when the TCP New Reno congestion window is smaller than the XCP congestion window, TCP New Reno throughput is slightly lower than 20Mb/s. However, when the XCP sender receives misleading congestion window in non-fully XCP-capable network, XCP might perform worse than E2E protocols [19].



(a) Congestion window (b) Throughput

Figure 1. Comparison between E2E and ERN protocols

When the bottleneck is ERN-capable, ERN protocols can compute the optimal congestion window. However, when the bottleneck is not ERN capable, E2E

protocols provide a more adaptive congestion window. Following this, the idea is to execute both E2E and ERN protocols at the same time in a single sender and to use the minimum between the E2E and ERN congestion windows size in order to be compliant with current protocols (i.e., no more aggressive than TCP) while using the optimal congestion window when possible. Note that this approach is completely different from [29] (see arguments in Section 7). Our proposition does not need either to decide between a predicted ERN rate and the real rate seen by the receiver or to apply delay-based mechanisms to detect non ERN-capable routers as in [29].

2.2 Proposed IP-ERN architecture

Our proposition relies in the implementation of two different congestion control protocols in the sender node. Since ERN protocols do not introduce complex operations at the sender side (usually, ERN senders only update the congestion window according to the received feedback without any other heuristic), we believe that current computers have enough resources to run both one E2E and one ERN protocols.

The proposed architecture (denoted IP-ERN architecture in the following) requires only slight modifications at the end-hosts and few additional mechanisms inside the routers, that will be explained in the next subsections. Figure 2 gives a general view of the proposed IP-ERN architecture. Basically our proposal consists in the introduction of a Congestion Aware Layer (denoted CAL) which takes place between the IP and the transport layers. We detail in the following the internal mechanism of this additional layer.

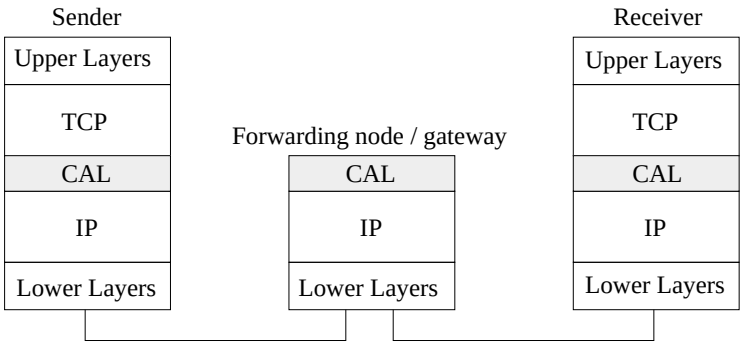


Figure 2. The proposed IP-ERN architecture.

2.2.1 At the sender side

The sender implements both a transport layer and the Congestion Awareness Layer. The transport layer hosts the core of the TCP-based congestion control

protocol while the CAL layer hosts the core of the ERN protocol and all needed mechanisms to properly combine both E2E and ERN capabilities.

When a SYN packet is sent to establish a connection, the CAL layer inserts a TCP option field to indicate an IP-ERN capable sender [3]. Later, at the reception of a SYN-ACK, the sender checks inside the packet a TCP option indicating that the receiver is IP-ERN capable. If so, the E2E-ERN connection is established. Otherwise, a standard E2E connection is created.

After establishing a connection with an E2E-ERN capable receiver, three different options are possible to include the ERN header in every packet:

(1) in IPv6 networks, we propose to create a new Extension Header (EH) which belongs to the upper-layer group to carry up the ERN parameters. Then, CAL places such an EH according to the IPv6 principles. To get an identifier for our protocol will require important discussions at the Internet Engineering Task Force (IETF). Moreover, without such an identifier, IPv6 routers may process the packets by software, and not only by hardware (to pass through the so-called "slow-path") [16], which will potentially delay the packets in the network;
(2) in current IPv4 networks, we propose to encapsulate the ERN header in a TCP packet, that is why we call this ERN-over-TCP. Our main objective is to re-use as much as possible the idea behind DCCP-over-UDP [26]. Hence, a server supporting the IP-ERN architecture, will accept connection through a given TCP port, which may not correspond to the ERN port. Also, the first TCP option will be the same as inserted in the SYN packet in order to signal to IP-ERN routers that this packet comes from an E2E-ERN capable node. We will not detail this proposition that is more related to a technical aspect;
(3) in current IPv4 networks, it is also possible to place the ERN header between the IP header and the TCP header (as suggested in [10]). This should allow routers to quickly find the ERN parameters. However, this solution requires to signal at the IP header a different transport protocol than TCP and UDP, which make cause packets to be dropped by middle boxes or processed by software in legacy routers.

Between these three propositions, we advise the second one, since at the moment, this is only one able to successfully cross legacy middle boxes and avoid the slow-path in routers. Therefore, in the remaining of this article, we will assume that each IP-ERN capable node implement the ERN-over-TCP protocol.

[3] Non IP-ERN capable receivers should only ignore this option and following [23], a majority of TCP stacks correctly handle unknown TCP option (i.e. packets that contain unknown options are not dropped).

Concerning incoming packets, upon reception of an ACK, CAL extracts the ERN feedback to compute the ERN congestion window (*cwnd_ern_*). Finally, once the congestion window of the TCP layer is modified (*cwnd_tcp_*), CAL takes the minimum value according to (1):

$$cwnd\_ = min\{cwnd\_tcp\_, cwnd\_ern\_\} \quad (1)$$

As we will see later, by taking the minimum value between *cwnd_tcp_* and *cwnd_ern_*, this architecture does not require any explicit mechanism to detect whether there are non IP-ERN capable routers in the network. In other words, the detection is automatically done by this comparison. Additionally, the time needed to switch from E2E to ERN capabilities (or reverse) is not longer than one RTT (time to detect a loss when going from ERN to E2E or time to receive the signaling from routers when going from E2E to ERN). Figure 3 presents this new architecture at the sender side.

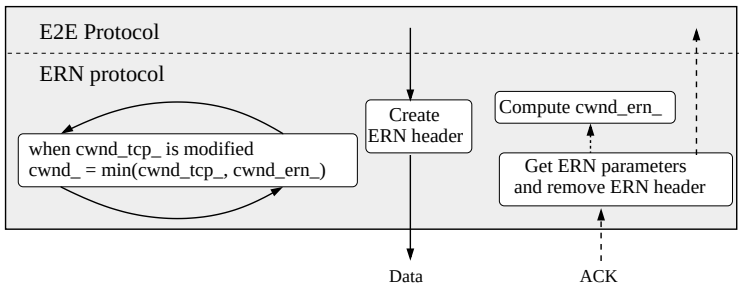


Figure 3. Interaction between the E2E protocol and the ERN protocol at the sender.

