Buffer Management for TCP/NC Tunnel in Lossy and Congested networks

Nguyen Viet Ha and Masato Tsuru
Graduate School of Computer Science and System Engineering
Kyushu Institute of Technology
680-4 Kawazu, Iizuka-shi, Fukuoka, Japan
nguyen.viet-ha503@mail.kyutech.jp

Abstract: Tunneling scheme based on Transmission Control Protocol with Network Coding (TCP/NC Tunnel) is a potential proposal for applying TCP/NC on networks having the unable implementing TCP/NC end devices and saving the bandwidth in across loss-free but thin networks. However, it currently lacks an efficiency packet dropping method in network congestion resulting in causing many TCP timeouts at end devices, and non-stop increasing the number of redundancy packets on tunneling. It also lacks the method to drop unuseful retransmitted packets cause gateway buffer overloading. In this paper, we propose a new buffer management mechanism for TCP/NC tunnel system to solve all these issues.

Keywords: Network Coding, TCP/NC, Tunneling, Buffer management, Congestion control

Classification: Network management/operation

References

[1] C. P. 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. DOI:10.1109/JSAC.2002.807336
[2] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, “TCP westwood: Bandwidth estimation for enhanced transport over wireless links,” Proc. IEEE 7th annual International conference on Mobile Computing and Networking, Rome, Italy, pp. 287–297, Jul. 2001. DOI:10.1145/381677.381704
[3] J. K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher, and J. Barros, “Network coding meets TCP: Theory and implementation,” Proceeding of the IEEE, vol. 99, no. 3, pp. 490–512, Mar. 2011. DOI:10.1109/JPROC.2010.2093850
[4] N. V. Ha, K. Kumazoe, K. Tsukamoto, and M. Tsuru, “Benefits of multiply-cascaded TCP tunnel with network coding over lossy networks,” Proc. 15th International Conference on Wired/Wireless Internet Communications. Lecture Notes in Computer Science, pp. 247–258, Springer, Jun. 2017. DOI:10.1007/978-3-319-61382-6_20
[5] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, “The benefits of coding over routing in a randomized setting,” Proc. IEEE International...
1 Introduction

Transmission Control Protocol (TCP) has played an essential role in a wide of applications and been the leading option for reliable transmission due to its advantages on connection-oriented and congestion control features. However, TCP’s performance is poor on lossy networks when its congestion control mechanism considers packet loss to be a congestion signal and decreases the sending rate by reducing the congestion window (CWND). Some TCP variants use the Round Trip Time (RTT) monitoring based to distinguish congestive and non-congestive losses to enhance performance over wireless networks such as TCP Veno [1], TCP Westwood+ [2]. However, they are only useful in the small link loss rate. Besides, TCP with Network Coding (TCP/NC) [3] is a potential proposal. TCP/NC enables a proactive recovery of lost packets with the redundant transmission of coded packets in cooperation with a reactive recovery of lost packets with retransmission based on the standard TCP Acknowledgment (ACK) mechanism, to maintain the goodput correctly in lossy networks. Nevertheless, TCP/NC is still not standardized and deployed in standard OSs. Besides, in some specify networks such as IoT (Internet of Things), end-devices which are tiny is hard to implement TCP/NC in itself because of limitations of the hardware. Therefore, we proposed an IP tunneling system across lossy networks using the TCP/NC (TCP/NC tunnel) without any change of protocol stack of end-devices [4].

In network congestion case, the gateway running TCP/NC (TCP/NC gateway) must drop the packet right at TCP sending buffer ($B_{tcp}$). If not, the packet is dropped at link buffer, and lead to two severe problems. First, the system identifies the packet loss due to the channel and increases the number of redundancy combinations mistakenly. That is a non-stop increasing because the congestion more and more becomes worse. Second, the packet passed over the NC layer is guaranteed to transmit successfully to destination TCP/NC gateway. Therefore, End-devices never see any packet loss via triple-duplicate-ACKs. End-devices only retransmit the packet when TCP timeout happens, resulting in long adapt with packet loss as well as network congestion. The previous studies limited packet dropping at link buffer by trying to increase the $B_{tcp}$ size. However, how much need to increase is hard to answer. Also, inaccurately dropping still happens, especially at a high degree of network congestion. Besides, the issue of Dead-packet did not solve, which severely affects the goodput performance of TCP/NC tunnel system.

