Hindawi
Security and Communication Networks
Volume 2020, Article ID 6693094, 7 pages
https://doi.org/10.1155/2020/6693094

Research Article

On Enhancing TCP to Deal with High Latency and Transmission Errors in Geostationary Satellite Network for 5G-IoT

Liang Zong,[1] Yong Bai,[2] Chenglin Zhao,[1] Gaofeng Luo,[1] Zeyu Zhang,[3] and Huawei Ma[4,5]

[1]*College of Information Engineering, Shaoyang University, Shaoyang 422000, Hunan, China*
[2]*School of Information and Communication Engineering, Hainan University, Haikou 570228, Hainan, China*
[3]*School of Applied Sciences, Macao Polytechnic Institute, Macau 999078, China*
[4]*Institute of Data Science, City University of Macau, Macau 999078, China*
[5]*School of Information Technology, Beijing Institute of Technology, Beijing 519088, China*

Correspondence should be addressed to Huawei Ma; mkmhw@bitzh.edu.cn

Received 12 October 2020; Revised 6 November 2020; Accepted 26 November 2020; Published 9 December 2020

Academic Editor: Lingwei Xu

The geostationary (GEO) satellite networks have two important influencing factors: high latency and transmission errors. Similarly, they will happen in the large-scale multihop network of the Internet of things (IoT), which will affect the application of 5G- (5th-generation mobile networks-) IoT. In this paper, we propose an enhanced TCP mechanism that increases the amount of data transferred in the slow start phase of TCP Hybla to mitigate the effect of long RTT and incorporates a refined mechanism of TCP Veno, which can distinguish packet loss between random and congestion. This scheme is evaluated and compared with NewReno, Hybla, and Veno by simulation, and the performance improvement of the proposed TCP scheme for GEO satellite network in the presence of random packet losses is demonstrated. At the same time, the enhanced TCP scheme can improve the transmission performance in the future 5G-IoT heterogeneous network with high delay and transmission .

1. Introduction

With the advent of the 5G era and the wide application of the IoT, satellite networks will play an increasingly important role as a cross-regional relay network [1]. The dominant transport layer protocol of Internet Transmission Control Protocol (TCP) provides reliable delivery of data streams with its intertwined flow control, congestion control, and error control functions. The original design of TCP protocol is used for terrestrial wired networks, and it has been proved to be very successful for supporting Internet data communications for decades. However, there are several factors that decrease the performance of the TCP over satellite networks [2–5]. Firstly, due to the ultralong distance of GEO satellite link, there will inevitably appear high delay phenomenon in data transmission. The high latency imposed by the long RTT (round-trip time) will delay the execution of TCP and affect its performance. For satellite links with long RTTs, the bandwidth delay product (BDP) can be quite high, and TCP needs to keep the amount of packets "in flight" (i.e., sent but not yet acknowledged), which means that the TCP must be able to handle larger data in a single transfer window. Secondly, transmission errors can exist in satellite links due to some harmful effects (e.g., atmospheric conditions, jamming, and interference), and the resulting bit-error rates (BER) can be as low as 10^{-7} on average and 10^{-4} in the worst case, which are higher than typical terrestrial networks. In contrast with the packet loss caused by network congestion (due to router buffer overflow), the packet loss caused by transmission error is referred to as corruption loss, random loss, or error loss in the literature. Explicit congestion notification (ECN) [6] is a proposed addition to IP which allows routers to inform TCP senders about imminent congestion before the queue overflows. Other factors that may affect TCP performance include bandwidth asymmetry, variable RTTs, and intermittent connectivity.

With the above negative characteristics of satellite networks, improving TCP throughput over satellite links is a hot area of research [5]. The enhancements for TCP over satellite networks can be classified into three categories: lower-level approaches, end-to-end approaches, and performance-enhancing proxies (PEP). Two lower-level approaches are path maximum transmission unit (MTU) discovery and forward error correction (FEC). Path MTU discovery is conducted at the data link layer. In the end-to-end approach, only the sender and receiver are involved in flow control and congestion control. The MUT and FEC are employed in the TCP variants (Tahoe, Reno, and NewReno) [7, 8]. Several proposals show the enhanced performance to make the slow start less time-consuming, including using a large initial value of *cwnd* [9, 10] and turning off the mechanism of delayed ACKs. Another TCP enhancement is employing selective acknowledgments (SACKs) [11].

