nmop
Internet-Draft
Intended status: Standards Track
Expires: 3 June 2026

Z. Xing
X. Di
H. Qi
Changchun University of Science and Technology

30 November 2025

The SDN-based MPTCP-aware and MPQUIC-aware Transmission Control Model draft-xing-nmop-sdn-controller-aware-mptcp-mpquic-04

Abstract

This document aims to study and implement Multipath Transmission Control Protocol (MPTCP) and Multipath QUIC (MPQUIC) using application layer traffic optimization (ALTO) or Link Layer Discovery Protocol (LLDP) in Software-Defined Networking (SDN). In a traditional Software-Defined Networking, ALTO server collects network cost indicators (including link delay, number of paths, availability, network traffic, bandwidth and packet loss rate etc.), and the controller extracts MPTCP or MPQUIC packet header to allocate MPTCP or MPQUIC packet to suitable transmission path according to the network cost indicators by ALTO and YANG topology modules, which can reduce the probability of transmission path congestion and improving path utilization in a multipath transmission network.

Table of Contents

1. Introduction 2
2. Terminology 3
3. Original transmission control mode of MPTCP or MPQUIC in SDN 3
4. Delivering functions by ALTO 4
5. Architectural Framework 4
6. Implementation and Deployment 7
7. Security Considerations 11
8. IANA Considerations 11
9. Discussion 11
10. Acknowledgments 11
11. References 11
11.1. Normative References 11
11.2. Informative References 12
Authors' Addresses 14

1. Introduction

The conventional TCP protocol only uses one path between a server and a client to exchange data. In order to realize the simultaneous transmission of data via multiple paths between a server and a client, the IETF standardized MPTCP [RFC8684] . MPTCP uses multiple paths between hosts to transmit data at the same time, but it is necessary to modify the operating system kernel to change the protocol stack of both parties or use an application proxy [RFC8803] in order to increase the MPTCP protocol. MPTCP has disadvantages such as difficulty in deployment. In order to solve the drawbacks in the transmission network and adapt to the faster development of the Internet, Google proposed the HTTP/3 protocol which is QUIC [RFC9000]. QUIC has many new features, such as: 0-RTT, forward error correction, connection migration, flexible congestion control, multiplexing without head-of-line blocking, and more. MPQUIC [MPQUIC] is a multi-path transmission protocol designed on the basis of QUIC. SDN [RFC7149] is a new network innovation architecture. By separating control and forwarding, it breaks the closedness of traditional network equipment, and uses programming to make network management more concise and programmability. ALTO [RFC7285] can

obtain and expose global network information to a controller, such as network traffic, link delay, etc. MPTCP and MPQUIC have their own characteristics [MultipathTester].

Realize the coupling control of MPTCP and MPQUIC subflows in the context of SDN, and obtain the network state information and allocate the optimal path according to the information conveyed in the ALTO Protocol, so as to improve the bandwidth utilization and resource allocation fairness, effectively alleviate the network congestion and realize the load balance between paths.

At present, some scholars have studied the model of deploying MPTCP or MPQUIC in Software-Defined Networking, [QUICSDN] \ [SDN_for_MPTCP] \ [SDN_MPTCP], but their SDN controller cannot manage the headers of MPTCP and MPQUIC data packets at the same time, and cannot manage of MPTCP and MPQUIC links at the same time.The ALTO Protocol can easily obtain various network states (including multiple SDNs, dynamic networks) from controller without the internal details of the network provider, and deliver controller decisions [SDN_ALTO_proof] \ [SDN_ALTO], which is already a successful solution.

The purpose of this document is to:

Describe the model that an SDN controller can use to MPTCP or MPQUIC data packets in the Software-Defined Networking.

