Cybertwin-aided Transmission for Cloud-Native Applications in Multi-homing Wireless Networks

Zhou Zou*, Mengyang Zhang*, Jiacheng Chen†and Haibo Zhou*
*School of Electronic Science and Engineering, Nanjing University, Nanjing, China, 210023.
†Department of Mathematics and Theories, Peng Cheng Laboratory, Shenzhen, China, 518000.
Email: zouzhou@smail.nju.edu.cn, Kolimn_Zhang@163.com, chenjch02@pcl.ac.cn, haibozhou@nju.edu.cn

2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring) | 979-8-3503-8741-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/VTC2024-SPRING62846.2024.10683080

***Abstract*—Multi-homed devices, designed to adapt to the heterogeneous wireless networks, have been available for users. However, practical challenges such as inadequate support from applications, hinder users from fully leveraging the multi-homing feature to enhance their service quality. Considering the undergoing evolution of applications towards cloud-native paradigm, we propose a Cybertwin-aided multi-application transmission scheme. It can fully reap the benefits of multi-homing transmission while mitigating the inherent limitations in traditional multi-path protocols. This is achieved through utilizing the so-called Cybertwin service as the communication agent for all cloud-native applications of the user. We multiplex different applications over the always-on multi-homing connections of Cybertwin service, leveraging the stream feature of Quick UDP Internet Connection (QUIC). We further support seamless stream migration between these connections to handle link fluctuation. We conduct extensive experiments under a test environment deployed in the real world and demonstrate the superiority of our scheme in realizing more efficient and stable transmission.**

***Index Terms*—Multi-homing wireless networks, Cybertwin, multi-path transmission, QUIC, Cloud-native**

I. Introduction

Multi-homed devices, which are equipped with the capability to connect to multiple network interfaces simultaneously, have been popular in the market for years. With the widespread adoption and advancement of various wireless communication technologies, such as 5G, Wi-Fi, and Direct-to-Cell satellite access, users equipped with such multi-homed devices are poised to benefit from the heterogeneous multi-homing wireless networks [1], including enhanced bandwidth aggregation and more stable communication quality, such that the increasing demand of high-bandwidth, low-latency, and reliable communication services can be satisfied. For example, multi-homing transmission was applied for video delivery in [2]. Further, [3] considered cognitive radio interface in multi-homing video transmission.

To efficiently utilize the multi-homing network, multi-path transmission protocols are required. However, current protocols such as Multi-path TCP (MPTCP) [4] and Multi-path Quick UDP Internet Connection (MPQUIC) [5] are not fully supported by various Internet applications, due to their intrinsic complexity and deficiency originated from packet management and packet blocking. On the other hand, the motivation for using these multi-path protocols is not strong for each application, since the bandwidth of a single path (e.g., 5G or Wi-Fi) is sufficient in most cases. However, from the perspective of an individual user, it is often the case that a bunch of applications are kept active simultaneously on the user's device, thus the total bandwidth demand can drain the capacity of a single path. Furthermore, single-path transmission inherently lacks stability, considering moving scenarios and interruptions of wireless links.

Meanwhile, the past decade has also witnessed a paradigm shift in the development, operation and deployment of applications. Conventional applications are tightly coupled with the underlying hardware and exhibit a severe degree of inflexibility, hindering updates and scaling. Consequently, cloud-native applications have emerged as a novel paradigm [6] tailored for cloud computing architectures. They fully leverage the infrastructures of major cloud providers and adopt the microservice architecture for better modularity and scalability. The microservices are further containerized and automatically managed by system-level software like Kubernetes during operation. Thus, it can be easily foreseen that data communication will become increasingly centralized to the cloud.

