Integration of Multipath TCP and Network Coding for Satellite Networks

Torsten Braun[1], Pengyuan Du[2], Mario Gerla[3]

***Abstract*— This paper investigates how to support reliable satellite communication using TCP assuming that multiple satellite links are available and Multipath-TCP (MPTCP) as well as Software-Defined Networking (SDN) can be used to map sub-flows to different satellite links. To improve redundancy we propose integrating Network Coding into MPTCP. The paper discusses in detail the related work on MPTCP, SDN, Network Coding, and satellite communication, in particular certain combinations. We present a possible approach to integrate these for our target satellite communication scenario and discuss various design options.**

I. Introduction

Despite high coverage of cellular networks, satellite communication has still its importance, e.g. on the sea as well as in desert and mountain areas. However, errors such as link breaks can suddenly happen in satellite communication scenarios due to movement of mobile satellite terminals, shadowing effects etc. To address reliability, we propose using Multipath-TCP (MPTCP), which splits TCP connections into sub-flows, and Software-Defined Networking (SDN) for mapping these sub-flows to different paths, in this case to different satellite links. Figure 1 depicts our target scenario with multiple MPTCP sub-flows mapped to different satellite links by an SDN controller. We propose to use Network Coding (NC) to achieve additional protection of link breaks and other failures. Network Coding can avoid delays by MPTCP retransmissions. After discussing related work in Section II, we present our proposed MPTCP/SDN/NC operation in Section III-B. The contribution of this paper is that it concretely describes an original way how all these technologies can be used in a satellite communication scenario, including the needed modification of MPTCP options. Section IV concludes the paper.

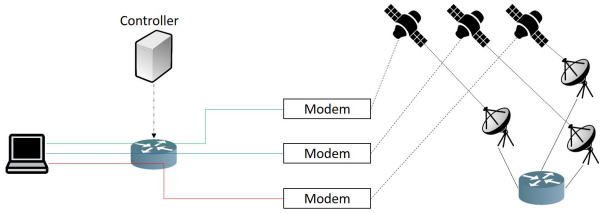

Fig. 1. Target satellite communications scenario

[1]T. Braun is with the Institute of Computer Science, University of Bern, 3012 Bern, Switzerland braun@inf.unibe.ch
[2]Pengyuan Du is with the Department of Computer Science, UCLA, Los Angeles, CA 90095, USA pengyuandu@cs.ucla.edu
[3]M. Gerla is with the Department of Computer Science, UCLA, Los Angeles, CA 90095, USA gerla@ucla.edu

II. Related Work

A. Multipath TCP

Multipath TCP (MPTCP) separates the data stream of an application into separate sub-flows, which can then be routed over heterogeneous paths, e.g., using SDN mechanisms [1]. Initially, TCP connections are established using the traditional TCP three-way-handshake [2], [3], where the involved parties signal each other by the MP_CAPABLE option in SYN and SYNACK messages that they are supporting MPTCP. Sub-flows can then be added using the MP_JOIN option. Sub-flows can be identified based on the quintuple (protocol, IP source address, IP destination address, source port, destination port). Each sub-flow uses conventional TCP signaling, i.e. the 32-bit sequence and acknowledgment numbers of each sub-flow's data stream.

On the connection level, MPTCP adds a 32 or 64-bit Data Sequence Signal (DSS) as an option (Figure 2), which describes how the sub-flow's sequence space maps to the connection level [4]. The mapping consists of a Data Sequence Number (DSN), a Sub-flow Sequence Number (SSN), and a Data-Level Length (DLL). DLL specifies for how many bytes of the data stream the mapping is valid. The DSS option does not need to be present in each packet. It includes a 32 or 64-bit (dependent on the DSN length) connection-level acknowledgement (DATA ACK), which is a cumulative ACK for the connection and indicates as in standard TCP how much data has been received without holes. On the sub-flow level ACKs acknowledge the reception of the data for that particular sub-flow. DATA ACKs on connection level and ACKs on sub-flow level are not necessarily aligned to each other. A receiver might acknowledge the reception of data on a sub-flow without progressing the DATA ACK. It might acknowledge the data on the sub-flow level and then drop the data, e.g., in case of memory constraints of the connection's receive buffer. Then, retransmissions on connection level are needed, i.e. the missing data must be retransmitted over one of the sub-flows.

