PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

DJILLALI LIABES UNIVERSITY SIDI BEL ABBES

FACULTY OF ELECTRICAL ENGINEERING

Department of Electronics Engineering

THESIS

Presented to obtain the degree of DOCTORATE 3rd cycle

Speciality: Networks, Architecture and Multimedia

Presented by **Mohammed Bencheikh**

Theme

Optimization of load balancing algorithms for IP traffic

Jury:

President	BOUNOUA Abdennacer	Professor	University of Djillali Liabes - SBA
Director	BOUKELIF Aoued	Professor	University of Djillali Liabes - SBA
Co-Director	AZAIZ Ahmed	Professor	University of Djillali Liabes - SBA
Examiner	DJEBBAR Ahmed Bouzidi	Professor	University of Djillali Liabes - SBA
Examiner	KANDOUCI Chahinez	Associate Prof.	University of Djillali Liabes - SBA
Examiner	BENDAOUD Fayssal	Associate Prof.	Higher School of Computer Science-SBA

2023-2024

Acknowledgements

Praise be to the Almighty God who has given me faith, courage, and patience to carry out this work.

I want to express my deep gratitude to my supervisor Pr. Aoued Bouklif from Djillali Liabes University, for the confidence he has placed in me, through his presence always with me, by his direction, his modesty, his advice, and constructive remarks for the good progress of this work.

I thank Dr. Ahmed Azaiz from Djillali Liabes Universityfor having co-supervised me, for his orientation, availability, listening, and patience during the realization of this job.

I want to express my deep gratitude to Pr. Nasereddine Taleb from Djillali Liabes University, for the confidence he has placed in me, through his presence always with me, by his direction, his modesty, his advice, and constructive remarks for the good progress of this work.
I thank Dr. Chakib Mustapha Anouar Zouaoui from Djillali Liabes Universityfor having co-supervised me, for his orientation, availability, listening, and patience during the realization of this job.

I would like to thank everyone who helps me to improve my work. and who gave me any remark that helped me to perfect this manuscript.

I express my deep gratitude to my parents, my sisters, and my whole family for their encouragement and prayers that allowed me to achieve this modest job. I am very grateful for the confidence they have placed in me.

Finally, I express my gratitude to all those who have contributed in one way or another

to the development of this work.

O Allah, send your blessings on your noble messenger, his family, and companions, and bless us in our life.

Dedication

I dedicate this work to my parents:

May they find here the testimony of my deep gratitude and acknowledgment

To my sisters and my family who give love and liveliness.

To all those who have helped me - directly or indirectly - and those who shared with me the emotional moments during the accomplishment of this work and who warmly supported and encouraged throughout my journey.

Dedicated to my best friend, "John Paul Martin ", and to all my friends who have always encouraged me, to whom I wish more success.

Thanks!

Mohammed Bencheikh

Contents

List of Figures

List of Tables

List of Abbreviations

ACK - Acknowledgment
ADD_ADDR - Add Address
BFL - Blockchain-based Federated Learning
BLEST - Block Estimation Scheduler
BGP - Border Gateway Protocol
CC - Congestion Control
CMT-TCP - Concurrent Multipath SCTP
CWND - Congestion Window
DAPS - Delay-Aware Packet Scheduling
DA - Data ACK
DRL - Deep Reinforcement Learning
DSN - Data Sequence Number
ECF - Earliest Completion First
FL - Federated Learning
HOL - Head-of-Line
HSR - Highest Send Rate
IETF - Internet Engineering Task Force
IPv4 - Internet Protocol version 4
IPv6 - Internet Protocol version 6
IoV - Internet of Vehicles
IoT - Internet of Things
ISP - Internet Service Provider
ISPs - Internet Service Providers
LACP - Link Aggregation Control Protocol
LAN - Local Area Network
LIA - Linked Increase Algorithm
LRF - Lowest RTT First

LTS - Largest Time Space
LTE - Long-Term Evolution
LL/RP - Lowest Latency with Retransmission Penalization
MAC - Media Access Control
MP_CAPABLE - Multipath Capable
MP_FAIL - Multipath Failover
MP_FASTCLOSE - Multipath Fast Close
MP_JOIN - Multipath Join
MP_PRIO - Multipath Priority
MPQUIC - Multipath QUIC
MPTCP - Multipath TCP
MPTCP-ML - MPTCP with Machine Learning
NAT - Network Address Translation
NN - Neural Network
OLIA - Opportunistic Linked Increase Algorithm
OTIAS - Out-of-Order Transmission for In-Order Arrival
QUIC - Quick UDP Internet Connections
QoS - Quality of Service
REMOVE_ADDR - Remove Address
RR - Round Robin
RTT - Round-Trip Time
SCTP - Stream Control Transmission Protocol
SONATA - SCTP Over NAT
STTF - Shortest Transmission Time First
SYN - Synchronize
SYN/ACK - Synchronize/Acknowledge
TCP - Transmission Control Protocol
TCP/IP - Transmission Control Protocol/Internet Protocol
WRR - Weighted Round Robin

General introduction

In today's technologically advanced landscape, the networking domain has become increasingly diverse, characterized by a variety of technologies, standards, and a wide array of terminals. This transformation has been driven by the growing need for connectivity and ubiquitous service provision, aligning with the vision set forth by the International Telecommunication Union (ITU) [1]. However, the original TCP protocol, designed in the 1970s, falls short of meeting current requirements and fails to consider terminals equipped with multiple network interfaces [2].