Since the geostationary (GEO) satellite networks have two important influencing factors, high latency and transmission errors, it is essential to focus on the satellite network with long RTT and high BER. In recent years, some algorithms have been proposed to solve the congestion control problem in satellite networks, such as TCP Hybla [12, 13], Vegas [14], Veno [15], and Westwood [16]. Aided by the normalized RTT, the update rules of *cwnd* in the actual congestion control algorithms are replaced to achieve the same transmission rate as the reference connection even over longer RTT connections.

The PEP approaches require some changes in the network, mainly on intermediate gateways. TCP connections may be split in the total network and different transport protocols can be employed on the split connections. TCP-Spoofing and TCP-Splitting are two such PEP solutions [17, 18]. In addition, Dubois et al. proposed some architecture elements for PEPs mechanisms considering mobility scenarios for satellite networks [19]. Pirovano and Garcia summarized and analyzed the common solutions of TCP over satellite and evaluated the scheme of splitting TCP connections [20]. Although promising, the PEP approaches intrinsically infringe the end-to-end semantics of TCP, which presents several disadvantages on deployment, security, and privacy issues.

In this paper, we present a scheme for the GEO satellite network that can distinguish packet loss between random corruption and congestion loss. Our proposal combines the benefits of Hybla (for dealing with long RTT) and Veno (for dealing with random errors). In the scheme, the new rules of *cwnd* of TCP Hybla are still used in the slow start and congestion avoidance phases to mitigate the effect of long RTT.

The satellite communication network has been an important part of the 5G-IoTs and Internet infrastructure. Whether it is the satellite-based NB-IOT (Narrow Band Internet of Things) network or the ground-based 5G-IoT network, both realize the interconnection of everything and expand the scope of 5G-IoT connection by expanding the connection mode of the Internet of things. This enables 5G-IoT communication technology to cover all scenarios, especially in cross-regional situations. Therefore, improving the transmission performance of the satellite network is essential to improve the cross-regional 5G-IoT performance.

Section 2 of the paper gives an overview of related TCP variants. In Section 3, our proposed enhanced TCP mechanisms are presented. The performance evaluation of our proposal by simulation is conducted and discussed in Section 4. Finally, conclusions are drawn in Section 5.

2. Overview of Related TCP Variants

2.1. TCP Tahoe and NewReno

2.1.1. TCP Tahoe. The implementation of TCP Tahoe [7] for congestion control is divided into three steps: slow start, congestion avoidance, and loss recovery algorithm. The sender performs Additive Increase of the congestion window (*cwnd*) in the initial phase of TCP connection. In the first slow start (SS) phase, *cwnd* increases from one or two initial values of the maximum segment size (MSS).

The increase of *cwnd* is exponential in the SS phase, and TCP slows down the increase of *cwnd* in the CA phase. TCP assumes that congestion exists when it detects a packet loss. The loss of packets can be detected via the timeout of the retransmission timeout (RTO) timer. Furthermore, Tahoe incorporates a so-called fast retransmit mechanism, in which the sender infers that a packet has been lost if three or more duplicate ACKs (DU-PACKs) are received successively. When the event of packet loss has been decided, the sender performs Multiplicative Decrease of *cwnd*: *cwnd* is reset to one, and ssthresh is set to half the current size of the congestion window. Then, TCP enters the SS phase. In summary, the update rules of *cwnd* and ssthresh of Tahoe are given as follows:

```
(1) New Ack received:
    if ($cwnd < ssthresh$)//SS phase
        set $cwnd = cwnd + 1$;
    else//CA phase
        set $cwnd = cwnd + 1/cwnd$;
(2) Packet loss detected (RTO timeout or three
    DUPACKs received):
    set $ssthresh = cwnd/2$; $cwnd = 1$;
    enter SS phase.
```

2.1.2. TCP NewReno. In TCP Reno [7], DUPACKs indicate that the data is still transmitting. If three DUPACKs are received, Reno is half the *cwnd* (instead of setting it to 1 MSS like Tahoe), set *ssthresh* to the new *cwnd*, and enter a phase called fast recovery. It avoids slow transmission of retransmitted data. In the fast recovery phase, TCP Reno retransmits the lost packet and waits for the information of the ACK before entering the congestion avoidance. In summary, the update rules of *cwnd* and *ssthresh* during Reno's fast recovery after three DUPACKs received are given as follows:

```
After three DUPACKs were received:
set $ssthresh = cwnd/2$;
```

```
        $cwnd = ssthresh + 3$;
    enter CA phase;
    $cwnd = cwnd + 1$ for each additional DUPACK.
```

If multiple packets are lost, the sender will receive the data acknowledgement from the receiver for retransmission (the fast retransmission algorithm has been entered at this time). In partial acknowledgement, TCP will cause the sender to exit and recover quickly. At this time, the sender must wait for the timeout. This directly leads to transmission performance degradation. TCP NewReno [8] handles this situation well and avoids going to the slow start.