According to the global information obtained by the AlTO, the controller allocates MPTCP or MPQUIC data packets with efficient transmission path.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Original transmission control mode of MPTCP or MPQUIC in SDN

In a Software-Defined Networking, the original controller cannot extract MPTCP or MPQUIC data packets. If MPTCP or MPQUIC as a server/client is deployed in SDN with a original controller and there are multiple transmission paths, the controller only selects one of the paths to exchange data, and the other paths are "idle" (i.e., there is only one path to transmit data). The utilization rate is low, and it is impossible to transmit data on multiple paths at the same time, resulting in low transmission efficiency.

4. Delivering functions by ALTO

An ALTO server is used to obtain network status information, and an SDN controller is considered as an ALTO client. The ALTO server collects network cost indicators (including link delay, number of paths, availability, network traffic, bandwidth and packet loss rate).

5. Architectural Framework

The architectural framework of multi-path transmission model based on SDN controller MPTCP and MPQUIC is shown in Figure 1.

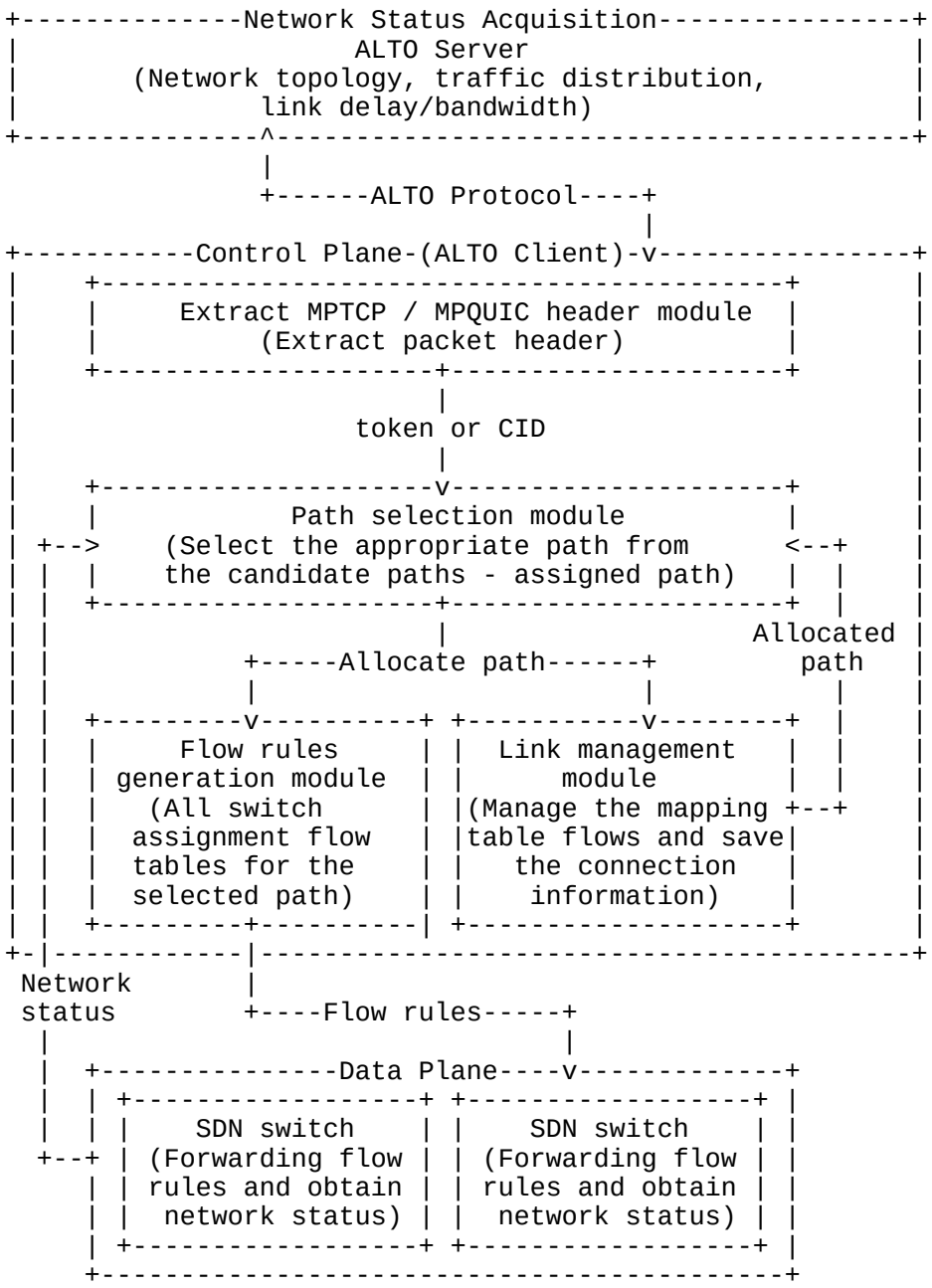


Figure 1: Schematic diagram of SDN-based MPTCP-aware and MPQUIC-aware transmission control model

The SDN-based MPTCP and MPQUIC transmission control model consists of three parts.

* The first part is the network status acquisition module, which acquires basic network status information from ALTO.