We want to point out that this architecture is compatible with any TCP-based congestion control mechanism and with most of existing ERN protocols. Furthermore, as this architecture does not modify the TCP algorithm, this allows the use of new proposed TCP extensions such as the Eiffel algorithm [21] for instance or new configuration such as increasing the initial window size [8].

2.2.2 At the router side

All needed ERN algorithms are placed in the Congestion Awareness Layer. Assuming we use the XCP-over-TCP to deploy the IP-ERN architecture, when a packet arrives at an IP-ERN router, CAL checks if the TCP source port or destination port used is the one reserved to encapsulate ERN in TCP. If this is the case and the first TCP option indicates that the sender is IP-ERN capable (similar to the SYN packet), then the router computes the feedback and updates the ERN header according to the ERN rules. Otherwise, packets are treated as default IP packets.

Since IP-ERN routers only assign bandwidth to E2E-ERN flows (as E2E flows do not interpret the feedback message), and because fairness between E2E and E2E-ERN traffic is enabled by the E2E capabilities of senders using our architecture, each router computes a feedback by taking into account the E2E-ERN traffic only. Indeed, most ERN protocols need to compute the input traffic rate. Therefore, this input traffic rate corresponds to the E2E-ERN traffic only. Also, ERN protocols usually need to estimate the buffer occupancy of ERN flows to decrease their rate and to drain packets in order to prevent congestion. Calculating the buffer occupancy without differentiation between E2E-ERN and pure E2E traffic might decrease IP-ERN senders' rate to drain E2E packets. To better distinguish between pure E2E and E2E-ERN traffic, we propose the following architecture:

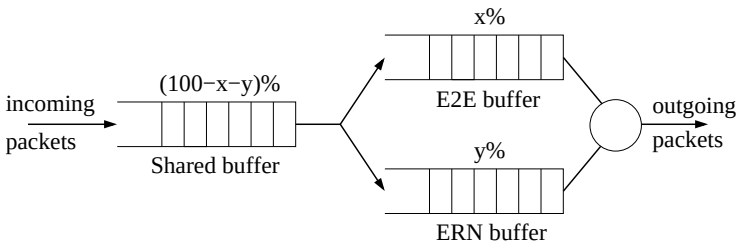


Figure 4. Egress buffer scheme for IP-ERN routers

In IP-ERN routers, there is one main buffer (the shared buffer), serving two non-shared buffers (E2E and ERN buffers with sizes X and Y respectively) as depicted in Figure 4. E2E buffer only stores packets from E2E flows and ERN buffer only stores packets from E2E-ERN flows. The packets from the shared buffer are immediately forwarded to the next buffer, even if the non-shared buffer is full.

Note that in this proposition, IP-ERN routers do not attempt to provide fairness (such as in [29,20]). Inter and intra fairness are ensured by E2E and ERN capabilities of IP-ERN senders.

2.2.3 At the destination side

The Transport Layer at the receiver side implements the same layers than the sender side (i.e. TCP and CAL layers). When a SYN is received, CAL looks at the TCP option field to know if the sender is IP-ERN capable. If so, CAL sends back a SYN-ACK packet with a code in the TCP option field to indicate an IP-ERN capable receiver.

During the connection, upon reception of a data packet, CAL copies the feedback from the data packet to the acknowledgment. CAL builds the outgoing ACK in the same way the sender builds a data packet (e.g. using the ERN-over-TCP strategy).

2.2.4 The IP-ERN architecture in the context of IPsec tunnels

The IP-ERN architecture is fully compatible with IPsec tunnels. Indeed, the encryption and/or authentication of the whole or partial IP datagram by legacy IPsec boxes makes IP-ERN senders to behave like a pure TCP sender. However, in the future, a new generation of IPsec-ERN boxes can be implemented. Indeed, we are currently working on the proposition of the IPsec-ERN architecture, as well as turn around to benefit from ERN capabilities in presence of legacy SatIPsec [9] or legacy IPsec boxes [1] .

3 Benefits of the proposed IP-ERN architecture

In the previous section, we introduced the IP-ERN architecture which allows the deployment of ERN protocols, while eliminating the problems of inter and intra fairness that can appear in non fully ERN networks. Hence, the following scenarios are possible: (*i*) the bottleneck is not IP-ERN capable, (*ii*) only E2E-ERN flows share an IP-ERN capable bottleneck and, (*iii*) both pure E2E and E2E-ERN flows share an IP-ERN capable bottleneck.

3.1 1^{st} scenario: the bottleneck is not IP-ERN capable

IP-ERN hosts should not benefit from the ERN capabilities but they should fully benefit from TCP capabilities.

Consider the topology from Figure 5. If router $R1$ is IP-ERN capable, but not $R2$, then the feedback will reflect the network state at router $R1$, and the ERN congestion window of an E2E-ERN flow will be higher than the TCP congestion window. Thus, according to (1), the congestion window $cwnd\_$ will follow the TCP behavior. We did an ns-2 simulation where we set up the conditions described above. IP-ERN end hosts implemented CUBIC TCP in the TCP layer and XCP in the CAL layer. As a result we call CUBIC-XCP this hybrid protocol. The result of our simulation confirms that CUBIC TCP and CUBIC-XCP have a similar behavior when the bottleneck is non IP-ERN capable (Figure 6).

[1] These propositions will be part of a document entirely focused on the security protocols and IP-ERN architecture interactions

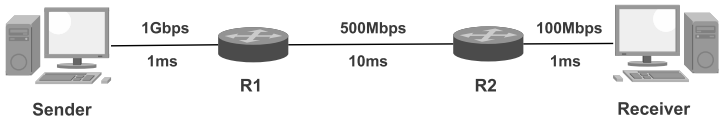
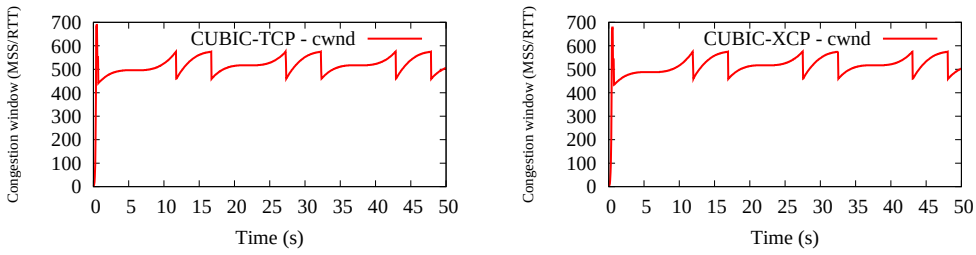


Figure 5. Simple topology



(a) CUBIC TCP (b) CUBIC-XCP

Figure 6. CUBIC TCP and CUBIC-XCP in a non IP-ERN capable bottleneck.