To address the limitations of TCP, the concept of multihoming has gained prominence with the rise of devices featuring multiple integrated interfaces. This approach has enhanced network resilience and performance for end users [3]. However, the mismatch between multipath networks and TCP's single-path design renders TCP inefficient and inflexible.

In response to TCP's dominant position, the natural progression was to develop an extension that allows terminals to communicate by leveraging the resources of multiple paths, particularly to optimize the utilization of multi-interface terminals. This gave birth to the Multipath Transmission Control Protocol (MPTCP) [4]. Published simultaneously as an experimental standard by the Internet Engineering Task Force (IETF) [6], MPTCP was designed to address the limitations of TCP and seamlessly integrate into the existing Internet architecture without requiring major modifications to the distributed devices already deployed across the network. MPTCP serves as a promising technology for bandwidth aggregation, aiming to enhance connectivity and efficiently utilize available resources [5].

However, the heterogeneity of paths in MPTCP multihoming networks can lead to performance degradation due to head-of-line blocking [9]. This phenomenon manifests as packet reordering at the destination, where "out-of-order" packets must wait for the arrival of missing packets from the source. Within this context, this doctoral thesis focuses on improving the efficiency of MPTCP, particularly by addressing scheduling challenges. The objective is to maximize bandwidth utilization and alleviate congestion issues, especially in asymmetric networks. The aim of this thesis is to provide an original contribution to the field of MPTCP

scheduling by proposing an innovative approach that optimizes bandwidth utilization and resolves congestion problems in asymmetric networks. Ultimately, it seeks to enhance network performance and reliability while ensuring optimal connectivity for end users.
It is important to note that despite significant technological advancements, there are still additional research opportunities in the field of MPTCP scheduling. As networks and technologies evolve, new challenges may arise. Therefore, this thesis does not claim to exhaustively cover all aspects of MPTCP scheduling, but rather aims to pave the way for new investigations and in-depth exploration. To achieve this, we propose a novel scheduling methodology, which will be thoroughly detailed in the following chapters.

The *first chapter* of this thesis provides a comprehensive exploration of the concept of multihoming and its technological significance. We conduct a meticulous analysis of its implementation across various layers of the OSI model. At Layer 2, we examine the Link Aggregation Control Protocol (LACP), which enables the aggregation of multiple physical links into a unified logical channel, facilitating simultaneous connectivity to multiple networks. At Layer 3, we delve into the Border Gateway Protocol (BGP), used for routing between different networks. Finally, at Layer 4, we examine the intricacies of the Single-path Transmission Control Protocol (STCP) and the Multipath Transmission Control Protocol (MPTCP), which allow for the simultaneous utilization of multiple communication paths, leading to performance improvements and enhanced resilience.

The *second chapter* focuses specifically on the scheduling module within the context of MPTCP. We provide a comprehensive examination of various concepts and typologies governing this essential component to ensure quality of service. Through an in-depth review of research literature, state-of-the-art approaches, and existing comparisons, we emphasize the critical role of scheduling algorithms in data transmission.

The *third chapter* presents the proposed methodology, which constitutes an original contribution to MPTCP scheduling. We provide a detailed account of real-world experiments comprising 170 tests conducted to validate our approach. Our methodology aims to maximize bandwidth utilization and mitigate head-of-line blocking in asymmetric multihoming networks, thereby improving overall network performance.

Finally, the *fourth chapter* is dedicated to presenting and discussing the obtained results.

We demonstrate the effectiveness and relevance of our approach by comparing it to other existing algorithms. The results obtained confirm the significant contribution of our approach in terms of performance enhancement and bandwidth optimization. This thesis aims to contribute to network improvement by providing innovative solutions to maximize resource utilization and ensure better quality of service for end users.

Chapter 1

Multihoming Technologies

1.1 Introduction

1.2 Multihoming

Multihoming is a method of network connectivity that involves configuring multiple network interfaces or IP addresses in an end user's equipment. This technique has become necessary due to the increasing need for reliable and redundant Internet connectivity for businesses and individuals. The advantage of multiple network connectivity over a single network channel is that multiple connections are based on different network segments, making them more reliable. When a host has multiple IP addresses and is physically connected to multiple data networks, it is called a multi-homed server or computer. It acts as a router or gateway device, allowing the end user to benefit from faster and more reliable Internet connectivity.

1.3 Multihoming advantages :

The advantages of multihoming in communication networks are numerous and play a key role in improving the performance and reliability of connections. This section explores these advantages in detail to highlight the benefits they bring in a multipath connectivity context.

- Failover management and load balancing: One of the key benefits of Multihoming is the ability for a node to be simultaneously connected to multiple access networks. This feature enables better fault management by ensuring rapid recovery of connectivity in the event of a network failure. Multihoming also enables a load-balancing policy to be implemented between the different access networks, guaranteeing optimum use of available resources.