MPTCP has been mainly designed to increase reliability by having multiple paths, e.g., in handover scenarios or link breaks, but not primarily for increasing throughput. In [5] different modes are proposed: Normal mode exploits all available sub-flows. Backup mode uses only a subset of sub-flows and leaves pre-established backup sub-flows for failure situations. In case of link or path breaks, MPTCP can be beneficial by rescheduling packets from the failed link / path to properly working paths. This requires an appropriate scheduler at the TCP sender. [6] discusses several

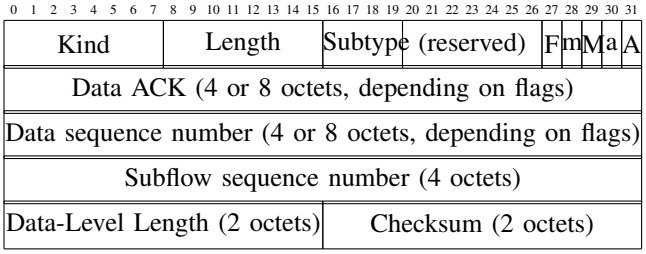

Fig. 2. MPTCP DSS Option

options for schedulers such as Round-Robin, Lowest-RTT-First, Retransmission and Penalization (RP), and Bufferbloat Mitigation. RP reinserts possibly lost segments causing head-of-line blocking on a sub-flow with space in its congestion window.

B. (Multipath) TCP in Satellite Networks

Early research on TCP over satellite networks focused on improving the performance of standard TCP over satellite networks with high bandwidth delay product [7], [8]. MPTCP can be used to separate sub-flows of a single connection, but a MPTCP connection should not be more aggressive than a single TCP connection. This is achieved by coupled congestion control [9], which ensures that a single sub-flow does not get more bandwidth than a single TCP connection. The main advantage of MPTCP in satellite networks is that it can avoid suffering from single satellite link disruptions and better exploit the available capacity in case of multiple available satellite links. In [10] authors propose the On-demand Multipath Source Routing (OMSR) protocol to support routing of MPTCP sub-flows across different paths of a LEO (Low Earth Orbit) satellite, in particular to support smooth hand-overs. In [1] it is proposed to use a SDN controller to control the network path of MPTCP sub-flows. Otherwise, it might happen that all or multiple sub-flows are routed over the same path, i.e. the same satellite channels, even if multiple paths exist and, thus, can not exploit the available bandwidth of different multiple satellite channels.

C. (Multipath) TCP and Network Coding

CoMP is a network coding multipath forwarding scheme to exploit path diversity in wireless mesh networks (WMNs) [11]. It sits below TCP/IP and can even re-encode packets in intermediate WMN nodes. CoMP tries to estimate the WMN's loss probability and adjust the rate of sending linear combinations. CodeMP [12] applies network coding to support TCP in multi-path mobile ad-hoc networks.

In [13] authors propose a scheme for integration of TCP and Network Coding. It is proposed to transmit linear combinations of packets instead of original packets in a congestion window. The proposed Network Coding layer is put in-between the TCP and IP protocol. The paper also proposes to change the TCP semantics. Acknowledgments are sent per received packet, i.e. per received linear combination of packets. In contrast to traditional TCP semantics an ACK does not acknowledge the reception of an original packet that can be delivered to the application. To reconstruct an original TCP packet, the receiver has to receive a certain number of linear combinations containing the packet. An advantage of the approach is that no packets are retransmitted in an identical form and the congestion window can permanently progress for each received ACK. TCP Vegas has been proposed as TCP congestion control mechanism in this case.