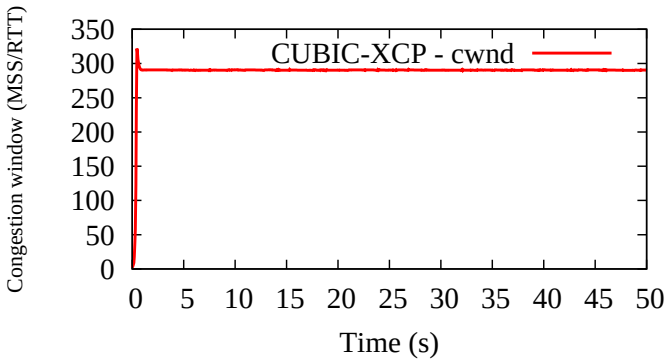


Figure 7. CUBIC-XCP in an IP-ERN capable bottleneck.

3.2 2nd scenario: the bottleneck is IP-ERN capable and only E2E-ERN flows share the resources

Each host implementing the IP-ERN architecture should fully benefit from ERN capabilities. Although this scenario does not correspond to what we usually call an incremental deployment (we consider here only E2E-ERN flows), there are some cases where network administrators might consider this solution to improve the performance of some portions of the network where long lived flows are frequent (e.g., in some parts of a Data Center).

If router $R2$ from the topology described in Figure 5 is also IP-ERN capable, then the feedback will reflect the state at the bottleneck link. Therefore, when TCP will send above the bottleneck capacity, the ERN congestion window will be smaller than the TCP congestion window and the congestion window of the E2E-ERN flow will behave like a pure ERN protocol (see simulation results shown in Figure 7).

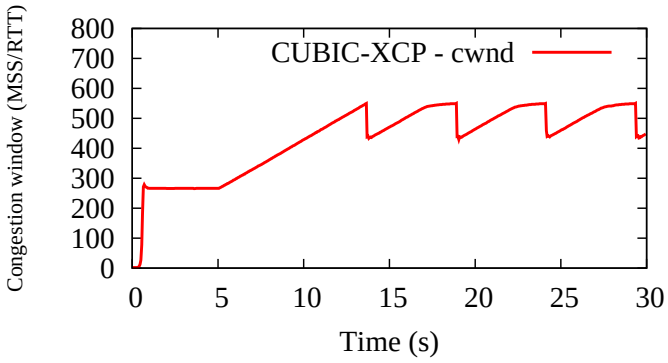


Figure 8. Behavior of IP-ERN in presence of standard TCP and IP-ERN capable routers.

3.3 3rd scenario: the bottleneck is IP-ERN capable and both TCP and E2E-ERN flows share the resources

In this case, IP-ERN hosts will use their TCP capabilities to compete against pure TCP flows. When an IP-ERN router is shared between standard TCP and E2E-ERN flows, the feedback will be calculated taking into account only the E2E-ERN traffic and the optimal rate targeted by the router will be the maximal output link capacity. However, since E2E-ERN flows will be unable to reach the maximal output link capacity due to the presence of pure TCP flows, the router will persistently send positive feedbacks that will inflate the congestion window size of the E2E-ERN flows. Thus, the congestion window of the CAL layer (*cwnd_ern_*) will tend to infinity and according to (1), the congestion window of E2E-ERN flows will be handled by TCP and it will behave like any other TCP source.

Figure 8 shows a simulation result where one CUBIC-XCP flow shares a 100Mb/s bottleneck, which is XCP capable (same topology as shown in Figure 5), with a background traffic limited to only 1Mb/s. During the first 5 seconds, when the E2E-ERN flow is alone in the network, IP-ERN host follows the XCP behavior. However, after second 5 when the background traffic is started, the CUBIC-XCP flow automatically switches to the CUBIC capabilities. Hence, if the background is replaced by a non-limited long lived CUBIC TCP flow, both CUBIC-XCP and CUBIC TCP will converge to the fairness using the CUBIC TCP properties.

Last simulation results shows that IP-ERN sources will always use its TCP capabilities to enable the inter-fairness (i.e. fairness between IP-ERN and non-IP-ERN flows). Moreover, additionally in this case, IP-ERN can benefit from the ERN capabilities to improve the intra-fairness, even in presence of standard TCP flows. The intra-fairness can be highly improved by E2E-ERN where pure E2E flows will not. This property depends directly on the kind of ERN

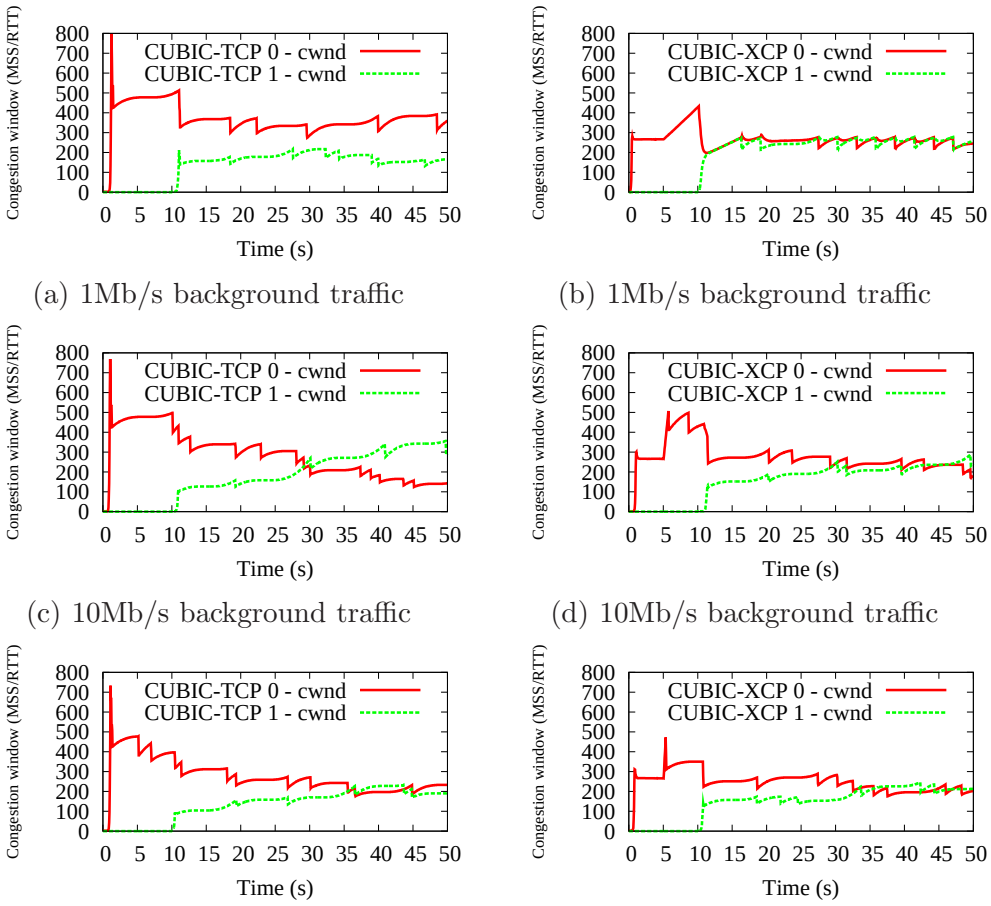
protocol used by IP-ERN sources.