- Reliability and fault tolerance: Multihoming offers high reliability by enabling a site to migrate its connection to other access providers in the event of a network outage, ensuring continuity of connectivity for all nodes on the network. This capability can be divided into two sub-classes: non-transparent, where the loss of a connection results in the closure of the current transport session, but a new session can be established; and transparent, where the loss of a connection does not affect outgoing transport sessions.

- Load balancing: Multihoming allows the load of network traffic to be spread between different ISPs. By simultaneously distributing the traffic load between multiple connections between the mobile network and the Internet, load balancing ensures a balanced use of resources and offers a significant gain in terms of bandwidth.

- Performance: In the event of congestion on an interconnection link between two operator networks, Multihoming enables the "multihomed" site to ensure that its traffic does not use the congested link. This optimisation of communication paths helps to maintain high performance and low latency in the network.
- Routing policy : Multihoming offers the flexibility to switch the connection to a different ISP at any time, respecting legal constraints and user or application preferences. This ability to choose from among the available interfaces means that criteria such as cost, efficiency and specific needs can be taken into account.

1.4 Multihoming scenario

There are several multihoming scenarios, such as single links with multiple IP addresses, multiple interfaces with one IP address per interface, multiple links with one IP address, and multiple links with multiple IP addresses. Dual homing is a network architecture that has the advantage of being able to connect two devices to the network, each with two different access points. The first is the primary connection, while the second is the backup connection in case the first connection fails.

1.5 Multi-homing solutions

Nowadays, it is increasingly common for end hosts to be connected to the Internet via multiple interfaces. Similarly, Internet Service Providers (ISPs) offer multiple paths to allow routers to send traffic with the same destination on different paths. In data centres, a large number of servers often communicate with each other via redundant links. However, the protocols widely deployed today were not originally designed for multipath redundant networks. The lack of multi-path aware protocols hinders the potential benefits that multi-homing can provide. Therefore, different solutions have been proposed at each layer to bridge the current gap in layer use cases.

1.5.1 Data link layer

With regard to the link layer, link aggregation mechanisms have been introduced to aggregate the capacity of multiple switching interfaces. Typically, interfaces are aggregated under a single logical interface and share the same MAC address. This has the effect that the upper

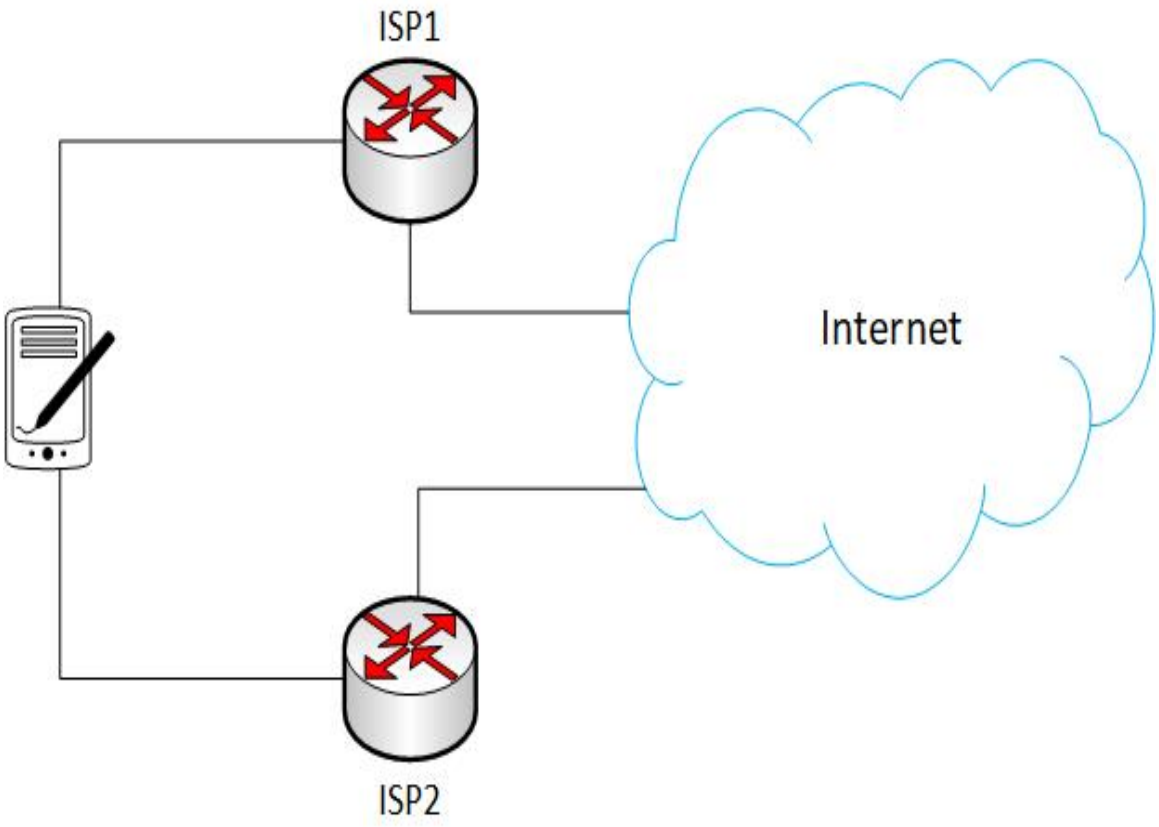

Figure 1.1: Host Multihoming.

