Impact of Delayed Acknowledgments on Transport Layer Performance over Low Orbit Constellations

Bastien Tauran, Emmanuel Lochin, Jérôme Lacan, Fabrice Arnal, Mathieu Gineste, Nicolas Kuhn

Impact of Delayed Acknowledgments on Transport Layer Performance over Low Orbit Constellations

Bastien Tauran*
Université de Toulouse, 31500 Toulouse, France
Emmanuel Lochin† and Jérôme Lacan‡
Université de Toulouse, 31400 Toulouse, France
Fabrice Arnal§ and Mathieu Gineste¶
Thales Alenia Space, 31100 Toulouse, France
and
Nicolas Kuhn**
Centre National d'Etudes Spatiales, 31400 Toulouse, France

DOI: 10.2514/1.A34190

Satellite transmissions can suffer from high channel impairments, especially on the link between a satellite and a mobile end user. To cope with these errors, physical and link-layer reliability schemes have been introduced at the price of an increased end-to-end delay seen by the transport layer [for example, Transmission Control Protocol (TCP)]. By default, the TCP enables delayed acknowledgment (DelAck) that might increase the end-to-end delay when performing over satellite link-layer recovery schemes. As a matter of fact, even if this option enables decreasing the feedback path load and the stack processing overhead, it might be counterproductive in a satellite context. This motivates the present paper, which aims to quantify the impact of such a TCP option in the context of low Earth orbit satellite constellations. Several simulation measurements are performed with two well-deployed TCP variants, and it is shown that DelAck should be disabled when used over link-layer Hybrid Automatic Repeat Request schemes, particularly when these schemes enable reordering the buffer.

I. Introduction

LOW earth orbit (LEO) satellite constellations allow us to cope with digital fracture by providing Internet access to isolated or rural areas. The delays of LEO constellations are in the same order of magnitude as terrestrial links that authorize the use of a transmission control protocol (TCP) as a transport protocol without deploying Proxy Enhanced Proxy (PEP) systems [1]. Moreover, the new satellite constellations[††],[‡‡] have a broadband goal, in which part of the TCP is expected to be significant [2] but the high channel constraints, mostly on land mobile satellite (LMS) channels [3], may induce challenges for end to end protocols.

Reliability mechanisms have been introduced on the LMS channel [4 6] to counteract the high potential error rate on this link. One of the most efficient is the Hybrid Automatic Repeat Request (HARQ), which combines forward error correcting coding and link layer retransmission. In our evaluations, such a mechanism is deployed between the last satellite on the route and the ground receiver. We select, in this paper, the type II HARQ, which is commonly deployed over the physical layer. The HARQ can retransmit data on the LMS channel, causing higher end to end delay and jitter that directly impact on the TCP. This may result in a decrease of the throughput of the TCP. To improve TCP performance, a low layer reordering mechanism is proposed [7] to mitigate the jitter caused by the HARQ. However, this buffer might increase the end to end delay, which is unfortunately high due to the path length and the possible HARQ retransmissions. At the transport layer, delayed acknowledgments (DelAcks) [8] are also introduced to improve TCP performance, but they mostly consider terrestrial communication links. This mechanism is now activated by default over each TCP stack. DelAck reduces the number of acknowledgments sent by the TCP receiver, making some packets acknowledged later. Although this scheme decreases the number of acknowledgments, and thus the feedback path load, it also increases end to end performance. In particular, DelAck decreases the CPU load [9] as the number of acknowledgments processed decreases, explaining why this option is activated by default. In short, these reliability mechanisms exhibit a tradeoff between reliability and delay. However, when deployed at different layers, their conjoint use can lead to counterproductive effects that need to be investigated and are tackled throughout this paper.

This paper assesses whether the TCP DelAck option improves TCP performance when used conjointly with layer two reliability mechanisms, in the context of mobile users of LEO constellations. We found only a few studies investigating the impact of DelAck. Chen et al. [10] studied the impact of DelAck on a TCP over wireless links and showed that their impact on TCP performance depended on 1) the topology used, and 2) the path length. They concluded that activating DelAck did not always improve the TCP throughput. In the context of satellite systems, Wood et al. [11] also raised some concerns about the use of such a mechanism. As losses on satellite constellations might be due to transmission errors on the LMS channel and not to congestion, such particular loss patterns and their corresponding recovery mechanisms might have an impact on the consistency of using DelAcks. All these reasons motivate the present study, which aims at investigating this default TCP option and filling

*Ph.D. Student, ISAE SUPAERO, TéSA, 7 Boulevard de la Gare; bastien.tauran@isae supaero.fr.