With the above considerations, in this paper, we introduce a novel Cybertwin-aided transmission scheme tailored for cloud-native applications. This approach fully leverages the multi-homing network for users, while avoiding the drawbacks of conventional multi-path protocols. Our core idea is to introduce a common communication agent service, namely Cybertwin (CT) [7], to carry out the transmission of all the user's cloud-native applications on the same cloud, meanwhile fully leveraging the device's multi-homing capability. The Cybertwin service comprises a client (`ct-client`) and a server (`ct-server`), running on the user's device and the cloud, respectively. It maintains an always-on multi-homed connection, and all the cloud-native applications are multiplexed over this connection.

Unfortunately, existing methods such as MPTCP and MPQUIC are not designed for multi-application transmission. To this end, we take advantage of the Quick UDP Internet Connection (QUIC) stream feature. Specifically, the independent data streams of applications are encapsulated in separate QUIC streams and transmitted over the same established connection of Cybertwin. In this way, these applications will not block each other even if some of them fail. Notably, compared with multi-path protocols, our scheme does not split the same data

979-8-3503-8741-4/24/$31.00 ©2024 IEEE

stream onto different paths, so as to avoid the complexity of packet management on multiple paths. Our scheme essentially decouples the end-to-end communication between the application's client (`app-client`) and server (`app-server`), and uses Cybertwin as an intermediate to facilitate the request-response procedures. The main contributions of this paper are summarized as follows:

- We propose a novel Cybertwin-aided transmission scheme tailored for cloud-native applications in multi-homing wireless networks. With Cybertwin acting as the communication agent, the benefits of multi-homing transmission can be achieved for users without introducing the complexities of conventional multi-path schemes. Application developers can also benefit from Cybertwin owing to the client-server decoupling.
- Leveraging the stream feature in QUIC, we realize the multiplexing of data streams from multiple applications over the multi-homed connection of Cybertwin service. Thus, the head-of-line blocking issues among different applications are circumvented. We further realize seamless migration of streams across different paths so as to adapt to link fluctuation. We present an implementation based on `quic-go`.
- We showcase the key features of our proposed Cybertwin-aided transmission scheme including content prefetching, multi-application transmission, and stream migration in a test environment deployed in the real world, demonstrating its feasibility and superiority.

The remainder of the paper is organized as follows. A brief review of related work is given in Section II. Section III delves into a detailed exposition of our motivation and design. Section IV articulates the architecture and the three-segment communication procedure of our scheme. Our implementation with `quic-go` is also presented in this section. Experimental results are showcased in Section V. Finally, the paper concludes in Section VI.

II. Related Works

QUIC [8], [9] is a transport layer protocol based on UDP. Its features are implemented in the user space, so it is compatible with existing network equipment and is further extensible [10]. QUIC develops the concept of stream to enable the multiplexing of different data streams of an application within a single connection, thus head-of-line blocking issues can be mitigated. Besides, its unique 0-RTT (Zero Round Trip Time) handshake mechanism is designed to reduce the connection and encryption negotiation time. According to the results in [11], [12], QUIC outperforms TCP significantly under poor network conditions. Numerous designs such as QFaaS [13] take advantage of QUIC to obtain better performance. In this paper, we utilize QUIC for establishing multi-homing connections of the Cybertwin service. With the aid of Cybertwin, applications without QUIC support yet can also gain the benefits of QUIC. We further take advantage of the QUIC stream feature to realize multiplexing of various applications and avoid blocking issues among them.

Multi-path transmission protocols are initially proposed to overcome network interruptions and aggregate bandwidth in multi-homing wireless networks. Recently, MPQUIC [5] was developed from the traditional MPTCP for QUIC. MPQUIC has been studied in various domains such as video streaming [14], [15] and satellite networks [16]. Both the benefits and complexities of multi-path transmission come from the following job: segmenting and dispatching data from a single application across various paths for transmission and reassembling data at the receiving end. Different packet schedulers aim to optimize performance in specific scenarios [17], [18], and machine learning methods are further used in [19], [20]. Additionally, energy-aware and cost-aware multi-homing video transmission is studied in [2], [3]. Unlike traditional multi-path packet schedulers, in our proposed scheme, scheduling is at the data stream or application level, since it is designed for transmission of multiple cloud-native applications. Thus, the complexity of packet-level schedulers is avoided, while the benefits of multi-path transmission can still be gained.