2.2. TCP Hybla.

2.2. TCP Hybla. TCP Hybla [12, 13] aims to eliminate the impact of long RTT caused by high delay ground line or satellite wireless link. It is modified by dynamic analysis of congestion window to reduce the performance dependence on RTT. Firstly, the normalized round-trip time ρ is defined and the round-trip time of the reference connection is RTT_0.

$$\rho = \frac{\mathrm{RTT}}{\mathrm{RTT}_0}. \tag{1}$$

The normalized round-trip time is helpful to maintain the high performance of TCP in the case of long delay. TCP Hybla's update window is as follows:

$$W_{i+1} = \begin{cases} W_i + 2^{\rho} - 1, & \text{SS phase}, \\ W_i + \dfrac{\rho^2}{W_i}, & \text{CA phase}, \end{cases} \tag{2}$$

where W_i is the update of the congestion window when the sender of TCP connection receives the *i*th ack. TCP Hybla also adopts some measures to improve transmission performance, such as forcing SACK strategy, using timestamp [21], using hoes channel bandwidth estimation [22, 23], and implementing packet spacing technology [24]. In theoretical simulation and practical tests, TCP Hybla has been greatly improved compared with the traditional TCP.

2.3. TCP Veno. TCP Veno [15] is an improved scheme of traditional TCP for solving the problem of high bit error rate in wireless network environment. It uses a mechanism similar to Vegas [14]. TCP Veno judges the state of the network by calculating the backlog of datagrams in the network. If the size of the backlog datagram is N, then

$$N = \mathrm{Actual} \times (\mathrm{RTT} - \mathrm{BaseRTT}). \tag{3}$$

In practical application, actual = $cwnd$/RTT, expected = $cwnd$/BaseRTT, and $cwnd$ is the size of the congestion window. Thus, N is also as follows:

$$N = (\mathrm{Expected} - \mathrm{Actual}) \times \mathrm{BaseRTT}. \tag{4}$$

TCP Veno takes the size of N as the basis to judge whether the network is congested or not and sets a threshold value β. When N exceeds β, it indicates that the data packets in the link are seriously overstocked, and the connection is in the congestion state. If packet loss occurs at this time, the control mechanism similar to Reno is adopted; when N is less than β, TCP Veno sets a threshold value of β even if the sender detects the packet loss, the connection is normal, the packet loss is determined to be caused by other reasons, and a congestion control algorithm different from Reno is adopted.

TCP Veno also uses two stages to achieve high performance for congestion control. In the congestion avoidance phase, when the packet loss is not caused by congestion, *cwnd* is increased by 1 for each new acknowledgement. When the packet loss is detected to be caused by congestion, *cwnd* is increased by 1 for every two new acknowledgments. In fast retransmission and fast recovery, when receiving 3 or more repeated acknowledgments, if $N < \beta$, it is considered that the network is not congested enough, the packet loss is determined as immediate packet loss, the congestion threshold is set to $4^* \ cwnd/5$, cwnd = ssthresh + 3, and the lost packets are retransmitted; if $N > \beta$, the network is considered to be congested, the congestion threshold is set to $cwnd/2$, and $cwnd = ssthresh + 3$ is set to retransmit the lost packets.

3. Proposal of Enhanced TCP Mechanisms

According to the TCP performance test of Hybla and Veno, the former can deal with the low efficiency caused by long delay, while the latter can deal with the problem of high bit error rate in wireless communication. To make full use of their advantages on processing TCP scheme, this paper proposes an enhanced TCP scheme to adapt to the problems of long RTT and high bit error rate in satellite networks.