Let us explain how it works for XCP, which is the protocol used in our analysis. In [4], the authors proved that the fairness principles of XCP can introduce an under-utilization at the bottleneck of 19% in the worst case. This under-utilization is caused by limited XCP flows which can be present at any part of the network. Indeed, the throughput of one flow can be limited to 81% of the bottleneck capacity in order to allow other flows to increase their throughput. Coming back to IP-ERN, it means that when E2E-ERN flows have reached a throughput higher than 81% (assuming that the remaining bandwidth is taken by pure E2E traffic), even though sources have an inflated congestion window, if a new IP-ERN flow come into the network, IP-ERN sources already present in the network will receive negative feedbacks, which can potentially lead to a faster convergence. As the foreign traffic throughput get closer zero, the intra-fairness increases.

Figure 9 shows the convergence time between two flows using CUBIC TCP or CUBIC-XCP when the background traffic takes 1% (1Mb/s, referred like the first case), 10% (10Mb/s, second case) or 20% (20Mb/s, third case) of the bottleneck capacity. In the first case, the benefits from XCP is easily visible if we compare the IP-ERN convergence against the CUBIC TCP convergence. Indeed, at second 10, flow labeled "CUBIC-XCP 0" decreases its congestion window before experiencing losses. Thus, "CUBIC-XCP 0" and "CUBIC-XCP 1" quickly converge and keep similar congestion window sizes. Our logs show that the first dropped packet of "CUBIC-XCP 0" occurs at second 16. CUBIC TCP flows are not able to convergence this time. The convergence time of CUBIC TCP is difficult to predict and depends on the amount of packets lost by each flow sharing the bottleneck.

In the second case, CUBIC-XCP has more problems to converge, since it receives less negative feedbacks (the background traffic is limited to 10% of the bottleneck capacity). Moreover, after "CUBIC-XCP 0" is aware that a new flow is present, "CUBIC-XCP 0" reduces its congestion window before experiencing losses (i.e. around second 10). We repeated this simulations several times and we observed that "CUBIC-XCP 0" always begins reducing its congestion window before experiencing losses. We never observed a window reduction due to negative feedbacks in "CUBIC-XCP 0" after the congestion window of "CUBIC-XCP 1" reached around 50 MSS (Maximum Segment Size), meaning that when the congestion window of "CUBIC-XCP 1" is higher than such value, the behavior of CUBIC-XCP flows is driven by CUBIC TCP. Concerning CUBIC TCP, flows were not able to converge during the simulation of 50 seconds.

Finally, in the third case, when the background traffic takes 20% of the bandwidth, E2E-ERN flows do not benefit from the ERN capabilities to converge,



(a) 1Mb/s background traffic

(b) 1Mb/s background traffic

(c) 10Mb/s background traffic

(d) 10Mb/s background traffic

(e) 20Mb/s background traffic

(f) 20Mb/s background traffic

Figure 9. Impact of background traffic on the convergence properties of CUBIC-XCP (right side) and CUBIC TCP (left side).

but their behavior is fully handled by TCP.

4 An application of the IP-ERN architecture for satellite networks

The proposed IP-ERN architecture can greatly improve the performance of flows in the following cases:

- in a Virtual Private Network (VPN) with frequent long-lived flows, assuming (*i*) we place an IP-ERN router at the entry point; (*ii*) most of the time the VPN tunnel encompasses the bottleneck link and (*iii*) most of senders and receivers implement the IP-ERN architecture;
- in a satellite scenario, assuming (*i*) we place an IP-ERN capable router at the gateway level and terminal level and (*ii*) most of senders and receivers implement the IP-ERN architecture.

In the remaining of this section we describe with more details how can be used the IP-ERN architecture in satellite scenarios.

Usually, when a flow of data is carried through satellite links, the performance of such a flow is improved by Performance Enhancing Proxies, as illustrated in Figure 10. The rationale behind is that if this flow is handled by standard TCP, it can face two major issues that directly impact on the overall performance of TCP: (*i*) the long delay inherent to satellite links and (*ii*) losses due to congestion that will require a retransmission from the source.

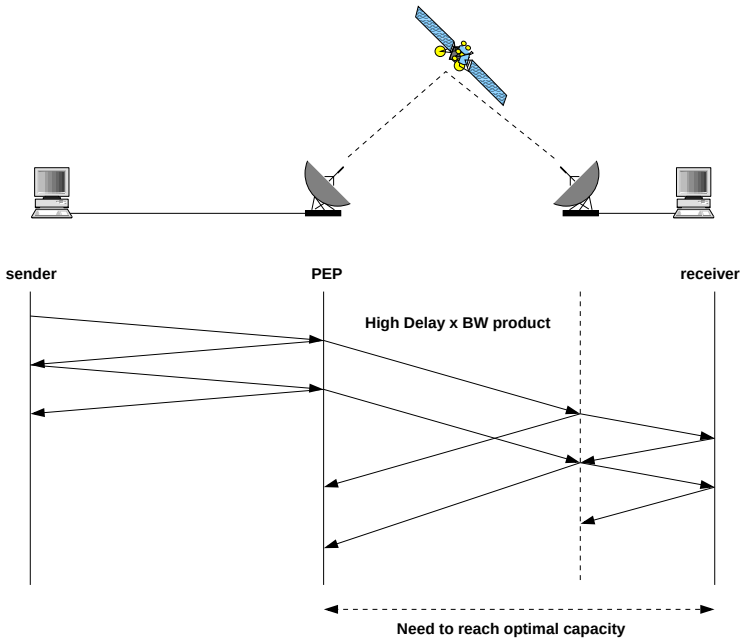


Figure 10. Standard PEP architecture

The PEP architecture optimizes the transfer by using a transport protocol specially designed for links with large propagation delay (e.g., SCPS-TP [3], TCP-Hybla [6]) and performing the retransmissions of packets lost to avoid as much as possible retransmission from the source. However, this architecture splits the connection and, as explained in the introduction, this splitting prevents the use of privacy protection protocols like IPSec; the establishment of an encrypted VPN. Also, splitting PEPs require complex fault tolerant mechanisms and enough resources (i.e. CPU and memory) to efficiently map the state and data of the sender at the splitting proxy.

In this article, we propose the use of our IP-ERN architecture in satellite-based networks and benefit from the ERN capabilities as depicted in Figure 11. We will refer to satellite-based networks with IP-ERN capabilities like SatERN for abbreviation. SatERN aims at substituting current PEPs in the satellite architecture. Thus, the objective is to propose a method to efficiently reach the resources and fairness.