III. Motivation and Design

Our motivation for Cybertwin-aided transmission scheme is distinctive in that we consider *from the user's perspective* instead of the application's. Traditionally, each application sets up its own connection between client and server and is responsible for the data communication quality. However, on the user's device, all the applications actually transmit data using the same underlying physical path, whose capacity is limited. To this end, we seize the end-to-end connection from the cloud to the user's device, covering the total transmission distance and the multi-homing wireless environment. We then employ Cybertwin as the common communication agent to handle all the issues in this connection for all the cloud-native applications. Further, advanced techniques including QUIC and multi-homing transmission are exploited, thus improving the overall QoS of all applications. In fact, we find through real tests that a variety of major Internet applications have not even deployed QUIC. The connection of Cybertwin is established at the time of turning on the device and kept alive until the device turns off. Then, the handshake process for connection establishment for all applications can be saved. Also, compared with maintaining long connections for all applications, using a common connection relatively saves resources for the device.

Cybertwin also brings the *client-server decoupling* paradigm for development and operation of cloud-native applications. Developers no longer have to consider data communication related issues, e.g., on achieving reliable and high-quality in wireless environments. Instead, they simply need to communicate with the `ct-server` using the `app-server` on the same cloud and communicate with the `ct-client` using the `app-client` on the same device. These two types of simple communications can even happen independently, instead of in the same request-response procedure. Therefore, developers can focus on designing the applications' core functions.

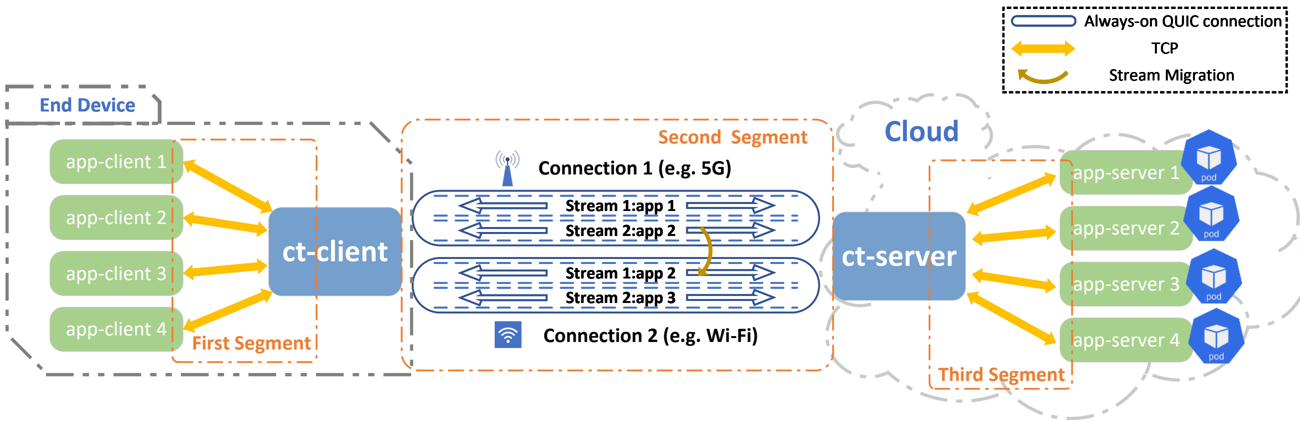


Fig. 1. The Cybertwin-aided communication architecture.