* The second part is the control plane, that is the SDN controller, also the client of ALTO, which includes extracting MPTCP / MPQUIC header module, path selection module, flow rules generation module and link management module. The main function is to extract the header identifier token of MPTCP (or CID of MPQUIC) according to the data packet (token from MPTCP unencrypted header and CID from MPQUIC unencrypted header, see Figure 2 and Figure 3 for details), obtain the global information of the whole network according to ALTO and allocate suitable paths and put flow rules to switches according to the global information of the entire network, and manage the links of the entire network at the same time.

* The third part is the data plane which is some OpenFlow switches. It executes the flow rules issued by the controller and realizes the forwarding of data packets.

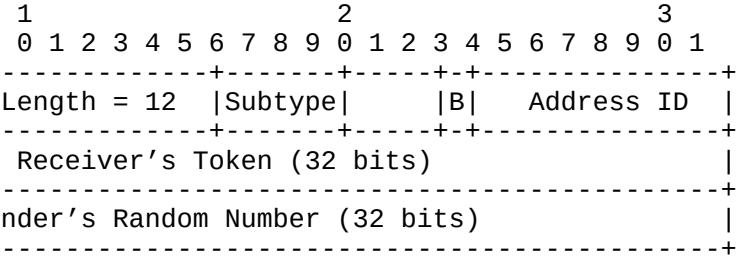
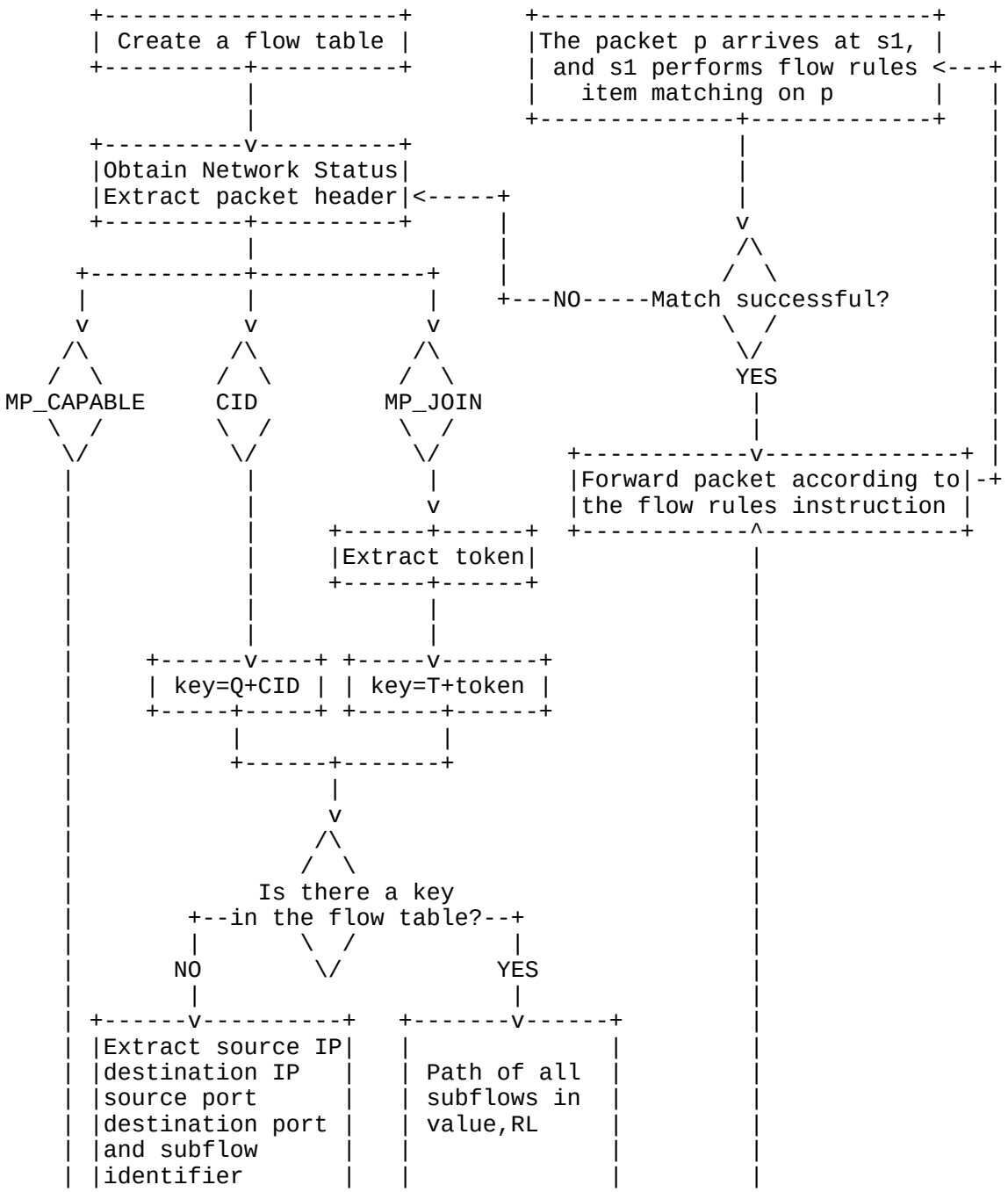