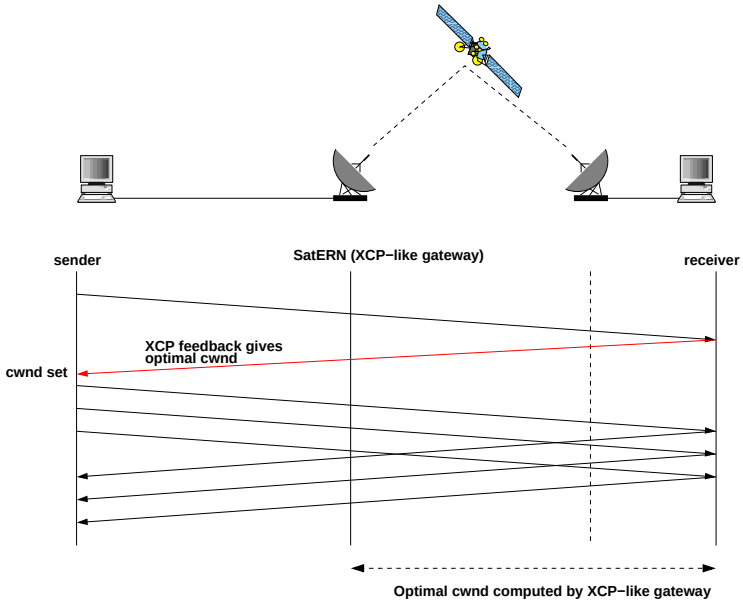


Figure 11. SatERN architecture

The optimal window size is computed in the SatERN gateway (which acts as an ERN-like router) then put inside the feedback messages. The sender retrieves this information and takes the minimum between the congestion window given by the ERN feedback and the actual TCP congestion window. This architecture does not need a full deployment of ERN routers on the entire path as the satellite link is the only one subject to the ERN congestion window computation.

To analyze the performance of SatERN, we have designed in the ns-2 network simulator an adapted version of the XCP protocol to emulates at the sender side an XCP-like service that would only respond to XCP-like gateways connected to satellite links. This XCP service would allow a host to transparently use an Internet connection like DSL or satellite without having to save a specific configuration for each connection context. Note that SatERN is compliant with most of ERN protocols and the choice of the ERN protocols is under the responsibility of the network administrator. The XCP-like service implemented here is used in combination with either CUBIC or FAST TCP.

5 Results and analysis

We propose to demonstrate the capability of the proposed approach within three scenarios. The first one verifies that TCP variants used with our SatERN framework behave like XCP protocol when possible and are able to correctly grab the available bandwidth. The second one tackles the intra-fairness of flows in order to assess whether arriving supplementary flows do not disturb

the previous ones. Then, we evaluate the inter-fairness when E2E-ERN flows share the link capacity with pure E2E flows. Finally, we evaluate our solution over a dynamic network.

The satellite-based network topology used in our simulation (Figure 12) has a base RTT of 640ms and a satellite capacity of 1Mb/s. The buffer size of satellite gateway is fixed to 20 packets and the packet size is 1000 bytes.

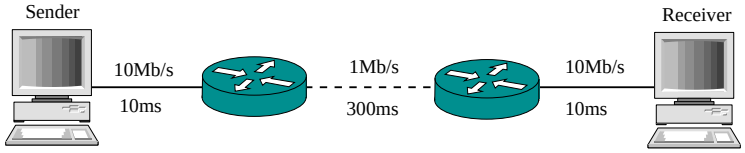
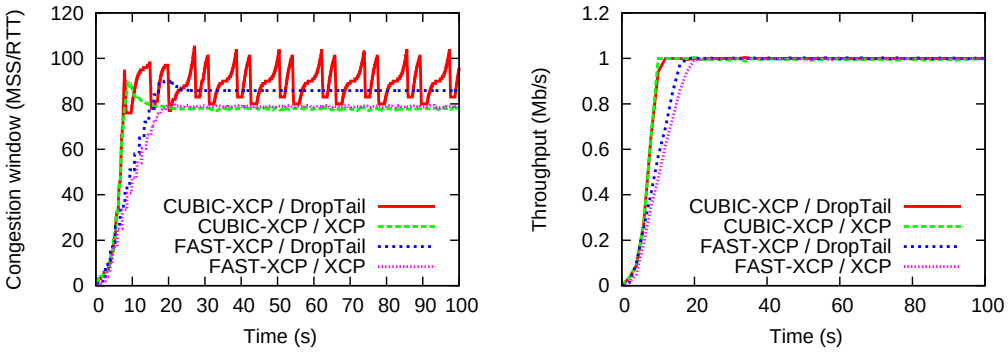


Figure 12. Satellite network topology

5.1 Correctness of the SatERN solution



(a) Congestion window (b) Throughput

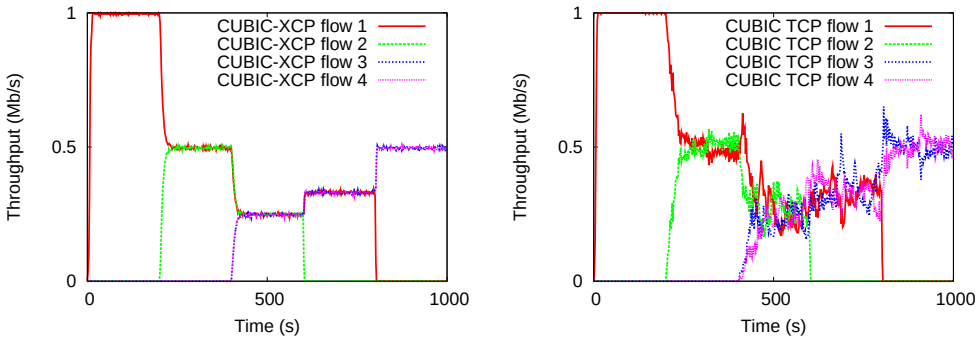
Figure 13. CUBIC-XCP and FAST-XCP with DropTail and XCP queue

With the network settings as in Figure 12, we perform four different experiments. In the first and second experiments, we run a CUBIC-XCP flow when bottleneck router is, respectively, XCP and DropTail. For the third and fourth experiments, FAST-XCP flow is sent when the bottleneck router is, respectively, XCP and DropTail. By taking the minimum value between E2E and ERN congestion windows, we switch between ERN or E2E behavior depending on the network conditions. As a result, CUBIC-XCP and FAST-XCP can be considered as E2E-ERN protocols which behave like ERN in the possible cases where the bottleneck router is IP-ERN capable and only E2E-ERN flows are present in the bottleneck. Otherwise, E2E-ERN protocols use their E2E capability to compete against other flows. The results in Figure 13 show that FAST-XCP and CUBIC-XCP protocols behave like FAST TCP and CUBIC TCP, respectively, in case the bottleneck router is DropTail. They act as XCP protocol in case the bottleneck router is XCP. This simulation shows that when the sender receives the misleading ERN information, it uses its E2E

capability for the connection where the misleading information in pure ERN protocol causes poor performance [19]. As CUBIC TCP is currently enabled by default in GNU/Linux, we only present simulation results with CUBIC-XCP from now.