We further elaborate on our novel design and its advantages. Generally, the advantages of multi-homing transmission are from two aspects: reliability and bandwidth aggregation. Since existing multi-path schemes all consider from the application's perspective, to utilize multiple paths, different packets of the application have to be separated at the source and assembled at the destination in sequence [15], [17], which significantly adds complexity to the protocol and design of multi-path scheduler. Furthermore, delay or failure of the packets on any path will lead to unsuccessful transmission. Moreover, with advanced wireless communications technologies such as 5G and Wi-Fi 7, the capability of a single physical path is enough for most applications. On the contrary, in our proposed scheme, the data from all cloud-native applications are multiplexed on the same multi-homing connections between `ct-client` and `ct-server`. Then, if we simply schedule the independent data streams of all applications onto different paths, we can still achieve the same advantages of multi-homing transmission, while avoiding the above packet management issues. These independent data streams will not affect each other if any of them fails, and each will be transmitted using one QUIC stream without being splitted. For example, one stream used for video transmission and another stream used for displaying the comments on the video are independent streams for a video streaming application. Cybertwin can also help avoid the "tragedy of the commons" problem on the user's device, where each application would greedily utilize the better link for its own transmission if there does not exist an application-level scheduler (i.e., the Cybertwin) for the user, leading to waste of resources and degradation of user's experience.

IV. Cybertwin-Aided Transmission Scheme

A. Architecture Description

As illustrated in Fig. 1, the traditional end-to-end communication between `app-client` and `app-server` is split into three segments by the intermediate Cybertwin service consisting of `ct-client` and `ct-server`:

- First Segment: It happens between all `app-clients` and the `ct-client` on the device, thus it is extremely efficient. Simple TCP connections can be used.
- Second Segment: It happens between `ct-client` on the device and `ct-server` on the cloud, as a traditional end-to-end connection between the client and server. Communications over multi-homing wireless interfaces such as 5G and Wi-Fi take place in this segment. The transmission distance from the device to the cloud can be relatively long and the involved wireless transmission can be volatile and unstable [21].
- Third Segment: It happens between the `ct-server` and various `app-servers` on the same cloud, considering the cloud-native deployment of applications. Similar to the First Segment, the communication is highly efficient and basic TCP connections can be used.

To overcome the challenges in the Second Segment, we extend the QUIC protocol. Multiple QUIC connections with independent connection IDs will be established over the heterogeneous physical paths of the multi-homing wireless network. Data streams of these applications are multiplexed onto the common connections on the Second Segment using QUIC streams, such that they do not affect each other. Furthermore, as shown in Fig. 1, `ct-client` and `ct-server` are like the gateways for all cloud-native applications, thus the QUIC connection can be always-on to carry all the applications' traffic, saving both the connection establishment overheads and the resource consumption for maintaining separate connections for various applications.

The general request-response process for an application in Cybertwin-aided transmission is described below. In the First Segment, the `ct-client` is responsible for receiving requests from the local `app-client`. Then, the path scheduler of Cybertwin selects an appropriate connection from the established ones to forward the request to the `ct-server` on the Second Segment. The decision of path selection considers real-time network conditions and the user's priority on the application. Finally, the request received by the `ct-server` is delivered to the corresponding app-server on the Third Segment. The response data from app-server is delivered back to app-client using the same path. Through the above interactions, we can find that the Cybertwin service does not behave like a proxy that is transparent to other applications. ct-client and ct-server are responsible for both receiving and forwarding data. The request-response is actually carried out by three request-response stages on the three segments. In this way, the end-to-end client-server connection is decoupled.

With client-server decoupling, it is straightforward to realize data prefetching, which requests data for the user in advance and caches them on the device, such that the user can acquire the data with minimum waiting time. At a certain time, ct-server sends the prefetching request to app-server, which then returns the prefetched data back to ct-server. The data is further pushed to ct-client and cached on the device. To retrieve the prefetched data, app-client just sends a prefetched data request to ct-client once it is launched.

Cybertwin-aided transmission also brings unprecedented flexibility for managing streams across different connections and devices. A feature named stream migration (SM) is realized, which seamlessly migrates one stream from one QUIC connection to another, without requiring a new connection establishment process. Obviously, it is useful when the current connection encounters interruptions or high utilization. SM can further take place between the QUIC connections at user's different devices, so as to realize multi-device collaboration for the applications. Note that SM in our proposed scheme can totally replace the original connection migration feature of QUIC since the data streams on the lost connection can be migrated to another existing connection seamlessly.