†Professor, ISAE SUPAERO, 10 Avenue Edouard Belin; emmanuel. lochin@isae supaero.fr.

‡Professor, ISAE SUPAERO, 10 Avenue Edouard Belin; jerome.lacan @isae supaero.fr.

§Research Engineer, Thales Alenia Space, 26 Avenue Jean François Champollion; fabrice.arnal@thalesaleniaspace.com.

¶Research Engineer, Thales Alenia Space, 26 Avenue Jean François Champollion; mathieu.gineste@thalesaleniaspace.com.

**SATCOM Research Engineer, CNES Telecommunication System Department, 18 Avenue Edouard Belin; nicolas.kuhn@cnes.fr.

††Iridium NEXT [online database], Iridium, McLean, VA, 2017, https:// www.iridium.com/network/iridium next/ [retrieved 09 January 2018].

‡‡"OneWeb," 2018, http://www.oneweb.world/ [retrieved 09 January 2018].

the gap by explaining the rationale in using or not using this mechanism over LEO systems.

We present, in the following, the scenario used. We first describe the satellite environment; then, we introduce the low layer reliability mechanisms used to deal with the high channel impairments on this environment; and then, we introduce the TCP versions chosen and the DelAck option. We study the impact of DelAck on TCP performance in our scenario in Sec. III, and then we explain the results obtained in Sec. IV.

II. Scenario

This section presents the satellite scenario, how we simulate the satellite environment, and the different schemes that are considered throughout this paper: the reliability mechanisms on the LMS channel to deal with the high error rate, and the transport protocol.

A. Satellite Environment

We have chosen Network Simulator 2 (hereafter referred to as *ns* 2) to simulate the satellite environment, which is composed of 66 LEO satellites, at an altitude of 800 km, ensuring a global coverage of any point on Earth at any time. Due to the movement of the satellite and route changes, the transmission delay varies from 70 to 90 ms within the satellite constellation. Moreover, except on the LMS forward link, we consider a path error free, as shown in Fig. 1, between the sender and the last satellite. Finally, messages are sent from the server to the mobile receiver through a satellite constellation representing the existing satellite constellations already deployed.

B. Hybrid Automatic Repeat Request

We previously explained that HARQ schemes are used to mitigate link layer impairments on the LMS channel. These schemes aim to compensate the high error rate characterizing such a channel by benefiting from both automatic repeat request and forward error correction (FEC) coding, as well as optimizing the usage of the LMS channel. There are two main types of HARQs:

1) Type I HARQ, which is also named Chased Combining, occurs when a retransmission is needed and the sending node again sends the same message as in the first transmission, which is composed of the message and redundancy bits.

2) Type II HARQ, which is also named Incremental Redundancy, occurs when the information bits are only sent at the first sending, with some redundancy bits. When a retransmission is asked, the sender sends more redundancy bits that are different from the bits sent in previous transmissions. Thus, each reception of additional data will add more redundancy bits to help decode the message.

We present in this paragraph the basic principle of the version used in our simulations, which is adaptive HARQ [12]. This version is an improvement of type II HARQ, and it uses the mutual information to compute at each reception if a packet can be decoded and, if not, how many additional redundancy bit are needed. This optimizes the use of the LMS channel capacity by sending only the desired number of bits. This version allows up to three retransmissions in case of erased packets. The principle detailed in Fig. 2a is as follows: the receiver side of the HARQ link stores the bits received while a packet has not been decoded. Each time useful or redundancy bits are received, the module computes whether enough data have been received to decode the packet. If not, a Negative ACK (NACK) is sent to the HARQ sender, asking for more redundancy bits. Transmissions and decoding times take between 10 ms (no retransmission of redundancy bits) and 70 ms (three retransmissions), depending on the channel quality. If the packet cannot be decoded after three retransmissions, it is dropped.

As mentioned in Sec. I, solutions have been proposed to improve the performance of the communication by mitigating the jitter caused by the HARQ. Due to the varying decoding times by the HARQ, packets are delivered out of order by this module to the upper layers. The TCP stores these out of order packets, but this generates a lot of retransmissions that lower TCP performance and should be avoided. The reordering mechanism proposed in Ref. [7] allows the packets to be reordered at the link layer and then delivered in order to the upper layers. This solution shows better results in terms of TCP performance. Thus, upper layers (particularly the transport layer) receive ordered packets that lead to better performance. The cost of this mechanism is an additional delay that needs to be below the TCP timeout value. An overview of this mechanism, including both the HARQ and the link layer reordering mechanism, is given in Fig. 2b.