The value of combining Network Coding and MPTCP has been demonstrated in [14], where a scenario with three different network interfaces, namely WiMax, WiFi, and Iridium satellites, of a mobile client has been considered. This can be considered as multi-path extension of [13]. None of these networks provide reliable communication but by combining these networks as well as using MPTCP and Network Coding, reliability can be improved. The paper analyses packet loss and round-trip-times over these networks and based on that derives an analytical model to estimate the end-to-end throughput of MPTCP using Network Coding. Two sub-layers with Network Coding are proposed, one at MPTCP level, i.e. above the sub-flows, and one on sub-flow level to overcome random packet loss in sub-flows. The analytical results suggest the use of MPTCP and Network Coding, but the work does not explicitly describe how to implement this approach and how protocol headers need to be designed. Moreover, the problem of violating TCP semantics is not addressed. Delays can be introduced by requiring the receiver to decode a certain number of packets until an original packet can be reconstructed.

To avoid the TCP semantics problem, the authors in [15], propose to perform Network Coding on network layer, i.e. IP, completely independent from MPTCP and TCP. In that case, only decoded packets will be delivered to the TCP receiver, which can add significant delays.

The problem of delayed decoding can be avoided by not coding all packets, but only a fraction of the packets [16]. Authors of [16] propose to schedule the original packets over multiple MPTCP sub-flows and use a dedicated sub-flow for carrying redundant coded packets. Although not explicitly mentioned in the paper, this approach can be used to overcome the TCP semantics problem with Network Coding. However, focusing redundancy on a single sub-flow can generate some problems, when this particular sub-flow and its path are suffering from heavy loss. In that case, lost packets of other sub-flows can not be repaired.

The work presented in [17] avoids this problem. There, redundant packets are distributed over all sub-flows. The authors propose to use generations of packets and transmit a linear combination of packets out of a single generation at the beginning, i.e. before the original packets are sent over a sub-flow. The approach is called SC-MPTCP (SC: systematic coding), since redundant packets are combined with original packets. This allows to reconstruct a lost original packet and minimizes the number of original packets to be buffered at the receiver side compared to an approach when linear combinations are transmitted at the end of a generation. The

problem, however, is then that a single linear combination at the beginning of a generation can only compensate for one packet loss in a generation of packets. The number of tolerable packet losses can be increased with the transmission of an according number of linear combinations at the beginning. However, this requires some estimation of the expected packet loss at the beginning of a generation. Otherwise, in case of underestimated packet loss, not all packets can be repaired, and in case of overestimated packet loss, too many redundant packets might be transmitted. An efficient scheme depends on accurate packet loss estimation, e.g., based on exponential averaging as proposed in [17], but in particular in satellite networks, this may be difficult to estimate, since packet losses are rather generated by sudden connection disruptions and in that case exponential averaging might not be meaningful. To recover packet loss, the authors propose a pre-blocking warning mechanism, which sends a message from the receiver to the sender in order to ask for additional redundant, network coded packets, if the receiver detects a head-of-line blocking situation in the receive buffer. Evaluations have been performed using network simulation, but no protocol specification or implementation is provided.

D. Multipath TCP and Network Coding in Satellite Networks

The issue of generations has been discussed in [18], where a sliding window for network coding instead of generations has been suggested. This means that instead of generating linear combinations out of fixed size generations, the coded packets can be taken from a sliding window that can be continuously adapted dependent on the reception progress of the receiver. This requires continuous feedback from the receiver to the sender. The paper states based on simulation experiences that the sliding window approach has superior performance compared to generations, but accurate control of the sliding window is essential for performance improvements. This is of course given in a TCP scenario, but might experience delays in satellite networks.

A similar architecture as in [14] has been proposed in [19], i.e. Network Coding (NC) on both MPTCP and sub-flow level. It is proposed to deploy MPTCP/NC functionality into middle-boxes, i.e. so-called Performance Enhancing Proxies (PEPs). The proposed approach avoids signaling of encoded packets by using a pseudo-random function at both ends of the TCP connection to generate coding vectors. However, this does not allow flexible reaction on particular lost packets or burst errors. The approach adopts the concept of sliding windows instead of generations. The MPTCP/NC layer returns a DATA ACK for any new innovative packet, which is still violating TCP semantics. Simulations used two satellite links assuming an on/off channel model with on/off phases in the range of seconds have been performed.