layers are unaware of the link aggregation. The purpose of the virtual interface is to increase throughput, provide redundancy and distribute the traffic load between switches.
The Link Aggregation Control Protocol (LACP) is part of the IEEE 802.3ad specification, which allows multiple physical ports to be aggregated into a single virtual interface. The protocol provides automatic configuration and link aggregation, as well as the ability to dynamically detect link failures. LACP exchanges LACP packets between LAN ports in active or passive modes. Active mode initiates negotiation, while passive mode responds to packet negotiation. The protocol dynamically identifies the capabilities of each port group and exchanges this information between the LAN ports. However, the protocol requires synchronisation between the switches. Furthermore, the switch can only aggregate its links to the next hop. If the bottleneck is not the next hop, the aggregated links provide no benefit to the network bottleneck.

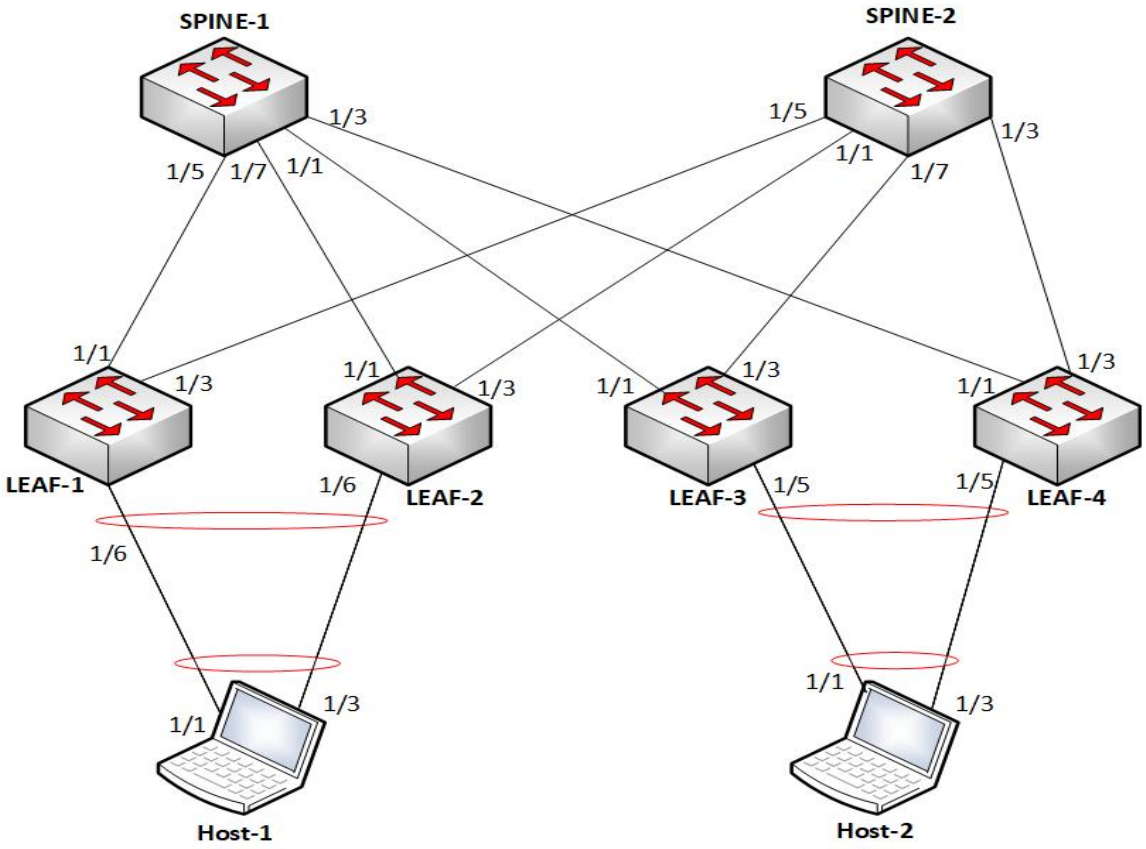

Figure 1.2: Multihoming Layer 2 "LACP".

1.5.2 Network Layer

There are various proposals for enabling multi-connection at the network layer. This subsection summarises some of the common methods.

- Site-Wide multi-homing: This method refers to a situation where a LAN is connected to the Internet via different transit providers . The hosts in such a network tend to be unique while the gateway router is responsible for multipathing. The router can move TCP flows from terminating hosts between different transit providers without interrupting any connections. Site-level multi-homing provides redundancy, load balancing and policy enforcement. However, the scheduler makes decisions based on the current network state and preferences rather than host requirements [1].