Figure 2: MPTCP Header Packet Format

```
 Long Header Packet {
  Header Form (1) = 1,
  Fixed Bit (1) = 1,
  Long Packet Type (2),
  Type-Specific Bits (4),
  Version (32),
  Destination Connection ID Length (8),
  Destination Connection ID (0..160),
  Source Connection ID Length (8),
  Source Connection ID (0..160),
  Type-Specific Payload (..),
 }
```

Figure 3: QUIC Header Packet Format

6. Implementation and Deployment



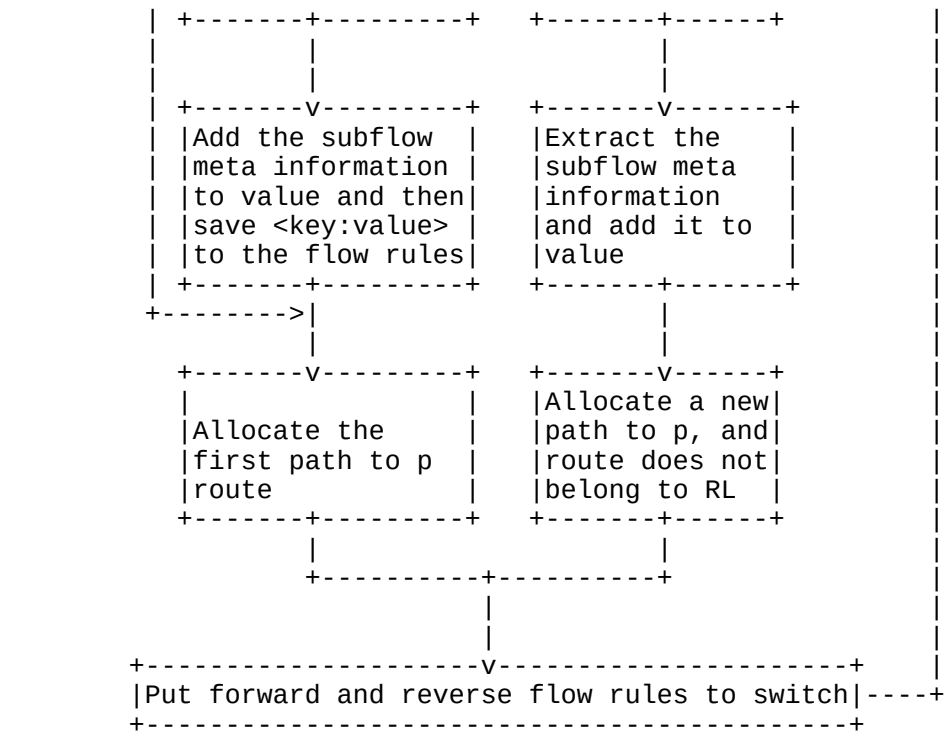


Figure 4: The flow chart of the SDN-based MPTCP-aware and MPQUIC-aware multi-path transmission control model

The flow chart of the SDN-based MPTCP-aware and MPQUIC-aware multi-path transmission control model is shown in Figure 4. The transmission control model is realized by the following steps:

* Step 1. The ALTO server collects network cost indicators (including link delay, number of paths, availability, network traffic, bandwidth and packet loss rate, etc.), which are recorded as: G (V, e), network topology g, V is the vertex and E is the edge.

* Step 2. The connection ID(CID) is the connection identifier of MPQUIC, and can be obtained from the QUIC header (connection ID, see Figure 3 for details). The SDN controller creates a mapping table flows for storing MPTCP or MPQUIC connection information, and each entry structure of the mapping table flows is <key:value>; wherein key is the unique identifier of MPTCP or MPQUIC connection, When the packet comes from MPTCP, key=T+token; and when the packet comes from MPQUIC, key=Q+CID (The letters T and Q are used to distinguish MPTCP and MPQUIC). value is a set of sub-stream meta-information, each item in the set is a sub-stream

meta-information; each sub-stream meta-information consists of source IP, destination IP, source port, destination port, MPTCP (or MPQUIC) sub-stream identifier and the path route composition.

* Step 3. When the data packet p from the MPTCP or MPQUIC server reaches the first switch s1, the first switch s1 extracts the header field of the data packet p, extracts the source IP, source port, destination IP and the destination port matches the source IP, source port, destination IP and destination port of the flow table in the first switch s1 respectively, and judges whether the matching is successful. If so, go to step 13; if not, then the first switch s1 encapsulates the data packet p and forwards it to the SDN controller, and at the same time adds the data packet p to the waiting queue.

* Step 4. After receiving the data packet p, the SDN controller extracts the header field of the data packet p, extracts the connection identifier of the data packet, and generates a key value, where when the data packet comes from MPTCP, key=T+token; When the packet comes from MPQUIC, key=Q+CID. Then query whether there is a key in the mapping table flows, if so, go to step 8, if not, go to step 5.

* Step 5. Extract the source IP, destination IP, source port, and destination port of the data packet p and generate a key value, where when the data packet comes from MPTCP, key=T+token; and when the data packet comes from MPQUIC, key=Q+CID.

* Step 6. ALTO to get basic network information. The controller calculates the threshold T according to the global network state information (network topology, number of switches, etc.). Using the depth-first traversal algorithm, find the available path set R={r_1,...,r_i,...,r_m } from all source nodes whose length does not exceed a certain threshold T to the destination node, r_i is the i available path, in the available path set Select a shortest path r_i in R as the path route of the sub-flow, where r_i=<s_(i,1),...,s_(i,j),...>, s_(i,j) represents the i available path The switch numbered j, where i belong to [1,m],j belong to [1,T].