III. Integrating MPTCP and Network Coding in Satellite Networks

A. Proposed MPTCP/NC Operation

Figure 3 describes the proposed MPTCP/NC operation of the sender side. The Network Coding component takes packets from the byte stream, performs network coding, and distributes coded packets to the various sub-flows according to their filled transmit buffers as well as a redundancy strategy. The packets to be coded need to be of the same length and the start of each packet should be a multiple of the packet length plus the initial offset although an original packet could start at any byte position. However, this would make decoding rather difficult. The start of each encoded original packet must be indicated in the encoded packet. We propose to modify the MPTCP DSS option as described in subsection III-B. The Network Coding coder should consider the acknowledged bytes for selecting which packets as discussed in Subsection III-D.

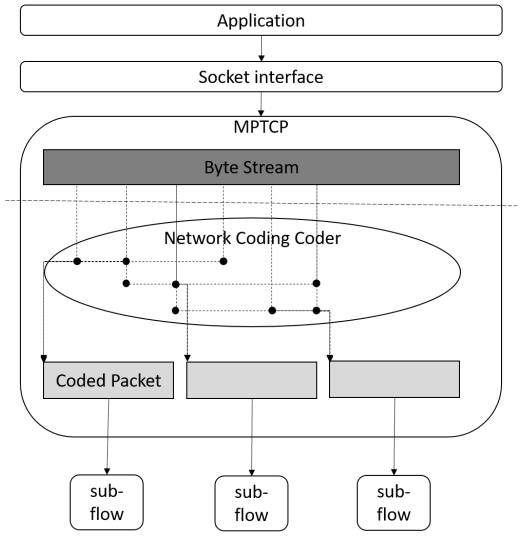

Fig. 3. MPTCP Operation at Sender

The operation of the receiver side is depicted in Figure 4. Coded packets are stored in the coded packet buffer. The decoder part analyses the received coded packets, reconstructs original packets, and stores them in the receive buffer. If a reconstructed packet increases the minimum sequence number until which all original data have been received without any holes, a new DATA ACK must be sent. Acknowledgements on sub-flow level are exchanged independent of whether original data can be reconstructed or not. Reconstructed original packets can be removed from the coded packet buffer, e.g., we can replace (A + B + C) by (B + C) after reconstructing A.

B. Proposed Protocol Changes for MPTCP/NC

The main issue is to define a Network Coding header to indicate which original packets contribute to the coded packet. RFC6363 [20] defines how to encode Forward Error Correction (FEC) data when transmitted over (unreliable) transport protocols such as DCCP or UDP. We propose to reuse the MPTCP DSS Option to specify which original packets are encoded in the Network Coded packet. One could use random linear network coding with a random number generator, which would allow us to minimize overhead for signaling which original packets are encoded in a packet.

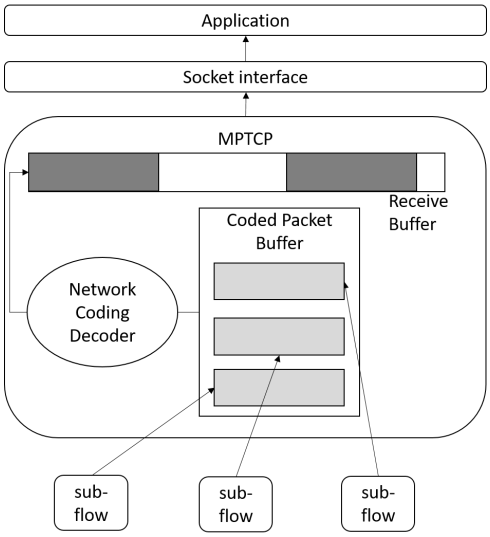

Fig. 4. MPTCP Operation at Receiver

0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19	20 21 22 23 24 25 26	27	28	29 30	31
Kind	Length	Subtype	(reserved)	F	1	M0	A
Data ACK (4 octets)							
Data sequence number of first original packet (4 octets)							
NC Packets	Offset (3 octets)						
Subflow sequence number (4 octets)							
Data-Level Length (2 octets)		Checksum (2 octets)					