3.1. Congestion Window in the Slow Start. In the TCP Hybla, we first modify the value of RTT_0. The original RTT_0 value is set to 25 ms. Considering that the value of round-trip delay in the satellite network is relatively large, generally the delay is 550 ms in GEO satellite network. For the congestion window in TCP Hybla, this will make the value of congestion window in the slow start phase become too large, which will affect its performance. Therefore, the original value of RTT_0 can be appropriately increased. The references considered set the value of RTT_0 to 70 ms [25, 26], and the value is set to 75 ms in the enhanced scheme. This mainly considers that the transmission control protocol will bring certain delays when dealing with real congestion. After the update RTT_0 is reset, the original update scheme is still maintained for the congestion window update.

Under the above congestion window update rules, the enhanced scheme refines the window update. In the traditional TCP, the size of the slow start congestion window needs to be compared with the threshold value. In the proposed scheme, the different congestion windows are set according to the comparison of congestion window and threshold as follows:

```
if (cwnd <ssthresh)//SS phase
{
```

```
if ($cwnd \le ssthresh/4$) && ($flightsize < rwnd/4$)//sub-phase 1
    set $cwnd = 2 * W_i + 1$
if ($ssthresh/4 < cwnd < 3 * ssthresh/4$) && ($rwnd/4$ $flightsize < 3 * rwnd/4$)//su-phase 2
    set $cwnd = W + 1$
if ($cwnd \le 3 * ssthresh/4$)//($flightsize > 3 * rwnd/4$)// sub-phase 3
    set $cwnd = W + 1/2$
}
```

*The above t*hree subphases are divided according to four factors: *cwnd*, *ssthresh, rwnd* (receiver advertisement window), and flightsize (the total number of unacknowledged bytes in the network). From the first subphase, it can be seen that the value of congestion window is a relatively small value, while the values of *rwnd* and flightsize indicate that the amount of data transmitted in the network connection is at a low value. At this time, the amount of data transmission can be increased and the utilization efficiency of the link can be improved.

In subphase 2, with the update of congestion window, its value will gradually increase. When the congestion window is in the middle of the slow start threshold, the state of the receiving data in the long delay link cannot be determined. Therefore, the value of flightsize is added to judge the state of the received data. If the value of flightsize is in the middle of *rwnd*, it indicates that the network state may be in a relatively stable state. The size of the congestion window keeps increasing to the TCP Hybla settings.

In the last subphase, the value of congestion window is close to the threshold size of the slow start, or the value of the flightsize is close to the value of *rwnd*. The enhanced scheme judges that there may be a lot of backlogs in the network. To avoid data loss caused by data backlog, the added value of congestion window can be appropriately reduced.

In the slow start phase of the proposed scheme, we fully consider the influence of different variables (*cwnd*, flightsize, *ssthresh*, and *rwnd*) on the congestion window. The value of congestion window is refined to better adapt to the different states.

3.2. Congestion Window in the Fast Recovery.

3.2. Congestion Window in the Fast Recovery. Traditional TCP considers the data loss caused by timeout as congestion in the network, which is correct for the network with the stable wired links. For the satellite network of the wireless links, we must distinguish the real cause of data loss: the loss caused by the congestion or the random loss caused by the wireless link. The difference of data loss in TCP Veno is not suitable for the satellite network.

In the enhanced TCP scheme, the congestion window is adopted as follows:

```
if ($N < \beta_1$)//random loss is most likely to have occurred
    set $ssthresh = 4 * cwnd/5$;
else if ($\beta_1 \le N < \beta_2$)//not certain which kind of loss occurred
    set $ssthresh = 3 * cwnd/5$;
else set $ssthresh = 2 * cwnd/5$;//congestion loss is most likely to have occurred
```

In the above scheme, β_1 and β_2 are set to 3 and 6, respectively. Considering that the data loss of satellite network is still caused by congestion loss and random loss, the enhanced TCP scheme refines the value of congestion window, and the size of N is an important basis for network congestion judgment. Different values of N represent the network congestion status, and the smaller N value indicates that the network congestion situation is relatively slight and the value of the congestion window can take a larger value. On the contrary, the backlog value of the network is high and congestion may be serious. At this time, the value of congestion window is smaller. In the fast recovery phase, different size of the congestion window value is given for different congestion conditions, which is conducive to adjust the amount of data transmitted by TCP in time and make full use of the link of satellite network.