* Step 7. Use the MPTCP and MPQUIC connection identifiers as the unique identifier key of the MPTCP and MPQUIC connections, where the key is the unique identifier of the MPTCP and MPQUIC connections. When the data packet comes from MPTCP, key=T+token; and the data packet comes from In MPQUIC, key=Q+CID. The source IP, source port, destination IP, destination port, MPTCP, MPQUIC sub-flow identifier and path route of the data packet p are added to the set value of sub-flow meta information as sub-flow meta-information, and then the <key:value> The form is saved to the mapping table flows, and go to step 11.

* Step 8. The SDN controller updates the flows table according to the global information of the network, and takes out the value from the connection identifier, and then composes all paths in the value into a set RL={r_1,r_2,...}.

* Step 9. The SDN controller searches for a suitable disjoint path for the data packet p according to the method in Step 5, and sets the found path as route=r_i, where r_i not belong to RL.

* Step 10. Extract the source IP, destination IP, source port, destination port, and MPTCP, MPQUIC sub-flow identifiers of the data packet p, and convert the source IP, source port, destination IP, destination port, MPTCP (or MPQUIC) sub-flow identifiers and the path route is added to the value as sub-flow meta information.

* Step 11. The SDN controller uses the source IP, source port, destination IP and destination port to issue the flow table to all switches in the route route, and set the route route=r_i=<s_(i,1),...,s_(i,j-1),s_(i.j),s_(i,j+1),...>, for the switch s_(i,j), the flow entry sent is the source IP, source port to the destination, the data packets of IP and destination port are forwarded to s_(i,j+1).

* Step 12. The controller sends the reverse flow table to all switches on the route route and sets the route route=r_i=<s_(i,1),...,s_(i,j-1),s_(i,j),s_(i,j+1),...>, for the switch s_(i,j) ,the flow table entry sent is to forward the data packets from the destination IP, destination port to source IP, and source port to s_(i,j-1).

* Step 13. The switch already contains a flow entry for processing the data packet p, and forwards the data packet according to the rules defined by the flow entry, and completes the processing of the data packet p. If an error occurs or execution fails, the timestamp, error code, source switch number and error switch number are sent to the ALTO server, which records them in the error.log file.

7. Security Considerations

The transmission control model uses the default security mechanism of SDN\ALTO\MPTCP\QUIC in the network, and does not modify the default security mechanisms such as encryption and authentication models [RFC7149], [RFC7285], [RFC6824] and [RFC9000].

8. IANA Considerations

TBD.

9. Discussion

The SDN transmission control model proposed in this document can simultaneously identify MPTCP and MPQUIC data packets and allocate optimal paths according to the network status obtained by ALTO, which expands the application scope of MPTCP and MPQUIC. In order to verify its comprehensive transmission performance, a fat-tree data center network is designed. The transmission control method proposed in this document improves the throughput by about 3 times compared to the default transmission control method [Survey] \ [Optimizing_multipath_QUIC]. This model can also be applied to satellite networks, marine networks, etc.

In future work, We will further research multipath transmission in Software Defined Information-centric networking [[RFC9236]] / [[RFC8793]] utilizing artificial intelligence technology to address multipath transmission issues in SDN.

10. Acknowledgments

The authors thank all reviewers for their comments.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <https://www.rfc-editor.org/info/rfc6824>.

[RFC7149] Boucadair, M. and C. Jacquenet, "Software-Defined Networking: A Perspective from within a Service Provider Environment", RFC 7149, DOI 10.17487/RFC7149, March 2014, <https://www.rfc-editor.org/info/rfc7149>.

[RFC7285] Alimi, R., Ed., Penno, R., Ed., Yang, Y., Ed., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, DOI 10.17487/RFC7285, September 2014, <https://www.rfc-editor.org/info/rfc7285>.

[RFC8684] Ford, A., Raiciu, C., Handley, M., Bonaventure, O., and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 8684, DOI 10.17487/RFC8684, March 2020, <https://www.rfc-editor.org/info/rfc8684>.