Concerning the buffer sizes, the buffers used to store packets being decoded by the HARQ (waiting for additional redundancy bits) and to reorder them have been designed to always handle all the packets being processed. Thus, we never have overflow in these buffers.

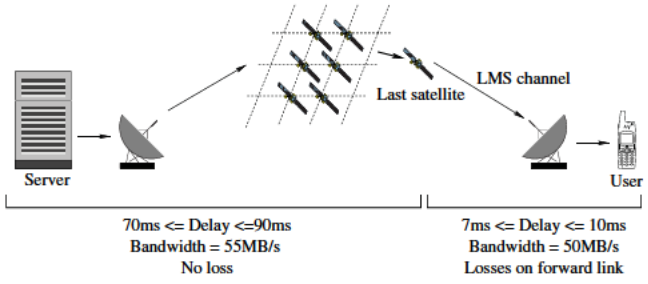
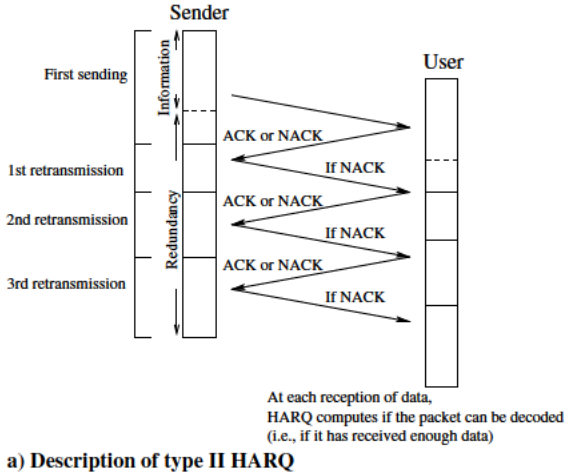
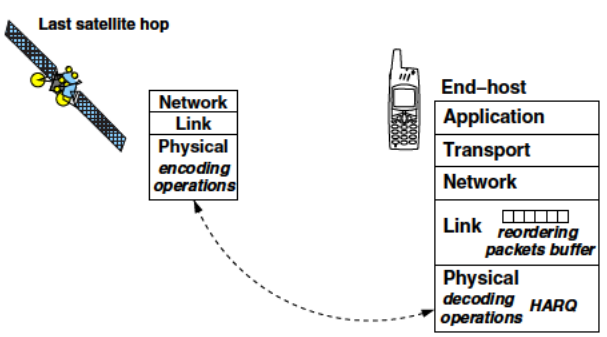


Fig. 1 Model for a satellite constellation (BW, Bandwidth; NACK, Negative Acknowledgment).



a) Description of type II HARQ



b) Description of HARQ with the reordering mechanism

Fig. 2 Description of the low layer mechanisms.

Finally, we reused the LMS channel details used in Ref. [12] to characterize the LMS channel: 1) modulation Quadrature phase shift keying; 2) mother FEC code, with Consultative Committee for Space Data Systems turbocodes of 1/6; and 3) a codeword length of 53,520 bits (data bits = 8920 bits).

In the following, we study the impact of DelAcks with and without this link layer reordering mechanism, and we assess whether enabling DelAcks is mandatory in both cases.

C. Version of TCP and Parameters

We experiment with both TCP NewReno and CUBIC in our simulations, which are presented in section V of [13] as recommended TCP variants. The TCP is, today, the major Internet transport protocol, and it has been shown to provide reasonable performance over LEO constellations [11,14] without requiring specific PEP optimization mechanisms. TCP NewReno features basic reliability functionalities of most TCP variants and is used here as a reference protocol in our simulations. We also consider CUBIC [15] because of the following:

1) It is enabled by default in GNU/Linux and macOS systems (since 10.9).

2) It performs better than TCP NewReno over long delay links [6].

The CUBIC congestion window is growing faster than the TCP NewReno window in order to reach the optimal value faster. It has been designed to improve the scalability of the TCP over fast and long distance networks.

DelAcks can combine two in order packets if, and only if, they are received within a fixed time window [16]. A timer is started when a first packet is received; if another packet is received before the timer expires, a single cumulative acknowledgment is sent for both packets. If the timer expires, the TCP acknowledges only the packet received, and it resets the timer. If the TCP receives an out of order packet, an ACK is instantly sent for this packet, allowing the TCP to adapt its congestion window.

