2ND WORKSHOP "MACHINE LEARNING & NETWORKING" (MaLeNe) PROCEEDINGS

SEPTEMBER 4, 2023

CO-LOCATED WITH
THE 5TH INTERNATIONAL CONFERENCE ON NETWORKED SYSTEMS (NETSYS 2023)
POTSDAM, GERMANY

Steps Toward a Supervised Machine Learning Scheduler for MPTCP

Reza Poorzare, Hadi Asghari, and Oliver P. Waldhorst
Data-Centric Software Systems Research Group at the Institute of Applied Research,
Karlsruhe University of Applied Science, Karlsruhe, Germany
reza.poorzare@h-ka.de, asha1011@h-ka.de, and oliver.waldhorst@h-ka.de

***Abstract*— The functionality of the MPTCP scheduler is a hurdle in the way of the protocol in achieving high performance. This drawback is even more severe in heterogeneous networks, where the differences in the characteristics of the paths impair the functionality of the scheduler drastically. In this paper, we introduce a dataset generated by an emulation environment, including diverse scenarios and traffic types, as an initial step toward having a supervised learning scheduler.**

Keywords—Transport protocols, Machine Learning, MPTCP, Proxy, OpenStack

I. Introduction

MPTCP (Multipath TCP), which is an extension of conventional TCP (Transmission Control Protocol), was developed to allow having more than one network path in the very same connection. This protocol can indeed provide some benefits relying on its scheduling mechanisms and simultaneous distribution of traffic into different paths. However, it cannot achieve full bandwidth aggregation, due to flaws such as out-of-order packets, frequent re-ordering processes, or HoL (Head of Line) blocking that waste time and energy. Most schedulers exploit single-criterion or multi-criteria approaches for traffic management. In the former, a parameter such as RTT (Round Trip Time) is used, while in the latter, more than one parameter is considered to select the path for the transmission. These reactive approaches lack the ability to properly distribute traffic to prevent the above-mentioned issues and are easily confused by random packet losses or other shortcomings that may occur in the network [1]. Therefore, to overcome these issues and make the best use of the available bandwidth aggregation, it is necessary to design schedulers based on machine learning techniques that can not only detect the current state of the subflows, i.e., paths, but also predict upcoming situations to enhance the functionality of MPTCP.

By considering these facts, the main questions in this ongoing work are: *(i)* How to create an emulation environment that can reflect various real-time networking circumstances? *(ii)* How to have a centralized node in the topology that has insight into the whole traffic to generalize the data set? *(iii)* Based on the generated data set, how can appropriate features be selected to be used in supervised learning techniques such as deep neural networks?

II. Related work

Several approaches have been proposed to deal with the inefficient functionality of the MPTCP scheduler. The initial steps approached existing problems, such as out-of-order delivery, reactively, leading to schedulers like BLEST (BLock ESTimation) [2] and ECF (Earliest Completion First) [3]. BLEST estimates the blocking time for different subflow selections and traffic distributions in a way that alleviates out-of-order and HoL blocking issues. With some similarities to BLEST, ECF attempts to find the path with the minimum transmission delay by exploiting parameters such as RTT and cwnd (congestion window).

There have been some other proposals, but most of them suffer from having static and non-intelligent methods. Thus, they are not able to fully utilize the available resources. A reinforcement learning scheduler called MPTCP-RL has been proposed recently in [4] to find the best optimal path and mitigate packet loss and network heterogeneity adverse impacts. This scheduler tries to create a table containing scheduling rules for subflow selection, and by relying on the rules, it could enhance the network's throughput. However, this approach can waste some time in the decision-making phase, since it should update itself frequently. Moreover, it cannot be generalized easily for different scenarios. As a result, there is a need for supervised learning techniques so that the training and decision-making parts can be separated. In this case, a machine learning engine that is frequently updated offline resides in the scheduling component. This mechanism can dramatically reduce the time spent on the decision-making process. However, to the best of our knowledge, there is no public dataset for this, so the first steps should be taken toward its creation.

III. MPTCP Proxy Deployment for Data Set Creation

The main problem in a supervised learning scheduler establishment is the lack of a stereotyped data set. A data set should be a reflection of diverse real-world scenarios, so it can generalize to most of the existing situations. As a result, we have divided the existing scenarios into three different categories, including short- medium- and long connections. For the first scenario, web page loading, for the second one, video streaming, and for the third one, test file download were the representatives. After identifying the problems and counterpart representatives for the scenarios, an approach for data gathering should be selected. As a result, we decided to use an intermediate node as a MPTCP proxy so that all traffic between clients and public servers goes through it and it can monitor all the traffic.

This approach can bring some advantages, including: *(i)* Public servers do not need to support MPTCP as the connections between the clients and proxy will be MPTCP ones. *(ii)* As the whole traffic is passed through a centralized node, the creation of a data set by using it can be a reflection of the network. On the proxy server, we have used microsock5, which is a lightweight proxy that handles traffic redirection without heavy use of resources.