Therefore, we propose a new scheme to determine when the buffer be full to fairness dropping the overload packets before adding to $B_{tcp}$ and distin-
guish the useful and unuseful duplicated packets to correctly dropping the Dead-packet to save the $B_{tcp}$ size.

2 TCP/NC tunnel overview

2.1 TCP/NC

In TCP/NC, a transparent NC sub-layer is complemented and placed between TCP and network layer. It handles the incoming and outgoing packets from TCP and network layer, respectively. TCP/NC allows the source to send $m$ combinations ($C$) created from $n$ original packets ($p$) with $m\geq n$ using Eq. (1). The coefficient $\alpha$ is selected randomly based on Random Linear Network Coding algorithm [5]. If the number of lost combinations is less than $k=m-n$, the sink can recover all the original packets using the received combinations without retransmission. Thus, the goodput is not affected by the lossy channel.

$$C_i = \sum_{j=a}^{b} \alpha_{ij}p_j; \quad i=1,2,...,m; \quad a= \begin{cases} 1 & (i \leq k) \\ i-k & (i> k) \end{cases}; \quad b= \begin{cases} i & (i \leq n) \\ n & (i>n) \end{cases} (1)$$

2.2 TCP/NC tunnel

The protocol stack and structure of TCP/NC tunnel are illustrated in Fig. 1. The TCP/NC gateway includes two interface types with the “internal” and “external” interface connect to the local network and other networks, respectively. If an IP packet comes from the internal interface and goes to the external interface, this packet is encapsulated with TCP/NC protocol and forward into TCP/NC tunneling to the adjacency TCP/NC gateway. An application called tunnel handler (handler) controls all the processes. The packet loss is recovered in each tunneling segment and forwarded to the next TCP/NC gateway. At the final destination TCP/NC gateway, the handler forwards the decoded packet to an end-host as an original IP packet. TCP/NC gateways do not interfere with the TCP establishment phase as well as the ACK returning process between TCP end-hosts. When packet
losses happen in local networks (outside the tunnel), E2E-TCP recovers them by TCP’s retransmission.

3 Buffer management

3.1 Congestion control

As mentioned in Sect. 1, to mitigate network congestion at the ingress gateway, overloaded packets and unnecessary duplicate packets should be dropped before entering TCP/NC tunnel, that is, dropped at TCP sending buffer ($B_{tcp}$). In this study, we propose a new buffer management method using cross-layer information such as the instance values of redundancy factor ($R$), queue size in $B_{tcp}$, CWND size, and the remaining space ($A_L$ in packets) of $B_{link}$, which are collected by the handler. To avoid $B_{link}$ overloaded, the handler must check $A_L$ before sending a new packet ($p$) to $B_{tcp}$. Note that the NC layer amplifies a single TCP packet to $R$ combinations on average and send to the Link layer. If $A_L$ is smaller than $\lceil R \rceil$, sending $p$ to $B_{tcp}$ has a risk of $B_{link}$ overflow because it may be amplified and forwarded to $B_{link}$ immediately. On the other hand, if the remaining size ($A_w$) of CWND (that is CWND minus the queue size in $B_{tcp}$) is shorter than the packet size of $p$, TCP does not send any new packet to $B_{link}$ because many packets are waiting for ACK. Therefore, when $p$ comes to the handler from upstream networks, if $A_L$ is smaller than $\lceil R \rceil$ and $A_w$ is equal or greater than the packet size of $p$ at the same time, the handler drops $p$. Otherwise, $p$ is also checked and dropped if it is dead-packet (Sect. 3.2). After those screening, the handler finally puts $p$ into $B_{tcp}$ as a harmless packet.