Fig. 5. MPTCP DSS Option for Network Coding

0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31
0	Coefficient 0	Packet No. 1	Coefficient 1
Packet No. 2	Coefficient 2	Packet No. 3	Coefficient 3
...			
Packet No.N-1	CoefficientN-1	Packet No.N	Coefficient N

Fig. 6. Network Coding Header

However, this would limit the sender to control the packets to be encoded and we do not like to loose this flexibility, in particular when retransmissions of specific packets are needed, e.g., in case of sudden satellite link failures. Systematic coding is another option, but requires some knowledge about expected error rates. This is might be difficult to achieve in satellite networks with sudden link breaks.

We propose to signal in the MPTCP DSS option which original packets have been encoded. We use the Data Sequence Number in the MPTCP DSS option for this. We call the Data Sequence Number field "Data Sequence Number for Network Coding", see Figure 5. The first sub-field (4 octets) describes the "Data Sequence Number of the first original packet" that is encoded in the Network Coding packet. "NC Packets" indicates how many original packets contributed to the encoded packet. The Network Coding header (see Figure 6) is put at the beginning of the user data and includes "Offset" bytes. "Offset" indicates the offset of the user data in the MPTCP data.

The Network Coding header as shown in Figures 6 starts with a "0" for the first encoded packet beginning at the sequence number as described in Figure 5. For the first packet we describe the coding coefficient thereafter. Then, we have 2N octets for each additional original packet, describing the relative position (as packet index) of the additional packets compared to the first one (0). The second octet describes the coefficient of that packet. Two padding bytes should be added to achieve word alignment.

C. MPTCP Deployment in Satellite Networks

In our target scenario depicted in Figure 7 we assume end systems, e.g., located at ships, connected via several modems to a certain number of satellites. We assume a router with a scheduler component connected to the various satellite modems. Such schedulers inside the routers are assumed to assign priorities to packets and schedule them to the available satellite links. We also assume a controller system (possibly a set of distributed controllers and a central controller) with some global knowledge about the bandwidth values assigned to the satellite links. The controller(s) and routers are also assumed to learn quickly about satellite outages such that they can reschedule packets to alternative links.

The TCP connections and sub-flows are terminated at the end systems and according to [1] an SDN-enabled switch assigns MPTCP sub-flows to different satellite links. This, however, creates a problem, since the assigned bandwidth on satellite links might vary and links can break. While the router's scheduler might know that quickly, the end system does not have this information. MPTCP will adapt after some time to the bandwidth available on the path to which a sub-flow is assigned. This takes some time, and if bandwidth changes significantly or loosing satellite link connections, congestion might occur or packets might get lost.

An alternative is to have the MPTCP entity not in end systems but co-locate the proxy mapping TCP connections to MPTCP sub-flows at the router. This approach has been similarly proposed in [19], where PEPs combine TCP to MPTCP sub-flow mapping and network coding. However, as discussed before, this violates the TCP semantics as a received ACK at the sender does not mean that the destination has received and decoded an original packet.

Another issue is caused by the combination of network coding and MPTCP. Assuming that the sender applies network coding to packets of the MPTCP connection, it might be beneficial to encode original packets across encoded packets that are sent over different sub-flows. Otherwise, if a satellite link breaks, the original packet can not be recovered without retransmissions. Again, a PEP-based MPTCP approach would have some advantage since Network Coding and MPTCP would be co-located and network coded packets can be better mapped to satellite links.

We propose three solutions addressing these issues. The first is based on splitting the MPTCP sender implementation between end system and an intermediate proxy. The second and third option keep the MPTCP implementation at the end system.