4. Simulation Results and Discussions

The enhanced TCP scheme compares the performance of various TCPs through the simulation. The simulation topology is shown in Figure 1. The server is connected with the client through the ground gateway and satellite network. The simulation parameters are shown in Table 1.

4.1. Performance in the Presence of Random Losses. In the first simulation scenario, the buffer size of the gateway is set large enough to prevent congestion loss during file downloading. The throughput and response time are two important indexes to evaluate network performance. The response time is the time from sending a request to completing the response. Throughput is the ability to process tasks per unit time. Its unit is usually bit/sec or MB/sec. It takes system resources as the object. Therefore, the performance of the system directly affects the (theoretical) limit value of throughput. Generally, the shorter the average response time is, the greater the system throughput is. The longer the average response time is, the smaller the system throughput is. When the BERs change from 10^{-9} to 10^{-5}, Figure 2 shows the response time of four TCPs (TCP NewReno, Veno, Hybla, and proposed scheme). The response time increased with the increase of BER. In the low BER range, the performances of the four TCP schemes are similar. When BER increases, the response time increases sharply, and the proposed scheme is clearly better than the other three TCP schemes.

The wireless link of high-altitude satellite communication is easy to be interfered, such as ground interference, space interference, natural interference, and man-made interference, especially in the open satellite communication system, and the transparent transponder is more vulnerable to malicious interference. When BER is 10^{-5}, the response times of our proposed scheme and Hybla are significantly lower than those of NewReno and Veno, and the response time of our proposed scheme is slightly lower than that of Hybla. Hence, the response time of our proposed scheme is

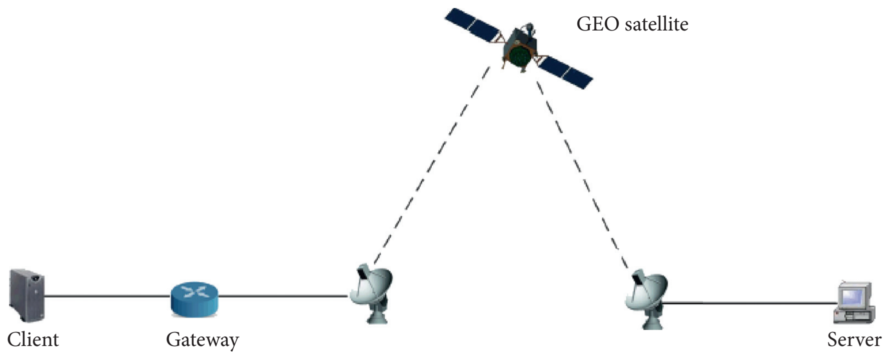

Figure 1: Network model.

Table 1: Simulation parameters.

Parameters	Value
Link rate of the server gateway	10 Mb/s
Uplink data rate	256 kb/s
Downlink data rate	2048 kb/s
One-way propagation delay	250 ms

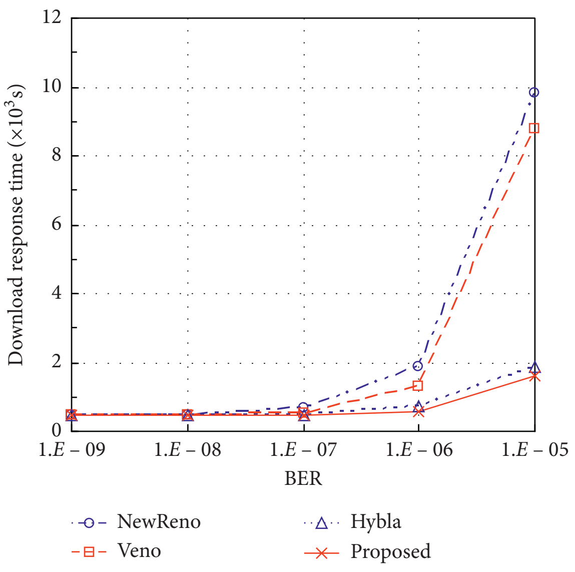

Figure 2: Response time for only random loss.