D. Simulation Scenario

To ease the simulation and make it faster, we performed simulations with only three nodes to represent the sender, the last satellite on the message route, and the receiver. The satellite constellation was simulated by varying the delays between the nodes. We used `SaVi` [17] to get the parameters of the previously described LEO satellite constellation; then, we played a first simulation to get the evolution of the delays between the sender and the last satellite, as well as between the last satellite and the ground receiver. Then all *ns* 2 simulations were played using the three nodes and the temporal traces generated.

The simulations are run with a LMS channel [18,19] between the last satellite and the ground gateway in an intermediate tree shadowed environment. The HARQ is always activated in our simulations. We also vary the quality of this channel by setting a reference signal to noise ratio (SNR) ranging from 7 to 13 dB. During the simulations, the link quality changes over time around this SNR reference value. Each simulation lasts 600 s, with one single TCP flow performing.

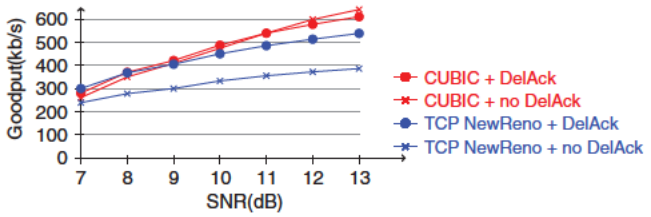


Fig. 3 Impact of DelAck on end-to-end goodput.

III. Study of the Impact of DelAck

To analyze the impact of DelAck, we report values of the TCP retransmission timeout (RTO) and TCP duplicate acknowledgment (DUPACK), which are essential metrics used by the TCP to recover lost packets. Basically, RTO is a timeout to trigger retransmissions when no acknowledgments have been received after a given period, whereas DUPACK detects a loss in the middle of a flow transmission. These mechanisms allow the receiver to inform the sender that a packet is missing and trigger retransmission of this packet as soon as possible. These congestion events lower the sending rate of the TCP.

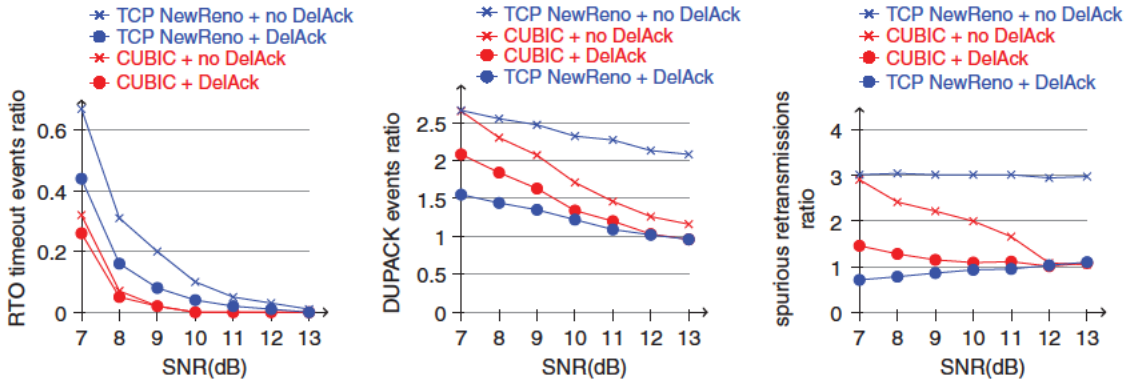
In this section and the following, the values of the RTO timeout events, DUPACK, and spurious retransmission (unnecessary retransmission of TCP segments that are delayed and not lost) have been normalized by the number of packets sent in order to have comparable results in the different graphs and tables.

We observe, as shown in Figs. 3 and 4, that we have a low goodput, whatever the value of the SNR; whereas we could expect a goodput of 40 Mb/s in the case of an error free network. On the other hand, the values of the DUPACK and spurious retransmissions remain high, and they do not decrease when the channel quality improves, as we could expect. This is mainly due to out of order packets arriving too late and, as a consequence, being interpreted by the TCP as a congestion event, which results in spurious retransmissions and congestion window halving.

Some packets cannot be decoded after three retransmissions by the HARQ and are dropped. This proportion of packets ranges from 4% with SNR = 7 to less than 0.5% with SNR = 13. As we could expect, the proportion of packets not recovered by the HARQ decreases when the channel quality improves.