1) **Proxy controlled.** The first option is inspired by [21], [22] and splits the MPTCP sender functionality into two parts, which are then distributed between the sender in the end system and an intermediate proxy. We split the two parts at the horizontally dashed line of Figure 3 and move the lower part (network coding and sub-flows) to a proxy co-located with the scheduler, while the upper part (socket interface and MPTCP connections) remains in the end system. Both parts have to communicate with each other to deliver MPTCP connection data, but also to establish or remove MPTCP sub-flows. For this, TCP or UDP tunneling can be used.
2) **SDN controlled.** The second option keeps all MPTCP functionality at the end system. If a satellite link breaks, the SDN controller takes appropriate actions to map affected MPTCP flows to other links, whereas the MPTCP sender generates network coded packets and distributes them over the established sub-flows. If a satellite link breaks, the SDN controller can assign the affected sub-flow to another satellite link. If there has been a single sub-flow per link, the selected satellite link might become more congested than others, which might result in packet loss. If we would have several, let's say M, sub-flows (from a single or multiple MPTCP connections) on one satellite link and that link fails, then we could map the affected M sub-flows over the other remaining L-1 links. If there are M = L (L-1) sub-flows in total (L-1 sub-flows on each of the L links), then we can move each of the L-1 sub-flows affected by a failed link to one of the other L-1 links. As example, in Figure 7, there should be for L=3 satellite links at least 6 sub-flows in total. Please note that those L (L-1) sub-flows could be created by multiple MPTCP connections. The more sub-flows we have per MPTCP connection, the better can sub-flows rebalanced. However, too many sub-flows might create too much overhead at both end systems and SDN routers. Therefore, a third option is proposed.
3) **End system controlled.** To avoid a larger number of sub-flows, the complete control of distributing the network coded data over sub-flows is done in the end system. In this case, the MPTCP implementation will detect at some time, e.g., by full buffers, that a sub-flow does not work properly any more. The MPTCP sender in the end system should then map the packets not any more to the failed sub-flow using an appropriate scheduler, e.g., RP, but to other sub-flows. New sub-flows could be established in case of failed links, and the SDN controller should map those to different satellite links. To avoid delays caused by sub-flow establishment, MPTCP allows the establishment of backup sub-flows [23], which can be used when the regular sub-flow failed.

All approaches keep the TCP semantics, but have their pros and cons. The first option allows full control and full flexibility at the proxy. The proxy can directly schedule packets to MPTCP sub-flows and those to satellite links based on the status of those satellite links to which the proxy is directly connected to. However, the interaction between end system and proxy requires some additional message exchange and thus overhead. It also requires to split the end system's sender implementation. The second option does not alter the MPTCP implementation, except integrating a network coding strategy. The control at the proxy is not as fine-grained as before, since packets arrive already encoded. However, it allows to simplify scheduling by applying scheduling functions to sub-flows instead of to individual packets. For example, if a satellite link breaks, sub-flows over the broken link can be distributed among the still existing links. The third option leaves the decision of sub-flow establishment and the scheduling of packets to the end system, i.e. MPTCP. It might take a longer time until packets are rescheduled from the initial, broken sub-flow to a new one.

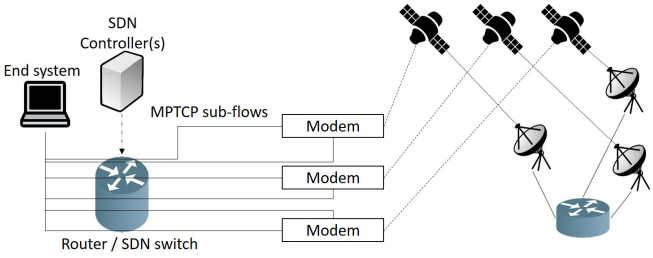

Fig. 7. MPTCP / Satellite Scenario

D. Network Coding Strategy for MPTCP/NC in Satellite Networks

The question is how decide which original packets are considered for a Network Coded packet. This decision should reflect typical satellite error characteristics. We assume that there are rather low probabilities for bit errors or single packet errors. We rather assume that satellite links may fail for a longer time, i.e. seconds or minutes.