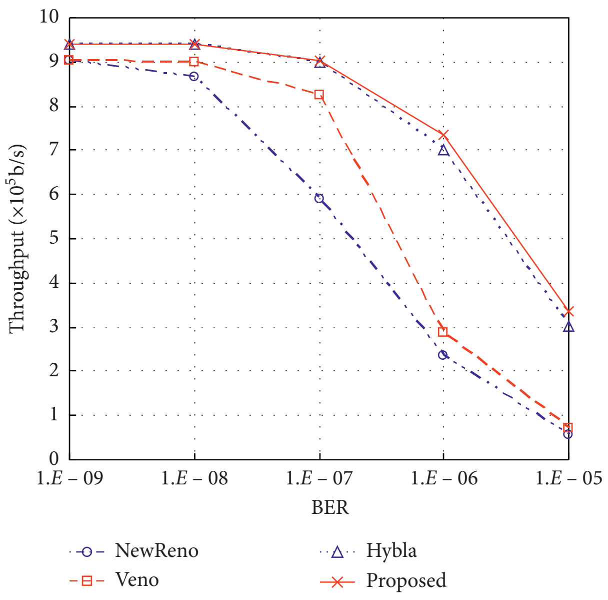

Figure 3: Throughput in the presence of random losses.

the lowest among the four TCP variants when there are random losses in the network. Figure 3 shows the satellite downlink throughput of four TCP variants versus BERs from 10^{-9} to 10^{-5}. It can be observed that the downlink throughput of the satellite network decreases with the increase of BER. When the BER changes from 10^{-9} to 10^{-7}, our proposed scheme exhibits comparable throughput to Hybla. The throughput rapidly decreases when the BER changes from 10^{-7} to 10^{-5}. With high BERs (10^{-7} to 10^{-5}), the throughput of our proposed scheme is slightly higher than that of Hybla and greatly higher than that of NewReno and Veno. From the simulation results, our proposed scheme can improve the throughput of GEO satellite links in the high BER range (10^{-6} to 10^{-5}).

4.2. Performance in the Presence of Both Random and Congestion Losses.

In the second simulation scenario, the buffer size of the gateway is set smaller (37.5 k bytes) such that congestion loss can happen during file downloading. When the BERs change from 10^{-9} to 10^{-5}, Figure 4 shows the response time of four TCPs (TCP NewReno, Veno, Hybla, and proposed scheme). Similarly, the response time increases with the increase of BER. In the low BER range, the performances of the four TCP schemes are also similar. When BER increases, the proposed scheme shows some advantages over the other three TCP schemes.

The response time of Veno is lower than that of NewReno if the BER is 10^{-5}. The time of Hybla is the longest, and the response time achieved by our proposed scheme is shortest among the four TCPs. At higher BERs, the large packet losses are mainly due to the random losses. It turns out that Hybla cannot effectively reduce the response time for both random and congestion losses. On the other hand, our proposed scheme can effectively improve TCP performance in the presence of both random and congestion

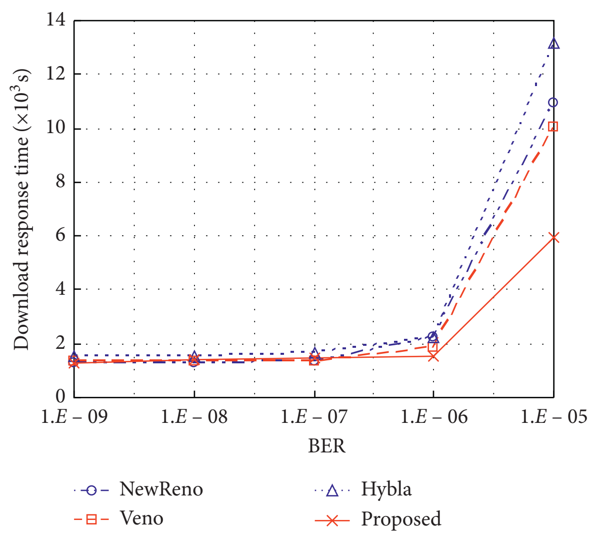

FIGURE 4: Response time for both random and congestion losses.