B. Implementation

In this subsection, our quic-go[1]-based implementation of the above key features of Cybertwin-aided transmission is introduced. To keep alive QUIC connections between ct-client and ct-server, we utilize the parameters MaxIdleTimeout and KeepAlivePeriod. To maintain the state of the QUIC connection and ensure its availability, KeepAlivePeriod is set to $1/4$ of MaxIdleTimeout. Then, the keep-alive packets will be sent at regular intervals to keep the connection active and prevent it from being prematurely terminated due to inactivity. As the QUIC protocol is built on top of the UDP protocol, we specify the underlying udpConnection of the QUIC connection with the IP address of the corresponding network interface, so as to establish different QUIC connections over the corresponding physical paths. We modify the APIs of original quic-go to support this function.

To achieve multiplexing of applications over QUIC connections, we implement a dedicated HTTP/3 client and server as ct-client and ct-server for the Cybertwin service. The multiple QUIC connections are established with the above configurations once the Cybertwin service is launched, so as to be prepared for future data transmission of other applications. To realize the request-response on the three segments, ct-client also acts as a server for app-client and forwards the original request to ct-server using a new QUIC stream on the selected QUIC connection specified by the path scheduler. Similarly, the response will be forwarded back with the QUIC stream.

Stream migration can be implemented with the help of the target QUIC connection. When an SM decision is made, ct-client first checks the data that has already been transmitted through the localSize parameter. It then sends a request over the target QUIC connection to the ct-server, which can continue transmitting data in the range of (localSize, fizeSize].

[1]https://github.com/quic-go/quic-go

V. Evaluation

A. Experiment Setup

We conduct experiments in a test environment deployed in real world. We use one laptop as the device with multi-homing capability to establish connections through multiple network interfaces. Additionally, we utilize commercial cloud services from a major cloud operator to run both ct-server and app-servers. The distance between the device and the cloud is about 1175 km. The total outbound bandwidth from the cloud is 5 Mbps, and the tc tool is utilized to control the bandwidth and pack loss rate (PLR) of each path.

We test the HTTP request-response procedures. The volume of response data from different applications is set to 10 MB and 2.5 MB, respectively. Our proposed Cybertwin-aided transmission scheme will be compared against traditional transmission method without Cybertwin. As explained earlier in this paper, QUIC and multi-path transmission will not be used in the traditional method. We use the completion time of response data as the main metric for evaluation. For each experiment, we repeat multiple times to average out the influence of network fluctuation in real world.

B. Results and Discussions

1) Cybertwin-aided transmission and prefetching: We start by validating the feasibility of Cybertwin-aided transmission scheme using only one connection. As we can see from Fig. 2, if we do not consider the handshake time, the RTT of three-segment communication in our proposed scheme is only slightly larger than traditional end-to-end communication. Particularly, we can find that the Second Segment accounts for the majority of total RTT, due to the long transmission distance over the Internet. Therefore, the incurred overheads of three-segment communication resulting from the intermediate Cybertwin service are trivial. On the other hand, when the handshake time is taken into account, our scheme performs much better. The connection on the Second Segment of our proposed scheme is already established, so the time of handshakes over long transmission distances can be saved.

Specifically, the time cost of initial handshake in Cybertwin-aided 3-segment transmission scheme is HTTP + QUIC (0-RTT) + HTTP $\approx$ 1 RTT, while it is about 3 RTTs in HTTPS (HTTP/2 + TLS 1.2). Note that the two extra handshake procedures take place in local devices and clouds, where communication is highly efficient.