Let us assume that the sender has a transmission buffer that is limited by the highest acknowledged byte (acknowledged by the MPTCP DATA ACK) and the congestion / flow control window. Within this buffer, there might be some data that have been sent recently, but the measured round-trip-time (RTT) is higher than the difference (actual time - time of transmission). Let us assume that a packet must be encoded K times (possibly with some additional margin ϵ) to allow a receiver to reconstruct it. As long as the packet has been encoded less than K + ϵ times, there should be a high probability for selecting that packet for an encoded packet. If the packet has been encoded more than K +

ϵ times and the difference D = (actual time - time of (K+ϵ)-th transmission) is smaller than the RTT, the selection probability might decrease to a minimum. As soon as D becomes larger than the RTT, the selection probability of the packet should gradually increase. As soon as an original packet has been acknowledged, the packet falls out of the transmission buffer and the selection probability falls to 0, see Figure 8.

Another issue is the decision whether to use generations or sliding windows for Network Coding. We think that the natural approach would be a window-based approach. The window is limited by the sequence number indicated in the Data ACK as well as the maximum congestion window as calculated by the coupled congestion control mechanism as specified in [9]. Each packet within that window should get a certain probability to be selected for a network coded packet.

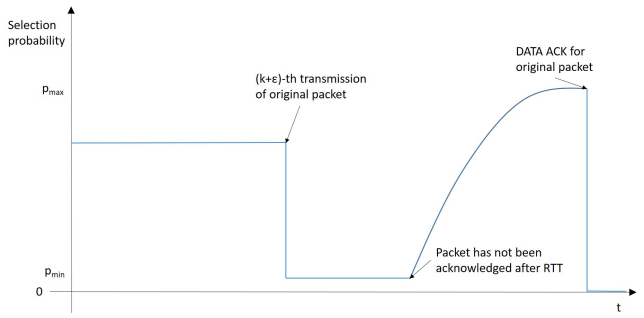

Fig. 8. Selection probability for original packets to be encoded

IV. CONCLUSIONS

This paper discussed proposed protocol changes and design options to support reliable communication in satellite networks using MPTCP, SDN, and Network Coding. The proposed design options need to be implemented and evaluated in order to select the most appropriate ones.

REFERENCES

[1] S. Nazari, P. Du, M. Gerla, C. Hoffmann, J. H. Kim, and A. Capone, "Software defined naval network for satellite communications (sdn-sat)," in *MILCOM 2016 - 2016 IEEE Military Communications Conference*, Nov 2016, pp. 360–366.

[2] G. Huston, "Multipath tcp," *Internet Protocol Journal*, vol. 18, no. 2, pp. 2–14, jun 2015. [Online]. Available: http://ipj.dreamhosters.com/wp-content/uploads/2015/07/ipj18.2.pdf

[3] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure, "Exploring mobile/wifi handover with multipath tcp," in *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design*, ser. CellNet '12. New York, NY, USA: ACM, 2012, pp. 31–36. [Online]. Available: http://doi.acm.org/10.1145/2342468.2342476

[4] A. Ford, C. Raiciu, M. J. Handley, and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses," RFC 6824, Jan. 2013. [Online]. Available: https://rfc-editor.org/rfc/rfc6824.txt

[5] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure, "Exploring mobile/wifi handover with multipath tcp," in *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design*, ser. CellNet '12. New York, NY, USA: ACM, 2012, pp. 31–36. [Online]. Available: http://doi.acm.org/10.1145/2342468.2342476

[6] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental evaluation of multipath tcp schedulers," in *Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop*, ser. CSWS '14. New York, NY, USA: ACM, 2014, pp. 27–32. [Online]. Available: http://doi.acm.org/10.1145/2630088.2631977

[7] M. Gerla, W. Weng, and R. L. Cigno, "Ba-tcp: a bandwidth aware tcp for satellite networks," in *Proceedings Eight International Conference on Computer Communications and Networks (Cat. No.99EX370)*, 1999, pp. 204–207.