5.2 Intra and inter-fairness

The aim of this simulation is to show the intra-fairness of CUBIC-XCP protocol. In the experiment, four CUBIC-XCP flows start and terminate at different time. In Figure 14(a), when new flows enter or old flows leave the bottleneck, the remaining CUBIC-XCP flows quickly converge to the new fairness line and are stable since then. In the presence of only CUBIC-XCP flows and the satellite gateway with XCP capabilities, CUBIC-XCP behaves like XCP protocol which provides good intra-fairness property as shown in [15] and confirms again the benefits of IP-ERN explained in Section 3.2. For comparison purpose, Figure 14(b) gives the result obtained without SatERN and clearly highlights both the slow convergence of CUBIC TCP flows and their oscillating behavior.



(a) CUBIC-XCP (b) CUBIC TCP

Figure 14. Intra-fairness

We also verify if CUBIC-XCP flows are not disturbed when the number of CUBIC TCP flows increases. In order to show this inter-fairness between both flows, we perform 100 experiments. In each experiment, we let CUBIC-XCP flows (ranged from 1 to 10) compete against CUBIC TCP flows (ranged from 1 to 10) with a duration of 1000 seconds. CUBIC-XCP and CUBIC TCP flows start at the same time under the same conditions. We remark that the observed link utilization is 100% for all experiments. We calculate the Jain's Fairness Index (JFI) for each experiment according to (2):

$$J(x_1, x_2, ..., x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n * \sum_{i=1}^{n} x_i^2} \tag{2}$$

where n is the number of competing flows in each experiment and $\{x_i, i \in [1, n]\}$ is the average throughput of flow i^{th} during 1000s. The fairness index closer to 1 indicates the better fairness among competing flows and vice versa. For the whole experiment, the JFI is around 0.95 allowing us to conclude that CUBIC-XCP flows fairly share the available bandwidth with CUBIC TCP flows.

5.3 Dynamic scenario

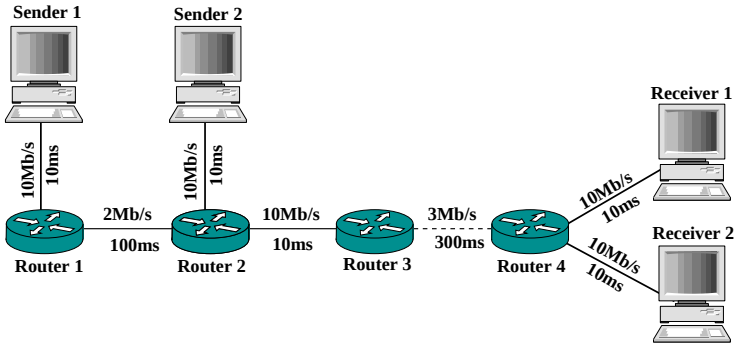
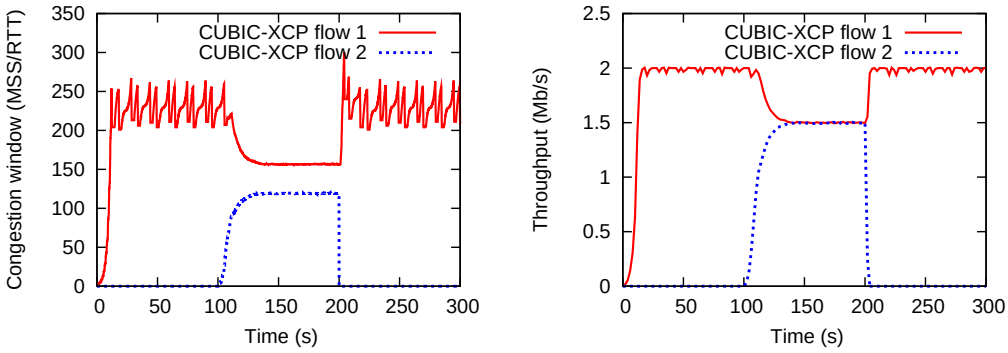


Figure 15. Network topology for dynamic scenario



(a) Congestion window (b) Throughput

Figure 16. Dynamic property for CUBIC-XCP

The aim of this simulation is to show the dynamic property of CUBIC-XCP using the network topology in Figure 15. The base RTT is 860ms for flow 1 between Sender 1 and Receiver 1 and 660ms for flow 2 between Sender 2 and Receiver 2. Queue type of all routers is XCP except Router 1 which is DropTail. CUBIC-XCP flow 1 starts at 0s, the bottleneck at this time is in Router 1. As shown in Figure 16, CUBIC-XCP flow takes link capacity of Router 1. When CUBIC-XCP flow 2 starts from Sender 2 at 100s, the bottleneck is now moved to Router 3. Since Router 3 is XCP, CUBIC-XCP flow 1 switches now to XCP mode. Both flows behave like XCP and fairly share 3Mb/s. It is noted that flow with larger RTT (CUBIC-XCP flow 1) is not penalized since CUBIC-XCP inherits good intra-fairness property of XCP. At 200s, CUBIC-XCP flow

2 stops, the bottleneck is now moved back to Router 1 and CUBIC-XCP flow 1 switches back to CUBIC mode. It has been shown in this simulation that the CUBIC-XCP automatically switches between CUBIC TCP or XCP modes depending on the network condition. This switch is transparently performed by the minimum comparison without any explicit notification mechanism.

6 Dealing with high bit error rates

Today and following the DVB-S standard, in clear weather condition, a satellite link is considered as mostly error free. Indeed, DVB-S standard reports $BER \approx 10^{-7}$ while in DVB-S2 $BER \approx 10^{-10}$. Thus, potential losses result from mobility which is out of the scope of this study. However, it is shown that in exceptional rain events (during heavy storms) and in high frequency band (Ka and above), the DVB-S2 ACM modes can not cope with the deepest attenuations (minimum required E_s/N_0: -2.35 dB with long frames for QEF BER -Quasi Error Free Bit Error Rate-) [22].

Figure 17 shows a bandwidth variation pattern obtained from a real satellite link equipped with ACM (Adaptive Coding and Modulation), by courtesy of Thales Alenia Space, during a heavy raining event. The collected data shows that the hardest bandwidth reduction is around 1Mbps at second 9. Also, the bandwidth is reduced down to nearly 0 Mbps by steps of 0.5 Mbps and 0.3 Mbps beginning from second 124. The bandwidth restarts to grow by steps of 0.5Mbps and 0.3Mbps at second 193. Both bandwidth increases and decreases are spaced by two seconds intervals.