A. Impact of DelAck Without Reordering Mechanism

The default value of DelAck on GNU/Linux systems depends on the system architecture, but it is often around 40 ms. This is also the default value in *ns* 2 that we kept in our simulations. As shown in Figs. 3 and 4, we observe that DelAck significantly improves TCP performance when using TCP NewReno; the goodput has increased; and the number of RTO timeout events, DUPACK, and spurious retransmissions has decreased. However, DelAck has no impact on



a) Impact on RTO timeout events b) Impact on DUPACK events c) Impact on spurious retransmissions

Fig. 4 Impact of DelAck on TCP performance.

CUBIC goodput, even if we observe a diminution of the number of DUPACK and spurious retransmissions. This shows that there is another mechanism, linked to DelAck and the topology studied in this paper, which counteracts the diminution of DUPACK, RTO timeout events, and spurious retransmissions, especially when using CUBIC. An explanation for this bad performance is provided in Sec. IV.

B. Impact of DelAck with the Link-Layer Reordering Mechanism

As seen in Fig. 5, when the link layer reordering mechanism is enabled, activating DelAck greatly decreases TCP performance, more particularly when using CUBIC, where the goodput fall is significant for high values of the SNR. This behavior can be observed for both TCP NewReno and CUBIC.

This points out the need to understand why the activation of DelAck decreases TCP goodput in this scenario; explications are given on Sec. IV.

C. Impact with Other TCP Protocols

Although we only presented detailed results with TCP NewReno and CUBIC, we also experimented without reordering mechanisms using other TCP protocols such as TCP Hybla, which is a TCP version designed for geostationary orbit satellite links [20]. Its performance on long delay links makes TCP Hybla a good candidate to study, and it can bring additional results in our scenario. We observe a strong decrease of the goodput when DelAck is activated. The fall can be up to 40% with this protocol, leading to the conclusion that DelAcks should be disabled when using TCP Hybla over a LEO satellite constellation.

However, despite this performance drop when enabling DelAck, TCP Hybla still performs better goodput than TCP NewReno or CUBIC. With DelAck, TCP Hybla goodput is always higher than 850 kb/s, whereas TCP NewReno and CUBIC cannot achieve goodputs higher than 650 kb/s. Without DelAck, the goodput with TCP Hybla ranges from 1141 kb/s (SNR = 7) to 1764 kb/s (SNR = 13).

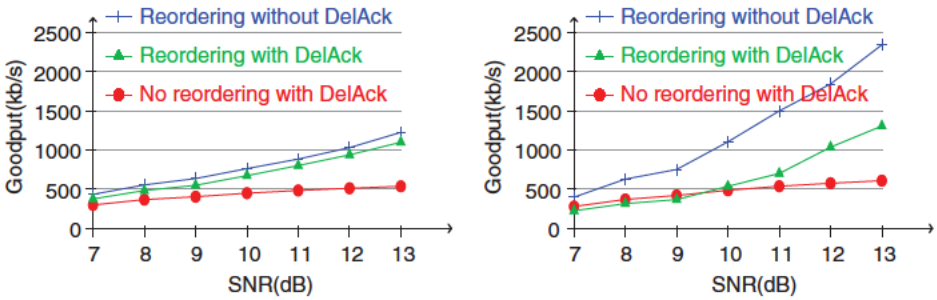
IV. Analysis

The TCP congestion window is generally updated when an acknowledgment is received, during the slow start or fast recovery phase; the use of DelAck slows down this growth because of the reduced number of acknowledgments, as observed in Ref. [11]. The evolution of the congestion window with and without DelAck is shown in Fig. 6. Each curve is generated independently with one single TCP flow between two nodes, without any loss, and without the HARQ. This is just a simple case illustrating the impact of DelAck on the congestion window evolution. Such pacing might have benefits for congested networks by delaying the filling of the buffer at the bottleneck and then allowing the flow to have a larger congestion window [21]. However, in our scenario, we are placed in a mostly uncongested network, meaning that losses are mainly due to transmission errors. As a result, pacing only delays transmissions and loss detection signals [22], resulting in a decrease of the goodput.

In our context of LEO satellite communications, in which delays are varying due to satellite movements, route changes, and reliability mechanisms such as the HARQ, any TCP variant triggers more retransmissions than on terrestrial networks. Moreover, the TCP needs to recover packets lost due to errors on the LMS channel, in addition to those lost due to congestion. All the solutions previously listed cannot totally mitigate the effects brought by the use of satellite constellations, and we still observe more slow start and fast recovery phases than on terrestrial networks due to these random losses [23].