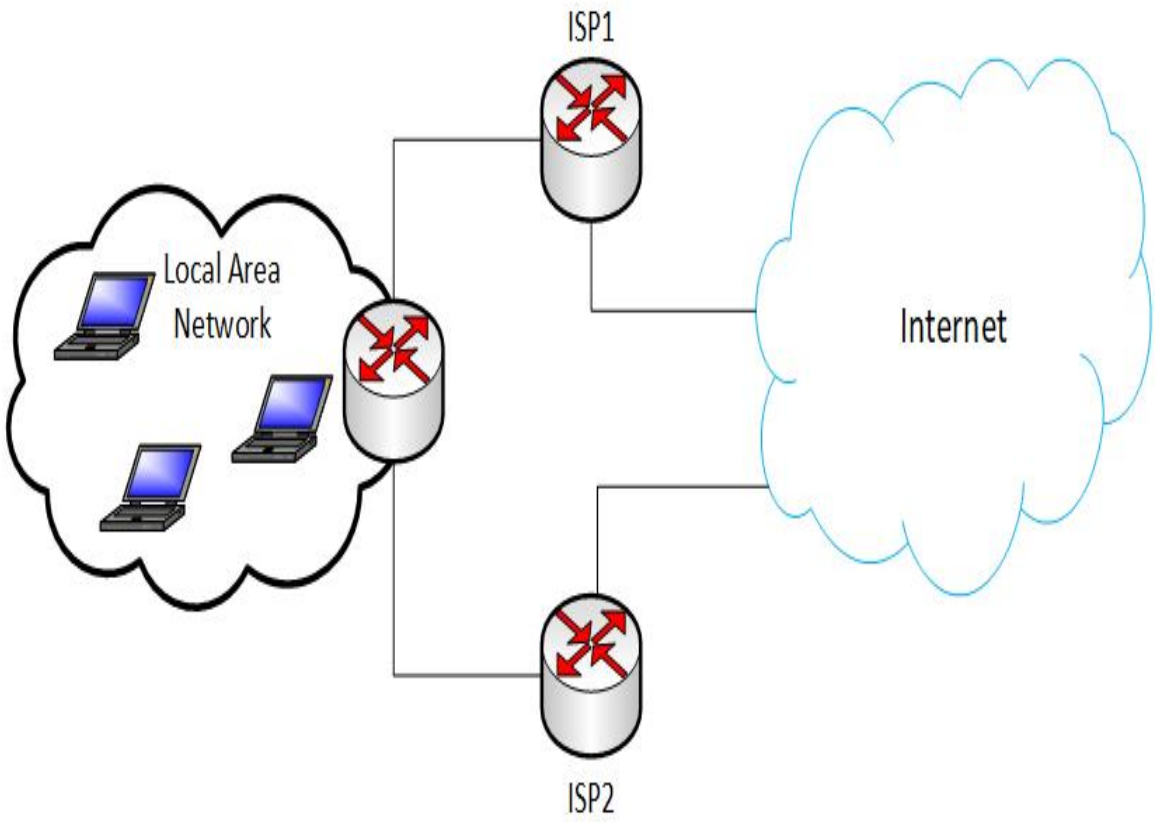

Figure 1.3: Network Multihomig.

1.5.2.1 BGP Multihoming case

To ensure increased network availability, multi-homed networks can also use the Border Gateway Protocol (BGP) in the event of a connection interruption due to a power failure. Multi-homed networks, working with BGP, thus provide redundancy by providing different routes in and out of the network for the traffic flow. Configuring BGP is a complex process that requires knowledge of BGP programming. Engineers have to plan the physical infrastructure, the number of routers needed, the connections in the facility, the bandwidth required by the ISP, etc. In addition, regular analysis should be performed to ensure that all destinations are accessible. A professional BGP configuration specialist should also be called in to optimise the BGP settings to ensure optimal traffic balancing according to the available bandwidth.