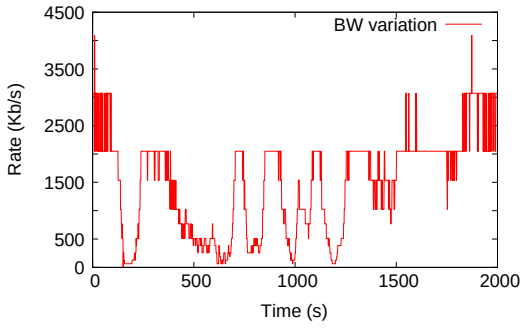
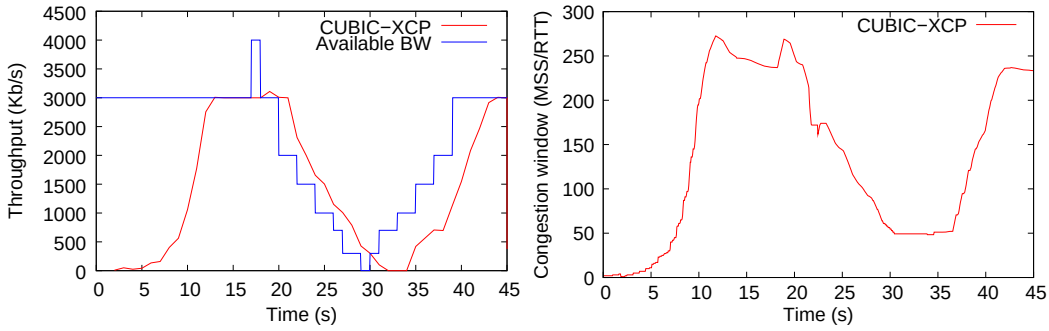


Figure 17. Bandwidth variation observed under hard weather conditions (by courtesy of Thales Alenia Space).

Following Figure 17, we evaluate the performance of the proposed SatERN solution in a highly-variable bandwidth environment. Figure 18(a) shows the bandwidth variation pattern used in our simulation and the instantaneous throughput seen by the receiver, while Figure 18(b) shows the congestion window of the sender. The base RTT of the topology is set to 640ms and the queue size at the bottleneck to 75 packets.



(a) Throughput (b) Congestion window

Figure 18. CUBIC-XCP in a highly-variable bandwidth environment.

The results show that the throughput of the sender is able to follow the bandwidth variations and the number of lost packets is kept at minimum: 13 lost packets in the simulation presented in this document. In our simulation, the lowest available bandwidth is 2Kbps, which is the minimum value to successfully send a single packet with a base RTT of 640ms in the simulator. In Figure 18(a), when the bandwidth decreases, the decrease of the throughput occurs a little bit more than one second after. This behavior is explained by the fact that we compute the throughput at the receiver each second, which introduces a delay of around 0.32s (time needed to the receiver to see the bandwidth reduction and is equivalent to the one-way propagation delay) plus one second (time to propagate the estimated throughput in our logs). At each bandwidth reduction, the buffer in the bottleneck stores the overload. The buffer occupancy decreases as XCP reduces the sender's congestion window. When the bandwidth increases, the bandwidth variation and the measured throughput is spaced by around four seconds. This delay depends on the same factors listed above for the case of bandwidth reduction, plus an extra delay, which depends on the time needed by XCP to update the congestion window of the sender and the window growth of CUBIC. Indeed, the increase of the throughput at $t = 40$ is similar to the one observed at $t = 10$.

Although not graphically shown, we also drive simulations with CUBIC-TCP and observed for both CUBIC TCP and CUBIC-XCP similar goodputs. However, CUBIC TCP experienced 352 lost packets, which shows that CUBIC-XCP is more friendly than CUBIC TCP with the network. Additionally, SatERN flows should better share the network resources regardless the E2E protocol used in the stack.

Before ending this section, we want to highlight that SatERN is compatible with any E2E protocol or E2E features. Thus, any solution to speed up the connexions during the Slow Start phase or the Congestion Avoidance phase of TCP can be used. If each end-host that share the satellite link is SatERN capable, then in all cases, they will fairly share the resources, like shown in Section 5.

7 Related Work: bringing ERN protocols to real networks

To provide TCP-friendliness (i.e., inter-protocol fairness), the authors in [20] proposed to probabilistically estimate the number of ERN and non-ERN flows (by the mean of a zombie list as described in [24]) to determine if non-ERN flows use more bandwidth than ERN ones. If so, an amount of non-ERN packets are probabilistically dropped. Later in [29], the authors improved this strategy and added an algorithm to calculate the aggressiveness of non-ERN flows (how much bandwidth obtain non-ERN flows over a period of time).

However, one weakness of the zombie estimator is its accuracy. For instance, flows with a short congestion window might not always be detected. Additionally, each router needs a period of estimation (denoted t_{est}) to perform the flow number computation. Furthermore, the estimated value will only be applied in the following t_{est} period. This latency between the estimation and the modification of the router behavior leads to some potential problems: (i) the needed time to modify the behavior of the sender is roughly $2 * t_{est} + RTT$ in the best case (note that standard TCP reacts in only one RTT in case of congestion); (ii) while in some specific situations (e.g. a LAN without incoming traffic from external networks), the number of flows might remain stable during $2 * t_{est} + RTT$ seconds or more. We believe further analysis is needed before applying this assumption to highly dynamic environments. This problem of latency between the estimation and its use also applies to the aggressiveness estimation proposed in [29]. Furthermore, heterogeneous RTTs might lead to a sensible difference between the computed aggressiveness and the aggressiveness of the flow with the largest RTT.

Concerning the cohabitation between ERN flows and non-ERN equipments, [29] proposed a heuristic to detect bottlenecks with non ERN capabilities and switch up to TCP when this case occurs. With this heuristic, if the current RTT reaches twice the base RTT, or the receiving rate does not match the ERN predicted rate, then the bottleneck is assumed to be non-ERN capable. Once again, further analysis is needed to assess whether the RTT would reach the threshold before experiencing losses due to congestion in non-ERN routers. Defining upper and lower bounds to compare both the received and the predicted rate is not trivial, thus, this heuristic might frequently return false-positive results. At last but not least, there is no way to detect when the bottleneck has moved to an ERN router again. Following the dynamic characteristic of the Internet, the sender might not correctly take advantage of ERN capabilities when possible.

8 Conclusion and future work

We have presented a novel architecture for heterogeneous networks, based on a E2E-ERN protocol which does not require a full deployment. In this architecture, a sender is able to adapt its emitting rate as a function of either a DropTail or an IP-ERN capable bottleneck. This operation is done dynamically following an ERN feedback allowing the sender to use the minimum between the ERN and E2E congestion window. Compare to other proposals, the main advantage of this solution is that the congestion window adapts dynamically when the bottleneck moves from a DropTail to an IP-ERN routers and conversely.