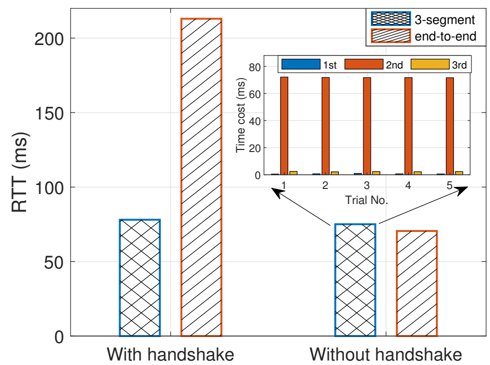
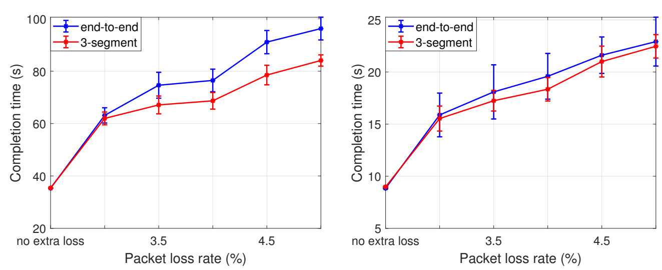


Fig. 2. RTT comparison between CT scheme and traditional end-to-end communication over HTTPS.

We also evaluate the performance of our proposed scheme with different sizes of response data under different network conditions. To be more specific, the PLR is artificially increased in different experiments. From Fig. 3, it can be observed that the completion time is nearly identical to that of traditional end-to-end communication when there is no extra loss, and the gap in the completion time becomes more substantial when the PLR increases. This is mainly owing to the utilization of QUIC, which is utilized in the Second Segment for all cloud-native applications.



(a) 10MB (b) 2.5MB

Fig. 3. Completion time comparison between CT scheme and end-to-end (HTTPS) communication.

Last, we demonstrate the advantage of prefetching. With the help of Cybertwin service, data can be prefetched to the device and cached at `ct-client` before the `app-client` is launched. As shown in Fig. 4, the time for fetching data is then largely reduced, which can strongly affect users' experience of using the application.

2) Multi-homing transmission for multiple applications: We will show the advantages of multi-homing transmission for multiple cloud-native applications in Cybertwin-aided transmission scheme from two aspects. In the first scenario, two paths are available, each offering a bandwidth of 2.5 Mbps with different PLRs. On the device, there are two applications requesting data of 10M and 2.5M, respectively. Under our proposed scheme, with the assistance of Cybertwin, the device is able to fully utilize the aggregated bandwidth of the two paths. Specifically, each path is used for transmitting data of one application, so we do not need packet splitting to achieve the benefit of bandwidth aggregation as in traditional multi-path transmission. From the user's perspective, this results in a reduced overall data transmission time for both applications as shown in Fig. 5(a), where the total duration is determined by the transmission completion time of the application with the larger response data volume. Furthermore, as observed in Fig. 5(b), with the our proposed scheme, not only the completion time is reduced, but also the stability of transmission is enhanced, particularly in scenarios characterized by relatively high packet loss rates. On the contrary, in traditional transmission method without the assistance of Cybertwin, despite the multi-homing capability of the device, only a single path can be used for the transmission of data from both applications. This leads to an increase in overall transmission time, owing to the underutilization of network resources from both paths.

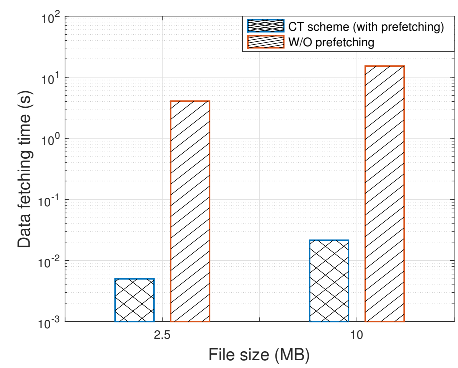
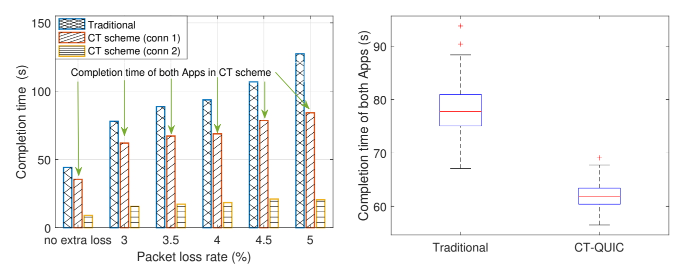