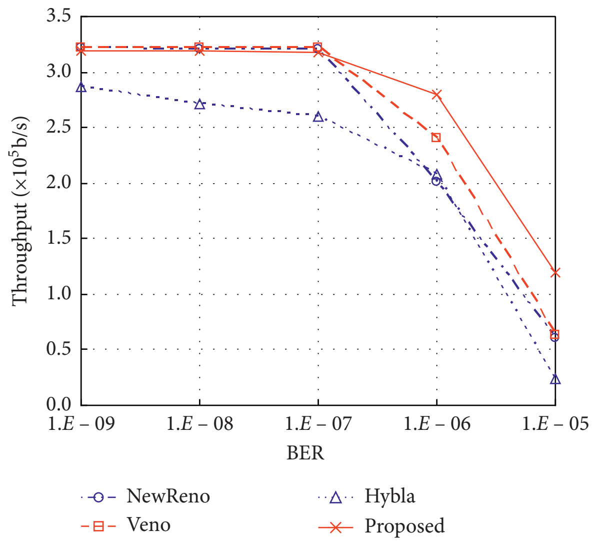

FIGURE 5: Throughput for both random and congestion losses.

losses. Figure 5 shows the satellite downlink throughput of the four TCPs versus BERs from 10^{-9} to 10^{-5}. It can be observed that the downlink throughput of the satellite network decreases with the increase of BER. When the BER changes from 10^{-9} to 10^{-5}, the throughput of Hybla is lower than that of the other three TCP variants. With higher BERs, the satellite link throughput of our proposed scheme exhibits the best performance among the four TCP variants. The above simulation results demonstrate that our proposed scheme can achieve smaller downloading time and higher throughput in the presence of both random and congestion losses. It inherently demonstrates that our proposed scheme updates the congestion window in more appropriate ways than NewReno, Hybla, and Veno. In the presence of both random and condensation losses when the BER is 10^{-5}, the throughput of the proposed scheme is 1.96 times, 1.88 times, and 5.09 times higher than TCP Reno, Veno, and Hybla, respectively.

5. Conclusion

Although there have been many studies for improving TCP performance on GEO satellite networks, there are few studies on TCP solutions that can handle both long RTT and high BER. This paper focuses on these two aspects, combining the advantages of TCP Hybla in dealing with long RTT and TCP Veno's good performance in dealing with high BER in wireless networks to deal with satellite networks and refining the congestion window.

In our proposed scheme, the congestion window update equations of Hybla are still employed in the slow start and congestion avoidance phases to mitigate the effect of long RTT. Our proposed scheme enhances Hybla by modifying the setting of RTT_0 (RTT of the reference connection) and allowing the congestion window to increase adaptively in the slow start phase corresponding to different networking conditions. Furthermore, our proposed scheme refines Veno's algorithm in fast recovery with multilevel differentiation for distinguishing different data losses. In the simulation test, the enhanced TCP scheme proposed in this paper can cope with random data loss and congestion data loss better than the other three schemes. The networking scenarios in the presence of only random packet loss and in the presence of both random and congestion losses are considered. The simulation results demonstrate that our proposed scheme exhibits better performance in the two networking scenarios compared with the other three TCP variants. The disadvantage of the proposed scheme is that the performance improvement is not very good at low BER, and there is no obvious advantage in throughput and response time. Finally, it needs to point out that our proposed scheme is an end-to-end approach for improving TCP performance over GEO satellite networks, where only the mechanisms of the TCP sender need to be modified, and the end-to-end semantics of TCP can be maintained, which has great value for large-scale practical applications, especially for the communication service providers and users who need to save communication costs [27].

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Hunan Provincial Natural Science Foundation (Grant no. 2020JJ40509), the Scientific Research Fund of Hunan Provincial Education Department (Grants nos. 19B512, 19A446, and 18B420), and the National Natural Science Foundation of China and Macau Science and Technology Development Joint Fund (Grants nos. 61961160706 and 0066/2019/AFJ).

References

[1] W. Zhang, G. Yang, F. Jiang et al., "Licklider transmission protocol for GEO-relayed space internetworking," *Wireless Networks*, vol. 25, no. 7, pp. 3747–3757, 2019.

[2] J. Liu, Z. Han, and W. Li, "Performance analysis of TCP new Reno over satellite DVB-RCS2 random access links," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 435–446, 2019.

[3] T. R. Henderson and R. H. Katz, "Transport protocols for Internet-compatible satellite networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 2, pp. 326–344, 1999.

[4] M. Allman, D. Glover, and L. Sanchez, "Enhancing TCP over satellite links using standard mechanisms," RFC 2488, 1999.

[5] M. Allman, S. Dawkins, D. Glover et al., "Ongoing TCP research related to satellites," RFC 2760, 2000.

[6] K. Ramakrishnan and S. Floyd, "A proposal to add explicit congestion notification (ECN) to IP," IETF RFC 2481, 1999.