The resulting IP-ERN architecture is compatible with most of TCP variants; most of proposed ERN protocols; with IPv4, IPv6 and IP-in-IP tunneling mechanisms; and present interesting benefit for long-delay link as illustrated with the PEP-less architecture. The use of IP-ERN in such context allows to efficiently grab the available resource of the satellite link without splitting the end-to-end connection. Furthermore, we have also demonstrated that our proposal allows intra and inter protocol fairness and does not require fully IP-ERN capable networks.

We are now considering a real implementation of this proposal and expect to compare the performance obtained

References

[1] The Network Simulator http://www.isi.edu/nsnam/ns/index.html.

[2] Testing tcp westwood+ over transatlantic links at 10 gigabit/second rate. In *PFLDNet*, Lyon, France, February 2005.

[3] CCSDS 714.0-B-2. Consultative committee for space communications, recommendation for space data system standards, space communications protocol sspecification transport protocol (SCPS-TP), October 2006. Tech. Rep.

[4] Lachlan L. H. Andrew, Steven H. Low, and Bartek P. Wydrowski. Understanding xcp: equilibrium and fairness. *IEEE/ACM Transactions on Networking*, 17:1697–1710, 2009.

[5] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. RFC 3135 (Informational), June 2001.

[6] Carlo Caini and Rosario Firrincieli. TCP hybla: a TCP enhancement for heterogeneous networks. *International Journal of Satellite Communications and Networking*, 22, 2004.

[7] David X. Wei Cheng Jin and Steven H. Low. FAST TCP: Motivation, Architecture, Algortihms, Performance. In *IEEE INFOCOM*, March 2004.

[8] N. Dukkipati et al. An argument for increasing TCP's initial congestion window. *ACM SIGCOMM Computer Communication Review*, 40(3), July 2010.

[9] L. Duquerroy, S. Josset, O. Alphand, P. Berthou, and T. Gayraud. SatIPSec: an optimized solution for securing multicast and unicast satellite transmissions. In *AIAA International Communications Satellite Systems Conference*, 2004.

[10] A. Falk, Y. Pryadkin, and D. Katabi. Specification for the Explicit Control Protocol (XCP). Internet Draft (expired in May 9, 2007), November 2006.

[11] S. Floyd. HighSpeed TCP for Large Congestion Windows. RFC 3649 (Experimental), December 2003.

[12] Sangtae Ha, Yusung Kim, Long Le, Injong Rhee, and Lisong Xu. A step toward realistic performance evaluation of high-speed tcp variants. *Elsevier Computer Networks*, 2006.

[13] Akyildiz I.F., Morabito G., and Palazzo S. Tcp-peach: a new congestion control scheme for satellite ip networks. *IEEE/ACM Transactions on Networking*, 9:307–321, June 2001.

[14] A. Kapoor, Aaron Falk, Theodore Faber, and Y. Pryadkin. Achieving Faster Access to Satellite Link Bandwidth. In *IEEE INFOCOM*, 2006.

[15] D. Katabi, M. Handley, and C. Rohrs. Congestion Control for High Bandwidth-Delay Product Networks. In *ACM SIGCOMM*, 2002.

[16] D. Katz. IP Router Alert Option. RFC 2113 (Proposed Standard), February 1997. Updated by RFC 5350.

[17] Junsoo Lee and Stephan Bohacek et al. A study of tcp fairness in high-speed networks. Technical report, University of California, Santa Barbara, April 2005. Tech. Rep.

[18] Douglas Leith, Lachlan Andrew, Tom Quetchenbach, Robert Shorten, and Kfir Lavi. Experimental evaluation of delay/loss-based tcp congestion control algorithms. In *PFLDNet*, 2004.

[19] Dino Lopez Pacheco, Congduc Pham, and Laurent Lefèvre. XCP-i : eXplicit Control Protocol for heterogeneous inter-networking of high-speed networks. In *IEEE GLOBECOM*, San Francisco, California, USA, November 2006.

[20] Dino Lopez Pacheco, Congduc Pham, and Laurent Lefèvre. Fairness issues when transferring large volume of data on high speed networks with router-assisted transport protocols. In *In Proceedings of High Speed Networks Workshop 2007, in conjunction with IEEE INFOCOM*, Anchorage, Alaska, USA, May 2007.

[21] Reiner Ludwig and Randy H. Katz. The eifel algorithm: making tcp robust against spurious retransmissions. *ACM SIGCOMM Computer Communication Review*, 30(1):30–36, January 2000.

[22] Vazquez M.A. and Pradas D. Voip cross-layer control for hybrid satellite wimax networks. *IEEE Wireless Communications Magazine special issue on Wireless Technologies Advance for Emergency Rural Communications*, June 2008.

[23] Alberto Medina, Mark Allman, and Sally Floyd. Measuring interactions between transport protocols and middleboxes. In *ACM SIGCOMM conference on Internet measurement*, 2004.

[24] Teunis J. Ott, T. V. Lakshman, and Larry H. Wong. SRED: Stabilized RED. In *IEEE INFOCOM*, pages 1346–1355, 1999.

[25] Dino Martin Lopez Pacheco and Emmanuel Lochin. Optimal Configuration for Satellite PEPs using a Reliable Service on Top of a Routers-Assisted Approach. In *International Workshop on Satellite and Space Communications*, Siena-Tuscany, Italy, September 2009.

[26] T. Phelan. Datagram Congestion Control Protocol (DCCP) Encapsulation for NAT Traversal (DCCP-NAT). Internet Draft (expires: September 10, 2011), March 2011.

[27] Injong Rhee and Lisong Xu. CUBIC: A New TCP-Friendly High-Speed TCP Variant. In *International Workshop on Protocols for Fast Long-Distance Networks*, February 2005.

[28] Mascolo S., Casetti C., Gerla M., Sanadidi M.Y., and Wang R. TCP westwood: Bandwidth estimation for enhanced transport over wireless links. In *ACM MOBICOM*, Rome, Italy, July 2001.

[29] Chia-Hui Tai, Jiang Zhu, and N. Dukkipati. Making large scale deployment of rcp practical for real networks. In *IEEE INFOCOM*, April 2008.

[30] Kun Tan, Jingmin Song, Qian Zhang, and Murari Shridaran. Compound TCP: An Scalable and TCP-Friendly Congestion Control for High-Speed Networks. In *IEEE INFOCOM*, Barcelona, Spain, April 2006.

[31] Kaiyu Zhou, Kwan Yeung, and Victor Li. P-XCP: a transport layer protocol for satellite IP networks. In *IEEE GLOBECOM*, November 2004.