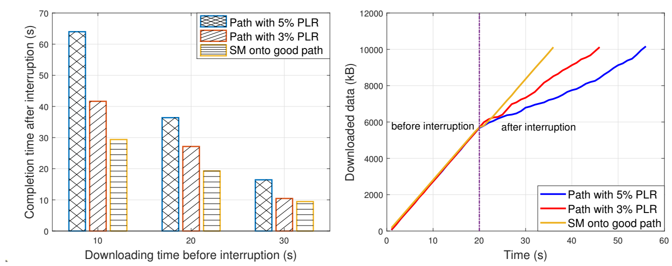
Fig. 4. Data fetching time comparison.



(a) Various packet loss rate (b) 3% packet loss rate

Fig. 5. Transmission performance of CT scheme in multi-homing network.

In the second scenario, we consider an application requesting data with a size of 10 MB using a certain path. After a moment, the path will encounter interruptions, leading to a significant increase in PLR. In Cybertwin-aided scheme, network performance measurement can be easily achieved by existing methods, since the other path can be utilized for synchronizing the condition of the current path between `ct-client` and `ct-server`. Also, the measurement cost is acceptable because we only need to measure two paths for all applications. Fig. 6(a) shows the remaining time for completing the data after interruption occurs at different time points. In Cybertwin-aided scheme, interruptions have minimal impact on the data transmission process since the data stream is seamlessly migrated to another good path. However, without the help of Cybertwin, the data continues transmitting on the original path, so the time costs are much higher. From Fig. 6(b), we can also clearly see that the difference in data transmission speed after interruption is evident and noticeable.



(a) Completion time after interruption (b) Downloaded data with time

Fig. 6. Stream migration performance of CT scheme.

VI. Conclusion and Future Works

In this paper, we have proposed a Cybertwin-aided transmission scheme for cloud-native applications to optimize the use of heterogeneous paths in multi-homing wireless networks. Utilizing the QUIC protocol, we design and implement our proposed scheme with the novel multi-application multiplexing and stream migration mechanisms, such that the benefits of multi-homing transmission are gained, while the complexities of traditional multi-path schemes are avoided. We have verified the feasibility and superiority of Cybertwin-aided transmission scheme through experimental results. Moving forward, our research will further demonstrate and leverage the Cybertwin service for more complex and diverse data streams. Besides, we will develop a QoE-aware path scheduler, so as to realize personalized data transmission of various applications for users.

Acknowledgment

This work was supported in part by the National Natural Science Foundation Original Exploration Project of China under Grant 62250004, the Natural Science Foundation of China (NSFC) under Grant 62271244, Innovation and Entrepreneurship of Jiangsu Province High-level Talent Program, Summit of the Six Top Talents Program of Jiangsu Province, The Major Key Project of PCL, and The Basic and Frontier Research Project of PCL.

References

[1] K. Habak, K. A. Harras, and M. Youssef, "Bandwidth aggregation techniques in heterogeneous multi-homed devices: A survey," *Computer Networks*, vol. 92, pp. 168–188, 2015.

[2] M. Ismail, W. Zhuang, and S. Elhedhli, "Energy and content aware multi-homing video transmission in heterogeneous networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 7, pp. 3600–3610, 2013.

[3] S. Agarwal and S. De, "Cognitive multihoming system for energy and cost aware video transmission," *IEEE Transactions on Cognitive Communications and Networking*, vol. 2, no. 3, pp. 316–329, 2016.