[8] M. Luglio, M. Y. Sanadidi, M. Gerla, and J. Stepanek, "On-board satellite "split tcp" proxy," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 2, pp. 362–370, Feb 2004.

[9] C. Raiciu, M. J. Handley, and D. Wischik, "Coupled Congestion Control for Multipath Transport Protocols," RFC 6356, Oct. 2011. [Online]. Available: https://rfc-editor.org/rfc/rfc6356.txt

[10] P. Du, X. Li, Y. Lu, and M. Gerla, "Multipath tcp over leo satellite networks," in *2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Aug 2015, pp. 1–6.

[11] S. Gheorghiu, A. L. Toledo, and P. Rodriguez, "Multipath tcp with network coding for wireless mesh networks," in *2010 IEEE International Conference on Communications*, May 2010, pp. 1–5.

[12] C. C. Chen, G. Tahasildar, Y. T. Yu, J. S. Park, M. Gerla, and M. Y. Sanadidi, "Codemp: Network coded multipath to support tcp in disruptive manets," in *2012 IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS 2012)*, Oct 2012, pp. 209–217.

[13] J. K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher, and J. Barros, "Network coding meets tcp: Theory and implementation," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 490–512, March 2011.

[14] J. Cloud, F. du Pin Calmon, W. Zeng, G. Pau, L. M. Zeger, and M. Medard, "Multi-path tcp with network coding for mobile devices in heterogeneous networks," in *2013 IEEE 78th Vehicular Technology Conference (VTC Fall)*, Sept 2013, pp. 1–5.

[15] P.-L. Ageneau and N. Boukhatem, "Multipath tcp over network coding for wireless networks," in *14th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, January 2017, pp. 373–376.

[16] M. Li, A. Lukyanenko, and Y. Cui, "Network coding based multipath tcp," in *2012 Proceedings IEEE INFOCOM Workshops*, March 2012, pp. 25–30.

[17] M. Li, A. Lukyanenko, S. Tarkoma, Y. Cui, and A. Yl-Jski, "Tolerating path heterogeneity in multipath tcp with bounded receive buffers," *Computer Networks*, vol. 64, pp. 1 – 14, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128614000425

[18] J. Cloud and M. Médard, *Network Coding over SATCOM: Lessons Learned*. Cham: Springer International Publishing, 2015, pp. 272–285.

[19] G. Giambene, D. K. Luong, V. A. Le, and M. Muhammad, "Network coding and mptcp in satellite networks," in *2016 8th Advanced Satellite Multimedia Systems Conference and the 14th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Sept 2016, pp. 1–8.

[20] V. Roca, M. Watson, and A. C. Begen, "Forward Error Correction (FEC) Framework," RFC 6363, Oct. 2011. [Online]. Available: https://rfc-editor.org/rfc/rfc6363.txt

[21] S. Kopparty, S. V. Krishnamurthy, M. Faloutsos, and S. K. Tripathi, "Split tcp for mobile ad hoc networks," in *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, vol. 1, Nov 2002, pp. 138–142 vol.1.

[22] M. Schläger, B. Rathke, S. Bodenstein, and A. Wolisz, "Advocating a remote socket architecture for internet access using wireless lans," *Mobile Networks and Applications*, vol. 6, no. 1, pp. 23–42, 2001. [Online]. Available: http://dx.doi.org/10.1023/A:1009857619490

[23] O. Bonaventure, Q. D. Coninck, M. Baerts, F. Duchłne, and B. Hesmans, "Improving Multipath TCP Backup Subflows," IETF, Individual Submission, Internet Draft draft-bonaventure-mptcp-backup-00, Jul. 2015. [Online]. Available: https://tools.ietf.org/id/draft-bonaventure-mptcp-backup-00.txt

ACKNOWLEDGMENT

We thank Ceilidh Hoffmann, Jae H. Kim, and Jorge Mena for valuable discussions that influenced the presented design.