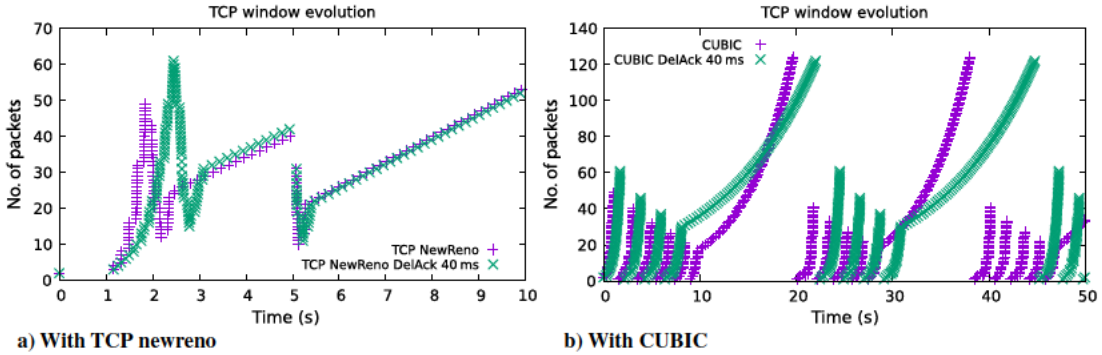
We observe in Table 1 that CUBIC triggers more retransmissions than TCP NewReno. CUBIC is more often in the slow start or fast recovery phase, whereas DelAck delays the progressions of the congestion window. Thus, the gain of performance for the TCP when DelAck is activated is lower with CUBIC than with TCP NewReno. This trend is also observed with TCP Hybla, in which the number of retransmissions is higher than with TCP NewReno or CUBIC due to the larger congestion window [20]. Thus, in our scenario, when a transport layer protocol is aggressive (e.g., TCP Hybla or CUBIC), enabling DelAck results in slower progression of the congestion window and a reduced goodput; whereas enabling DelAck for transport protocols with lower retransmission levels (e.g., TCP NewReno) improves the goodput of the connection by taking advantage of DelAck.

Concerning TCP performance when the link layer reordering mechanism was present, activating DelAck increased the



a) With TCP NewReno b) With CUBIC

Fig. 5 Impact of DelAck on TCP performance with a link-layer reordering mechanism.



a) With TCP newreno b) With CUBIC

Fig. 6 Evolution of TCP congestion window.

Table 1 Number of packets retransmitted during all the simulation, when DelAck is activated

	Number of retransmissions		
SNR, dB	TCP NewReno	CUBIC	TCP Hybla
7	1,068	1,333	6,969
8	932	1,100	16,914
9	925	1,105	23,073
10	789	949	16,782
11	748	923	22,399
12	742	780	23,852
13	706	789	20,688

transmission delay, which also increased the probability of TCP timeout. We measured that the waiting time between two packets forming a DelAck was higher than 10 ms in 10% of the cases. This implied that the acknowledgment of the first packet of DelAck was delayed by the same value. This waiting time was significant due to the reordering mechanism, and we observed that this additional delay was often proportional to the time needed to get an additional HARQ transmission to decode a packet.

Furthermore, we observe a decrease of the value of RTO for both TCP NewReno and CUBIC when DelAck is enabled, giving less time to the packets to reach their destination, and increasing the probability to trigger retransmissions due to timeout. We recall that the RTO timer is regularly updated during the transmission, and it is the sum of the Smoothed Round Trip Time (SRTT) and four times the round trip time variation (RTTVAR) [24]. The SRTT is the exponential moving average of the round trip time (RTT), and the RTTVAR is the exponential moving average of the absolute value of the difference between the last RTT measured and the SRTT. This algorithm is made to adapt the RTO to the RTT changes. Thus, if the RTT has a high variation, the computed RTO is higher to handle packets with a longer transmission delay and avoid spurious retransmissions. On the other hand, if there is no RTT variation, the RTO is smaller to detect, as soon as possible, a packet loss. We observe in the simulations that the RTT variation is lower when DelAck is activated. Thus, both the RTTVAR and RTO values are lower.