IV. EMULATION SETUP AND METHODOLOGY

The laboratory experiment conducted in this study utilized the OpenStack platform as the virtualization infrastructure, including thirty Linux clients, as shown in Figure 1. Moreover, a Python automation script was employed for streamlined deployment. By using a Linux traffic shaper tool on individual subflows, characteristics such as packet loss probability, latency, or bandwidth, for 5G (Fifth Generation) and Wi-Fi 6 networks were emulated. The experiment encompassed various scenarios, including data download, video streaming, and webpage loading, to imitate real-world requirements and have different traffic types.

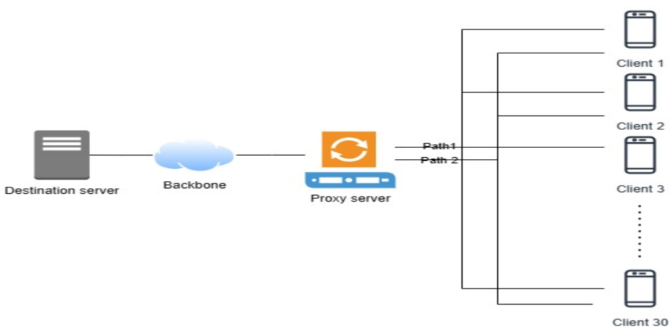

Figure 1 The emulation environment in OpenStack

Table I Selected parameters to create the data set

Parameter	Description
srtt_us	Round trip time for subflows
basertt	The minimum seen RTT of individual subflows
snd_cwnd	Sending congestion window
max_window	The maximal window ever seen from a subflow
bytes_acked	How many bytes were acked
bytes_sent	Total number of data bytes sent
prior_cwnd	The cwnd right before loss recovery
lost	Total data packets lost, including retransmissions

Once we have the emulation environment, we should choose the parameters carefully. This process should follow two main goals: 1. The parameters should be able to help the trained scheduler distinguish different states in the network 2. They should provide the feasibility of differentiating various circumstances in the network, such as shadowing or fading. With these goals in mind, the parameters in Table I were extracted from the tcp.h file. However, parameters can be added or deleted as needed. In the feature selection phase, some important criteria should be considered, including 1. When the network state is changed, it should be reflected. 2. Important states and conditions such as peak data rates, low latencies, recovery times, traffic load portion, and reasons for the packet loss should be distinguishable. In the next steps, a selection of variables can form the inputs of a supervised learning technique, such as a deep neural network, to train the engine. These variables should be affected by changes in the network to reflect the state of the network, and preferably reside between zero and one to avoid normalization. Some selective examples are given in Table II. For the final step and labeling of the outputs, random hashing is used to weigh the subflows and find out which weights appropriately reflect the states based on fair use of the bandwidths and user experience measurements such as packet loss and data rate.

Table II Selective inputs to feed the deep neural network

Variable	Goal
basertt/srtt_us	Traffic load detection
snd_cwnd /max_window	Determine the aggressiveness of the sending rate adjustment
bytes_acked /bytes_sent	Estimation of the BDP (Bandwidth-Delay Product)
prior_cwnd /max_window	Having a faster recovery
lost/bytes_sent	Distinguishing random losses

V. CONCLUSION AND FUTURE WORK

The MPTCP scheduler has some drawbacks in selecting the best possible subflow because of its static and non-intelligent mechanism. As a result, in this work, we took the first steps toward having a supervised machine learning-based scheduler. We have established an emulation environment reflecting different network conditions, and then, by using a MPTCP proxy node, a stereotyped data set was created that can be used in supervised learning approaches. In future work, we will feed this data set to a deep neural network to conceive of an intelligent scheduler that can function properly in various circumstances.

ACKNOWLEDGMENT

This work was supported by the bwNET2020+ project which is funded by the Ministry of Science, Research and the Arts Baden-Württemberg (MWK). The authors alone are responsible for the content of this paper.

References

[1] R. Poorzare and O. P. Waldhorst, "Toward the Implementation of MPTCP Over mmWave 5G and Beyond: Analysis, Challenges, and Solutions," *IEEE Access*, vol. 11, pp. 19534-19566, Feb. 2023, DOI: 10.1109/ACCESS.2023.3248953.

[2] S. Ferlin, Ö. Alay, O. Mehani, and R. Boreli, "BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks," *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, 2016, pp. 431-439, DOI: 10.1109/IFIPNetworking.2016.7497206.

[3] Y.-S. Lim, E. M. Nahum, D. Towley, and R. J. Gibbens, "ECF: An MPTCP path scheduler to manage heterogeneous paths," *in Proc. 13th Int. Conf. Emerging Networking Experiments and Technologies (CoNEXT)*, Incheon, Republic of Korea, Dec. 2017, pp. 147-159. DOI: 10.1145/3143361.3143376.

[4] P. Dong *et al.*, "Multipath TCP Meets Reinforcement Learning: A Novel Energy-Efficient Scheduling Approach in Heterogeneous Wireless Networks," *IEEE Wireless Communications*, vol. 30, no. 2, pp. 138-146, April 2023, doi: 10.1109/MWC.013.2100658.