3.2 Dropping dead-packet

The Sequence number and arrival time of the packet in $B_{tcp}$ is stored in the Packet Information table with the size greater than the size of $B_{tcp}$ (e.g., 255 records in this study). We divide this table into four parts (cases) corresponding to the place of the duplication ($p_d$) of the new coming packet ($p$) as shown in Fig. 2. $A$ is set to the initial minimum TCP timeout of 1 s (following RFC 6298 [6]) and $B$ is double of $A$. If $p$ has a duplication ($p_d$), handler calculates the distance $L$ (in packet) from the left of Packet Information table to $p_d$ and read the arrival time of $p_d$ ($T_d$).

Case 1: if $L$ less than or equal the size of the “Currently storing packets”
(Lc), pd is waiting for an ACK packet and be handled by NC layer because it still stays in the TCP buffer. Handler drops p.

Case 2: if Lc > L and Td < A, pd is acknowledged in the current TCP/NC tunnel, but it is lost at other gateways (congestion case – dropped before adding to TCP Sending buffer) or at the sink. p might be sent due to dup-ACKs retransmission at source; thus, handler forwards pn.

Case 3: if Lc > L and A ≤ Td < B, pd is transferring in the network but the E2E TCP time-out happened. Handler drops pn.

Case 4: if Lc > L and B ≤ Td, pd has been lost and the E2E TCP timeout happened. pn must be retransmitted.

4 Simulation results

4.1 Simulation settings

The simulation is accomplished using Network Simulator 3 (ns-3) [7]. The topology is a dumbbell network model with three gateways connect to at most twelve sources and twelve sinks. Each edge link connects an end-host and a gateway has a bandwidth of 1 Mbps and a propagation delay of 1 ms. The intermediate links connecting two neighbor routers have a bandwidth of 1 Mbps and a propagation delay of 10 ms. The B_{link} size is set to 100 packets, and the packet size is 536 bytes. The transferred data size is 100 Mbytes. The TCP type is NewReno. The intermediate link is considered as a lossy channel of random loss channel with a link loss rate ranging from 0.0 to 0.2 in the direction of transferred data.

B_{tcp} is set to the value of maximum congestion window plus the expected buffer size depended on the demand for the delay. In this study, we do not investigate the delay; therefore, we set B_{tcp} to the total of 114+100 packets. All simulations were performed at least 20 times to obtain the average value.

4.2 Goodput comparison

The goodput performance of the proposal, which includes congestion control and dropping Dead-packet mechanisms, is better than the case of using only one on two schemes shown in Fig. 3. Without dropping dead packet mechanism, the goodput performance is small when the degree of network congestion is high (e.g., in twelve sessions case). Without congestion control mechanism, the performance is a little smaller than the proposed method, but the goodput performance at no link loss is significantly worse than the proposed method.

Compare to E2E-TCP, the performance of TCP/NC tunneling is better in cases of high link loss rate such as more than 0.02 with three sessions and 0.06 with twelve sessions. In such cases, no more congestion happens at E2E-TCP. Based on these results, TCP/NC tunnel should control redundancy factor R based on the congestion degree and the link loss rate. However, this issue will be investigated and solved in the future.

Compare to E2E-TCP/NC, the performance of TCP/NC tunnel is similar on three sessions case but much better in twelve sessions case. It is a
reasonable behavior when E2E-TCP/NC keep increasing $R$ when network congestion happens; thus, the network becomes more congested. While the TCP/NC tunnel can drop packet right at $B_{tcp}$ to avoid mistaken increasing $R$.

5 Conclusion

In this study, we proposed a new buffer management scheme for TCP/NC tunnel system. It can intentionally drop overloaded packets or useless dead-packets before they enter the TCP/NC tunnel. The simulation results show that the proposal can achieve a good goodput compare to others. In future work, we will investigate and optimize $A$, $B$, and also the size of $B_{tcp}$ based on estimated RTT by using the TCP timestamp field.

Acknowledgment

The research results have been achieved by the “Resilient Edge Cloud Designed Network (19304),” the Commissioned Research of National Institute of Information and Communications Technology (NICT), and by JSPS Grant-in-Aid for Scientific Research (KAKENHI) Grant number 19K21535 and 16K00130, Japan.