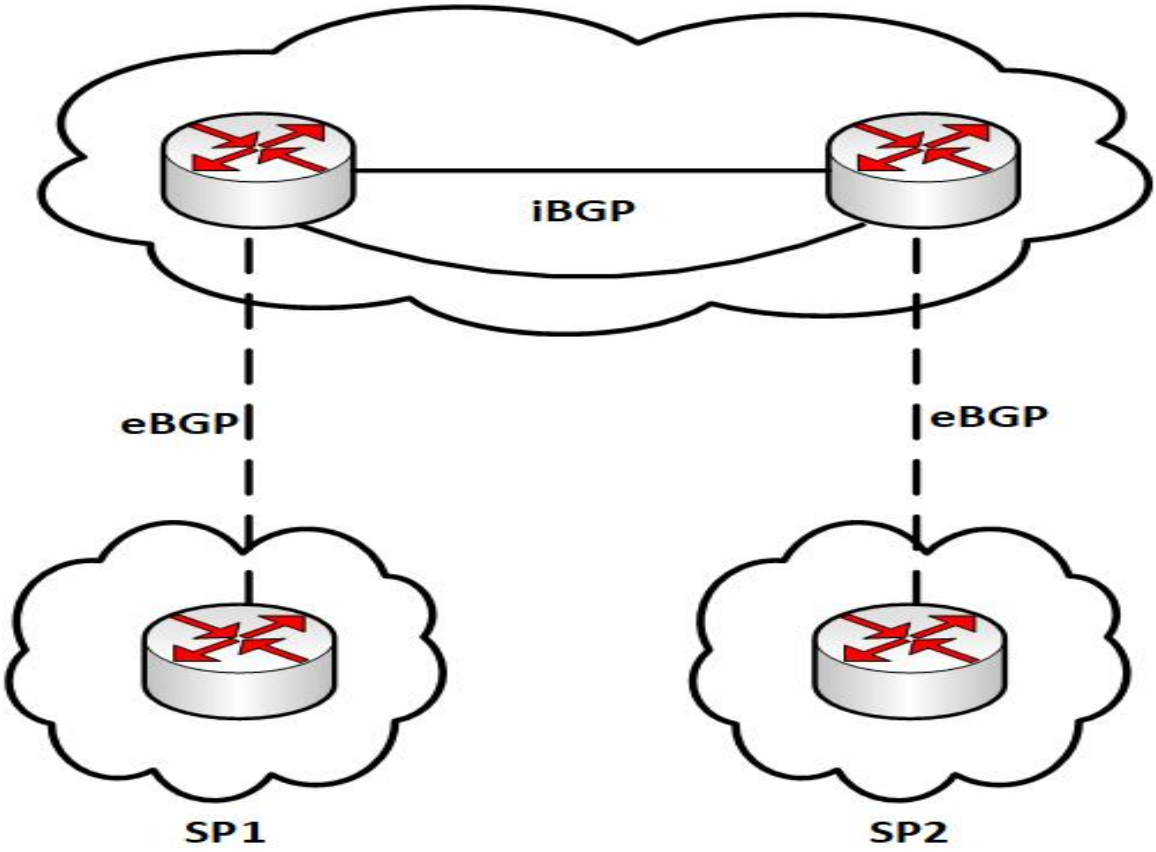

Figure 1.4: BGP Architecture.

1.5.3 Transport layer

In the transport layer, it is possible to take into account the characteristics and requirements of the end hosts. Various solutions have been proposed for this layer, including the SCTP and MPTCP protocols, which will be discussed in this section.

The Stream Control Transmission Protocol (SCTP) [16] is an alternative to the TCP transport protocol that allows data to be transferred in a non-strictly reliable manner. Data can be transferred in an ordered, partially ordered or unordered manner, and SCTP can support multiple IP addresses per connection. Unlike TCP, which uses a single data stream per connection, SCTP divides the data stream into small units called associations. In order to solve the problem of head blocking, SCTP breaks down the concept of a data sequence into what are called regions. These regions are essentially an association spread over several sequence streams. In addition, data transmission can take place over any path accessible to the association.

The congestion control (CC) function of SCTP is similar to that of TCP. For an association, all paths share the same CC metrics. However, when the association has multiple end nodes, CC can be performed separately on each path. One of the difficulties with SCTP is that it uses a different socket API than TCP. Therefore, applications must be specifically developed to use SCTP [1,2]. Originally, SCTP technology was incompatible with NAT and other middleboxes. However, a NAT module, called SONATA, was developed by Hayes and But [17] to support SCTP. Despite this, SCTP has not been widely adopted outside of niche applications [2].