This lower RTO value and the additional delay caused by DelAck, added to the constellation transmission delay, the HARQ, and the reordering mechanism, cause a higher number of retransmissions by the RTO, as presented in Fig. 7. This high number of timeouts badly impacts the performance of the TCP, decreasing its performance: it lowers the congestion window and switches to the slow start mode, in which the growing of the congestion window is slower. The goodput is finally highly impacted by the changes brought by DelAck. However, thanks to the reordering mechanism, we have a lower proportion of DUPACK, and TCP performance remains higher than without the reordering mechanism, even if DelAck has a counterproductive impact.

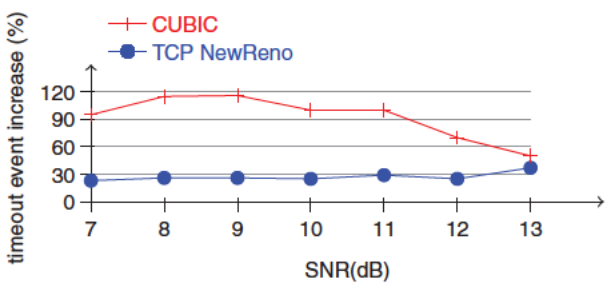


Fig. 7 Impact of the activation of DelAck on the retransmission ratio due to timeout when the reordering mechanism is present.

V. Conclusions

This paper assesses the impact of activating delayed acknowledg ment (DelAck) on the behavior of different Transmission Control Protocol (TCP) variants over low Earth orbit satellite constellations in which link layer reliability schemes are considered. It is shown that adding DelAck can have different impacts on TCP performance, depending on the variant of the TCP used. It is recommended to activate it when using TCP NewReno. However, with CUBIC, it can decrease TCP performance in some cases, and the DelAck activation is not mandatory. More generally, TCP variants that are more aggressive and retransmit often, such as TCP Hybla, are negatively impacted by DelAck, which should be disabled to improve performance. There is also a need to carefully take into account the presence of other link layer mechanisms aiming to improve TCP performance with the HARQ; the use of DelAck conjointly with them can be counterproductive by adding a delay to a network in which the transmission delays are already high.

References

[1] Border, J., Kojo, M., Griner, J., Montenegro, G., and Shelby, Z., "Performance Enhancing Proxies Intended to Mitigate Link Related Degradations," RFC Editor, Request for Comments RFC 3135, Informational, Fremont, CA, June 2001, http://www.ietf.org/rfc/rfc3135.txt [retrieved 16 Feb. 2017].

[2] Lee, D., Carpenter, B. E., and Brownlee, N., "Observations of UDP to TCP Ratio and Port Numbers," *2010 Fifth International Conference on Internet Monitoring and Protection*, Inst. of Electrical and Electronics Engineers, 2010, pp. 99 104.

[3] Blaunstein, N., Cohen, Y., and Hayakawa, M., "Prediction of Fading Phenomena in Land Satellite Communication Links," *Radio Science*, Vol. 45, No. 6, 2010, pp. 1 13. doi:10.1029/2010RS004352

[4] Sastry, A., "Performance of Hybrid Error Control Schemes of Satellite Channels," *IEEE Transactions on Communications*, Vol. 23, No. 7, 1975, pp. 689 694.

[5] De Cola, T., Ernst, H., and Marchese, M., *Application of Long Erasure Codes and ARQ Schemes for Achieving High Data Transfer Performance Over Long Delay Networks*, Springer, Boston, 2008, pp. 643 656.

[6] Kuhn, N., Lochin, E., Lacan, J., Boreli, R., and Clarac, L., "On the Impact of Link Layer Retransmission Schemes on TCP over 4G Satellite Links," *International Journal of Satellite Communications and Networking*, Vol. 33, No. 1, 2015, pp. 19 42. doi:10.1002/sat.v33.1

[7] Tauran, B., Lochin, E., Lacan, J., Arnal, F., Gineste, M., Clarac, L., and Kuhn, N., "Making H ARQ Suitable for a Mobile TCP Receiver over LEO Satellite Constellations," *Wireless and Satellite Systems*, Springer, Cham, 2017, pp. 33 42.

[8] Braden, R., "Requirements for Internet Hosts Communication Layers," RFC Editor, Request for Comments RFC 1122, Internet Standard, Fremont, CA, Oct. 1989, http://www.ietf.org/rfc/rfc1122.txt [retrieved 16 Feb. 2017].

[9] Judd, G., "Attaining the Promise and Avoiding the Pitfalls of TCP in the Datacenter," *Networked Systems Design and Implementation*, 2015, pp. 145 157.