[7] M. Allman and W. Stevens, "TCP congestion control," IETF RFC 2581, 1999.

[8] S. Floyd and T. Henderson, "The NewReno Modification to TCP's fast recovery algorithm," IETF RFC 2582, 1999.

[9] S. Floyd, T. Henderson, and C. Partridge, "Increasing TCP's initial window," IETF RFC 2414, 1998.

[10] I. Biswas, "An investigation on TCP large initial window," *International Journal of Satellite Communications and Networking*, vol. 31, no. 3, pp. 111–121, 2013.

[11] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgment options," IETF RFC 2018, 1996.

[12] C. Caini and R. Firrincieli, "TCP Hybla: a TCP enhancement for heterogeneous networks," *International Journal of Satellite Communications and Networking*, vol. 22, no. 5, pp. 547–566, 2004.

[13] C. Caini and R. Firrincieli, "End-to-end TCP enhancements performance on satellite links," in *Proceedings of the 11th IEEE Symposium on Computers and Communications*, pp. 1031–1036, Cagliari, Italy, June 2006.

[14] L. S. Brakmo and L. L. Peterson, "TCP Vegas: end to end congestion avoidance on a global internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, 1995.

[15] P. C. Fu and S. C. Liew, "TCP Veno: TCP enhancement for transmission over wireless access networks," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 2, pp. 216–228, 2003.

[16] C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP Westwood: endto-end congestion control for wired/wireless networks," *Wireless Networks*, vol. 8, no. 5, pp. 467–479, 2002.

[17] M. Luglio, M. Y. Sanadidi, M. Gerla, and J. Stepanek, "Onboard satellite "split TCP" proxy," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 2, pp. 362–370, 2004.

[18] E. Rendon-Morales, J. Mata-Diaz, J. Alins, J. L. Munoz, and O. Esparza, "Cross-layer architecture for TCP splitting in the return channel over satellite networks," in *Proceedings of the IEEE 6th International Symposium on Wireless Communication Systems*, pp. 225–229, Tuscany, Italy, 2009.

[19] E. Dubois, J. Fasson, C. Donny, and E. Chaput, "Enhancing TCP based communications in mobile satellite scenarios: TCP PEPs issues and solutions," in *Proceedings of the IEEE 5th Advanced Satellite Multimedia Systems Conference (Asma) and the 11th Signal Processing for Space Communications Workshop (Spsc)*, pp. 476–483, Cagliari, Italy, June 2010.

[20] A. Pirovano and F. Garcia, "A new survey on improving TCP performances over geostationary satellite link," *Network & Communication Technologies*, vol. 2, no. 1, pp. 1–18, 2013.

[21] V. Paxon and M. Allman, "Computing tcp's retransmission timer," IETF RFC 2988, 2000.

[22] J. C. Hoe, "Improving the start-up behavior of a congestion control scheme for TCP," *ACM SIGCOMM Computer Communication Review*, vol. 26, no. 4, pp. 270–280, 1996.

[23] L. Xu, H. Wang, and T. A. Gulliver, "Outage probability performance analysis and prediction for mobile IoV networks based on ICS-BP neural network," *IEEE Internet of Things Journal*, p. 1, 2020.

[24] C. Caini and R. Firrincieli, "Packet spreading techniques to avoid bursty traffic in satellite TCP connections," in *Proceedings of the IEEE 59th Vehicular Technology Conference. VTC 2004-Spring (IEEE Cat. No.04CH37514)*, pp. 2906–2910, Ottawa, Canada, May 2004.

[25] M. K. Park, M. S. Shin, D. G. Oh, B. C. Kim, and J. Y. Lee, "TCP Hybla+: Making TCP More Robust against Packet Loss in Satellite Networks," in *Proceedings of the International Conference on Computational Science and Its Applications ICCSA2011*, pp. 424–435, Santander, Spain, June 2011.

[26] H. Wang, L. Xu, Z. Yan, and T. A. Gulliver, "Low complexity MIMO-FBMC sparse channel parameter estimation for industrial big data communications," *IEEE Transactions on Industrial Informatics*, p. 1, 2020.

[27] J. Vankka, "Performance of satellite gateway over geostationary satellite links," in *Proceedings of the 2013 IEEE Military Communications Conference*, pp. 289–292, San Diego, CA, USA, November 2013.