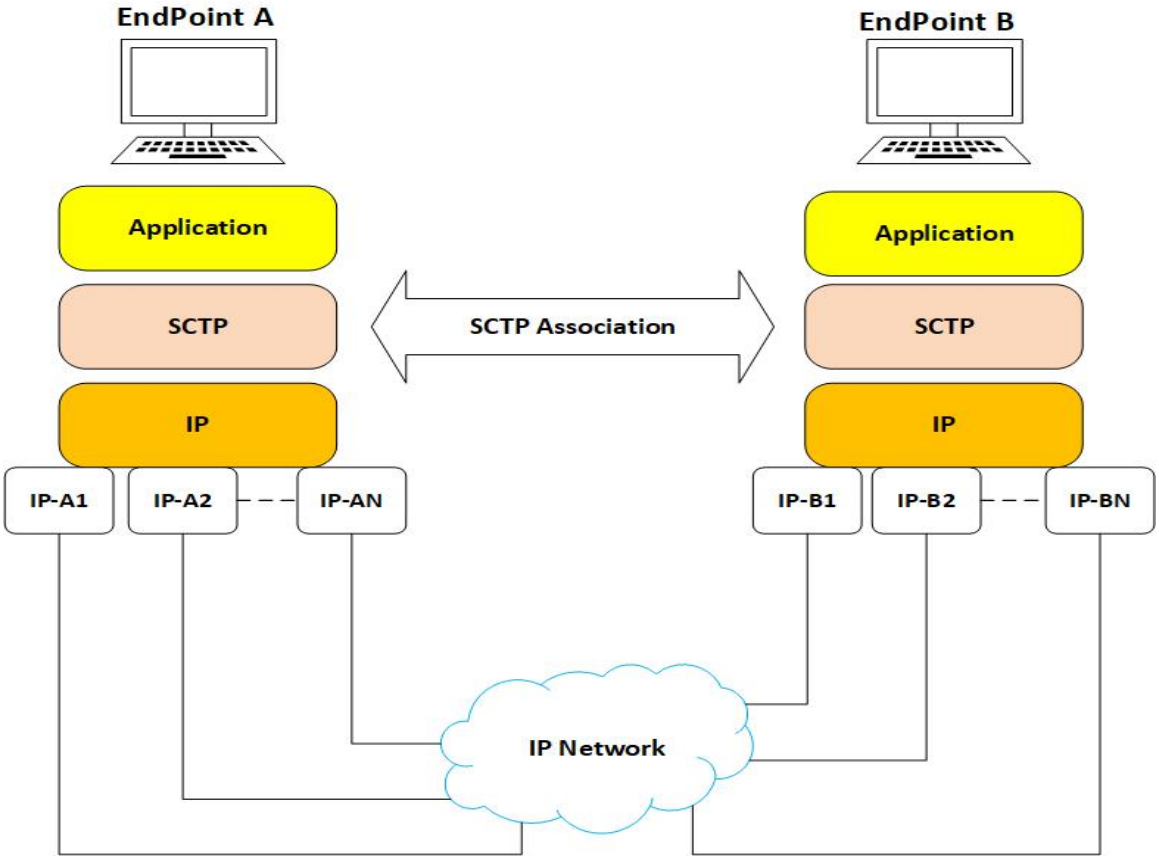

Figure 1.5: SCTP architecture.

Multi-path TCP [3], introduced in 2013, is considered an extension of TCP. It allows terminals hosted at different levels to communicate data simultaneously over different paths. MPTCP is compatible with existing TCP applications and functions within the current internet. The main difference between SCTP and MPTCP is that each MPTCP subflow behaves as a single, stand-alone TCP stream.

1.6 MPTCP architecture:

1.6.1 Overview

MPTCP (Multipath TCP) was published simultaneously as an experimental standard by IETF (Internet Engineering Task Force) [6]. The main driver for this was to be able to set up an optimal protocol that integrates with the current internet architecture, without the need to modify existing devices scattered on the network. Multipath TCP, or MPTCP, represents an alternative solution to TCP by incorporating a sub-level of semantics into its protocol stack, which combines different standard TCP sessions into an MPTCP connection, thus allowing multiple available interfaces to be used simultaneously to transmit data. The main features of MPTCP are presented in a concise manner, with an emphasis on how they are articulated.

The diagram in Figure 1.6 illustrates the MPTCP stack and its three main components: the congestion controller, the packet scheduler, and the path manager. This visual representation elucidates the relationships and interactions among these crucial elements, showcasing the underlying mechanism of MPTCP for optimizing its performance. The depicted general model demonstrates how the congestion controller provides input to the packet scheduler, which then determines the path selection criteria based on congestion calculations. The path manager subsequently manages the available paths based on the decisions made by the packet scheduler. This coordinated interplay among the components ensures efficient data transfer by optimizing packet scheduling, taking into account network congestion detected by the congestion controller.

1.7 MPTCP components

1.7.1 The Path-Manager

The Path-Manager is the first trigger for an MPTCP connection, when an exchange of the MP_CAPABLE option (indicating that the nodes support MPTCP) is sent in the SYN packet. It reports other available IP addresses using the MP_JOIN option after declaring them during the initialization configuration. The Path-Manager can also configure new subflows for an existing MPTCP connection using the ADD_ADDR option. MPTCP requires two types of sequence spaces, one for each subflow and one for the overall connection, as it aggregates several standard TCP sessions.

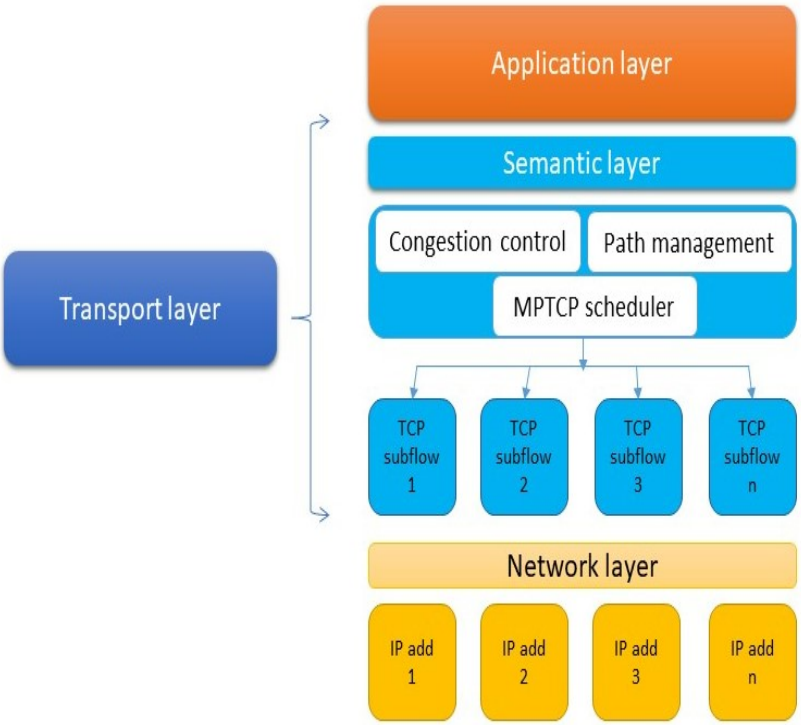

Figure 1.6: MPTCP Protocol Stack.