[4] C. Paasch and O. Bonaventure, "Multipath TCP," *Communications of the ACM*, vol. 57, no. 4, pp. 51–57, 2014.

[5] Q. De Coninck and O. Bonaventure, "Multipath QUIC: Design and evaluation," in *Proceedings of the 13th international conference on emerging networking experiments and technologies*, 2017, pp. 160–166.

[6] S. Deng, H. Zhao, B. Huang, C. Zhang, F. Chen, Y. Deng, J. Yin, S. Dustdar, and A. Y. Zomaya, "Cloud-native computing: A survey from the perspective of services," *arXiv preprint arXiv:2306.14402*, 2023.

[7] Q. Yu, J. Ren, Y. Fu, Y. Li, and W. Zhang, "Cybertwin: An origin of next generation network architecture," *IEEE Wireless Communications*, vol. 26, no. 6, pp. 111–117, 2019.

[8] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar *et al.*, "The QUIC transport protocol: Design and Internet-scale deployment," in *Proceedings of the conference of the ACM special interest group on data communication*, 2017, pp. 183–196.

[9] J. Iyengar, M. Thomson *et al.*, "QUIC: A UDP-based multiplexed and secure transport," in *RFC 9000*. Internet Engineering Task Force (IETF) Fremont, CA, USA, 2021.

[10] Q. De Coninck, F. Michel, M. Piraux, F. Rochet, T. Given-Wilson, A. Legay, O. Pereira, and O. Bonaventure, "Pluginizing QUIC," in *Proceedings of the ACM Special Interest Group on Data Communication*, 2019, pp. 59–74.

[11] P. Wang, C. Bianco, J. Riihijärvi, and M. Petrova, "Implementation and performance evaluation of the QUIC protocol in linux kernel," in *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2018, pp. 227–234.

[12] S. Cook, B. Mathieu, P. Truong, and I. Hamchaoui, "QUIC: Better for what and for whom?" in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[13] K. Hou, S. Lin, Y. Chen, and V. Yegneswaran, "Accelerate and secure serverless networks with QUIC," in *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*, 2021, pp. 477–478.

[14] W. Yang, J. Cao, and F. Wu, "Adaptive video streaming with scalable video coding using multipath QUIC," in *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. IEEE, 2021, pp. 1–7.

[15] Z. Zheng, Y. Ma, Y. Liu, F. Yang, Z. Li, Y. Zhang, J. Zhang, W. Shi, W. Chen, D. Li *et al.*, "Xlink: QoE-driven multi-path QUIC transport in large-scale video services," in *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, 2021, pp. 418–432.

[16] W. Yang, S. Shu, L. Cai, and J. Pan, "MM-QUIC: Mobility-aware multipath QUIC for satellite networks," in *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*. IEEE, 2021, pp. 608–615.

[17] X. Zuo, Y. Cui, X. Wang, and J. Yang, "Deadline-aware multipath transmission for streaming blocks," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 2178–2187.

[18] A. Rabitsch, P. Hurtig, and A. Brunstrom, "A stream-aware multipath QUIC scheduler for heterogeneous paths," in *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, 2018, pp. 29–35.

[19] M. M. Roselló, "Multi-path scheduling with deep reinforcement learning," in *2019 European Conference on Networks and Communications (EuCNC)*. IEEE, 2019, pp. 400–405.

[20] B. Liao, G. Zhang, Z. Diao, and G. Xie, "Precise and adaptable: Leveraging deep reinforcement learning for gap-based multipath scheduler," in *2020 IFIP Networking Conference (Networking)*. IEEE, 2020, pp. 154–162.

[21] J. F. A. Rida, "Overview of development performance for mobile phone wireless communication networks," in *2021 International Conference on Electrical, Computer and Energy Technologies (ICECET)*. IEEE, 2021, pp. 1–11.