[RFC8793] Wissingh, B., Wood, C., Afanasyev, A., Zhang, L., Oran, D., and C. Tschudin, "Information-Centric Networking (ICN): Content-Centric Networking (CCNx) and Named Data Networking (NDN) Terminology", RFC 8793, DOI 10.17487/RFC8793, June 2020, <https://www.rfc-editor.org/info/rfc8793>.

[RFC8803] Bonaventure, O., Ed., Boucadair, M., Ed., Gundavelli, S., Seo, S., and B. Hesmans, "0-RTT TCP Convert Protocol", RFC 8803, DOI 10.17487/RFC8803, July 2020, <https://www.rfc-editor.org/info/rfc8803>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9236] Hong, J., You, T., and V. Kafle, "Architectural Considerations of Information-Centric Networking (ICN) Using a Name Resolution Service", RFC 9236, DOI 10.17487/RFC9236, April 2022, <https://www.rfc-editor.org/info/rfc9236>.

11.2. Informative References

[MPQUIC] "Multipath Extension for QUIC", <https://datatracker.ietf.org/doc/draft-ietf-quic-multipath/>.

[MultipathTester]
"Coninck Q D , Bonaventure O . MultipathTester: Comparing MPTCP and MPQUIC in Mobile Environments[C]// 2019 Network Traffic Measurement and Analysis Conference (TMA). 2019.", <https://10.23919/TMA.2019.8784653>.

[Optimizing_multipath_QUIC]
"Zeng H, Cui L, Tso F P, et al. Optimizing multipath QUIC transmission over heterogeneous paths[J]. Computer Networks, 2022, 215: 109198.", <https://doi.org/10.1016/j.comnet.2022.109198>.

[QUICSDN] "Kumar P , Chen J , Dezfouli B . QuicSDN: Transitioning from TCP to QUIC for Southbound Communication in SDNs[J]. 2021.", <https://ui.adsabs.harvard.edu/abs/2021arXiv210708336K>.

[SDN_ALTO] "V. K. Gurbani, M. Scharf, T. V. Lakshman, V. Hilt and E. Marocco, "Abstracting network state in Software Defined Networks (SDN) for rendezvous services," 2012 IEEE International Conference on Communications (ICC), 2012, pp. 6627-6632.", <https://doi.org/10.1109/ICC.2012.6364858>.

[SDN_ALTO_proof]
"Faigl, Z. , Z. Szabo , and R. Schulcz . "Application-layer traffic optimization in software-defined mobile networks: A proof-of-concept implementation." IEEE(2014):1-6.", <https://doi.org/10.1109/NETWKS.2014.6959200>.

[SDN_for_MPTCP]
"Hussein A , Elhajj I H , Chehab A , et al. SDN for MPTCP: An enhanced architecture for large data transfers in datacenters[C]// IEEE International Conference on Communications. IEEE, 2017.", <https://doi.org/10.1109/ICC.2017.7996653>.

[SDN_MPTCP]
"7. K. Gao, C. Xu, J. Qin, S. Yang, , et al. QoS-driven Path Selection for MPTCP: A Scalable SDN-assisted Approach, 2019 IEEE Wireless Communications and Networking Conference (WCNC)[C], 2019.", <https://doi.org/10.1109/WCNC.2019.8885585>.

[Survey] "Afzal S, Testoni V, Rothenberg C E, et al. A holistic survey of multipath wireless video streaming[J]. Journal of Network and Computer Applications, 2023, 212: 103581.", <https://doi.org/10.1016/j.jnca.2022.103581>.

Authors' Addresses

Ziyang Xing
Changchun University of Science and Technology
Changchun
Email: xzynet@gmail.com

Xiaoqiang Di
Changchun University of Science and Technology
Changchun
Email: dixiaoqiang@cust.edu.cn

Hui Qi
Changchun University of Science and Technology
Changchun
Email: qihui@cust.edu.cn