[10] Chen, J., Gerla, M., Lee, Y. Z., and Sanadidi, M., "TCP with Delayed Ack for Wireless Networks," *Ad Hoc Networks*, Vol. 6, No. 7, 2008, pp. 1098 1116. doi:10.1016/j.adhoc.2007.10.004

[11] Wood, L., Pavlou, G., and Evans, B., "Effects on TCP of Routing Strategies in Satellite Constellations," *IEEE Communications Magazine*, Vol. 39, No. 3, 2001, pp. 172 181.

[12] Ahmad, R. A., Lacan, J., Arnal, F., Gineste, M., and Clarac, L., "Enhanced HARQ for Delay Tolerant Services in Mobile Satellite Communications," *Seventh International Conference on Advances in Satellite and Space Communications 2015*, International Academy, Research and Industry Association, Wilmington, DE, 2015, pp. 1 6.

[13] Kuhn, N., Natarajan, P., Khademi, N., and Ros, D., "Characterization Guidelines for Active Queue Management (AQM)," RFC Editor, Request for Comments RFC 7928, Informational, Fremont, CA, July 2016, http://www.ietf.org/rfc/rfc7928.txt [retrieved 16 Feb. 2017].

[14] Chotikapong, Y., Cruickshank, H., and Sun, Z., "Evaluation of TCP and Internet Traffic via Low Earth Orbit Satellites," *IEEE Personal Communications*, Vol. 8, No. 3, 2001, pp. 28 34.

[15] Ha, S., Rhee, I., and Xu, L., "CUBIC: A New TCP Friendly High Speed TCP Variant," *SIGOPS Operating Systems Review*, Vol. 42, No. 5, 2008, pp. 64 74.
doi:10.1145/1400097

[16] Allman, M., Paxson, V., and Blanton, E., "TCP Congestion Control," RFC Editor, Request for Comments 5681, Draft Standard, Fremont, CA, Sept. 2009, http://www.ietf.org/rfc/rfc5681.txt [retrieved 16 Feb. 2017].

[17] Wood, L., "SaVi: Satellite Constellation Visualization," *First Annual Centre for Communication Systems Research Research Symposium*, arXiv:1204.3265, 2011.

[18] Perez Fontan, F., Vazquez Castro, M. A., Buonomo, S., Poiares Baptista, J. P., and Arbesser Rastburg, B., "S Band LMS Propagation Channel Behaviour for Different Environments, Degrees of Shadowing and Elevation Angles," *IEEE Transactions on Broadcasting*, Vol. 44, No. 1, 1998, pp. 40 76.

[19] Fontan, F. P., Vazquez Castro, M., Cabado, C. E., Garcia, J. P., and Kubista, E., "Statistical Modeling of the LMS Channel," *IEEE Transactions on Vehicular Technology*, Vol. 50, No. 6, 2001, pp. 1549 1567.

[20] Caini, C., and Firrincieli, R., "TCP Hybla: A TCP Enhancement for Heterogeneous Networks," *International Journal of Satellite Communi cations and Networking*, Vol. 22, No. 5, 2004, pp. 547 566.
doi:10.1002/(ISSN)1542 0981

[21] Aggarwal, A., Savage, S., and Anderson, T., "Understanding the Performance of TCP Pacing," *Conference on Computer Communications. Nineteenth Annual Joint Conference of the Institute of Electrical and Electronics Engineers Computer and Communications Societies*, Vol. 3, IEEE Publ., Piscataway, NJ, 2000, pp. 1157 1165.

[22] Sallantin, R., Baudoin, C., Chaput, E., Arnal, F., Dubois, E., and Beylot, A. L., "Initial Spreading: A Fast Start up TCP Mechanism," *2013 Institute of Electrical and Electronics Engineers 38th Conference on Local Computer Networks (LCN)*, IEEE Publ., Piscataway, NJ, 2013, pp. 492 499.

[23] Altman, E., Avrachenkov, K., and Barakat, C., "TCP in Presence of Bursty Losses," *Performance Evaluation*, Vol. 42, No. 2, 2000, pp. 129 147.
doi:10.1016/S0166 5316(00)00027 4

[24] Paxson, V., Allman, M., Chu, J., and Sargent, M., "Computing TCP's Retransmission Timer," RFC Editor, Request for Comments RFC 6298, Proposed Standard, Fremont, CA, June 2011, http://www.ietf.org/rfc/rfc6298.txt [retrieved 16 Feb. 2017].

R. Vingerhoeds
Associate Editor