1.7.2 The packet scheduler

The packet scheduler segments the application layer data into segments and assigns each segment a sequence number, which it then directs to one of the sub-streams, which adds its own sequence number while taking into account the size of the congestion window, calculated by the congestion control algorithm.

1.7.3 The Congestion control

The Congestion control ensures reasonable allocation of network resources and guarantees fairness in the multipath configuration by adjusting the congestion window of the subflows to control the transmission rate. This is an important mechanism for packet scheduling to plan which segment needs to be forwarded by which subflow.

1.8 The data sequencing

The MPTCP protocol uses two levels of sequencing: one that denotes the data level (DSN) and the other that indicates the order number of the segment sent in this subflow. When the MPTCP sender starts transmitting data through different subflows, the data sequence number at the connection level must be mapped to the sequence number of the subflow. This operation must be performed like a normal TCP connection independently of the other subflows, with its own sequence numbers and accumulated acknowledgments (ACK). The MPTCP receiver uses the sequence number at the data level to reassemble segments from different subflows and transfer them in sequence to the application layer. Therefore, MPTCP employs a DSM (Data Sequence Mapping) to convert between the two sequence spaces.

1.9 MPTCP options:

The Multipath Transmission Control Protocol (MPTCP) allows an end device to create multiple TCP subflows simultaneously across different network interfaces. Each subflow is associated with a specific IP address and port. To enable the creation and management of these subflows, MPTCP uses specific options that are exchanged between the two ends of the connection. The main MPTCP options are :

1.9.1 MP CAPABLE:

This option is used to indicate that an end device supports MPTCP and to authenticate the configuration of more than one subflow. It is exchanged at connection establishment to allow both ends of the connection to negotiate the supported MPTCP options.

1.9.2 MP JOIN:

This option is used to associate a new subflow with an existing connection. It allows an end device to add a new path to an existing MPTCP connection without the need to reset the connection.

1.9.3 DSS (Data Sequence Signal):

This option is used to transmit the subflow data to the main connection. It allows data to be transmitted on different paths simultaneously and ensures that the data is received in the

$Value$	Total
$0X0$	MP CAPBLE
$0X1$	MP JOIN
$0X2$	DSS
$0X3$	ADD DDR
$0X4$	REMOVE ADDR
$0X5$	MP PRIO
$0X6$	MP FAIL
$0X7$	MP FASTCLOSE
$0X8$	unissiged
$0X8e$	unissiged
$0X9$	reserved for private use

Table 1.1: MPTCP option

correct order.

1.9.4 ADD ADDR:

This option is used to inform the other end of the connection of a new IP address associated with an existing subflow. It allows an end device to change the IP address associated with a subflow without the need to reset the connection.

1.9.5 REMOVE ADDR:

This option is used to remove one or more addresses previously added to a connection and terminate any subflows currently using that address.

1.9.6 MP PRIO:

This option is signalled when establishing the subflow to indicate whether to use the path as a regular or backup path.

1.9.7 MP FAIL:

This option is used to specify that a path can only be used if other links are not available. It ensures that data is transmitted on the most reliable path available.

1.9.8 MP FAST CLOSE:

This option is used to allow sudden closure of the MPTCP connection. It allows an end device to quickly close an MPTCP connection without having to wait for ongoing transfers to complete. These options play a key role in the operation of the MPTCP protocol by allowing the authentication of subflow configurations, the association of new subflows, the transmission of data between different subflows and the addition or deletion of IP addresses associated with different subflows.

1.10 The three ways handshake

When an end device seeks to establish an MPTCP connection with another device, it sends a SYN packet containing the MP CAPABLE option to indicate that it supports MPTCP. The receiver responds with a SYN/ACK packet also containing the MP CAPABLE option and other MPTCP options such as MP JOIN or ADD ADDR if required. The first device then responds with an ACK packet which confirms receipt of the first two packets and allows the data to be transmitted over the newly established MPTCP connection. The three-way handshake process is similar to that used to establish a standard TCP connection, but with the addition of MPTCP options that allow both ends to negotiate the supported options and transmit data over multiple paths simultaneously. Once the connection is established, data can be transmitted over different paths simultaneously by using the DSS (Data Sequence Signal) and MP PRIO options to specify the priorities of the different paths. When the communication is terminated, the connection can be closed with the MP FASTCLOSE option to allow for a quick and efficient closure of the connection. The MPTCP connection establishment process uses the three-way handshake similar to that used to establish a standard TCP connection, but with the addition of MPTCP options that allow communication over multiple paths simultaneously.

1.11 MPTCP Data transfer:

MPTCP data transfer takes place through subflows, which correspond to standard TCP connections with their own sequence numbering space. Bytes at the MPTCP level are numbered by a data sequence number (DSN) in addition to the subflow sequence numbers. All subflows share an MPTCP send buffer and an MPTCP receive buffer, while each subflow has its own

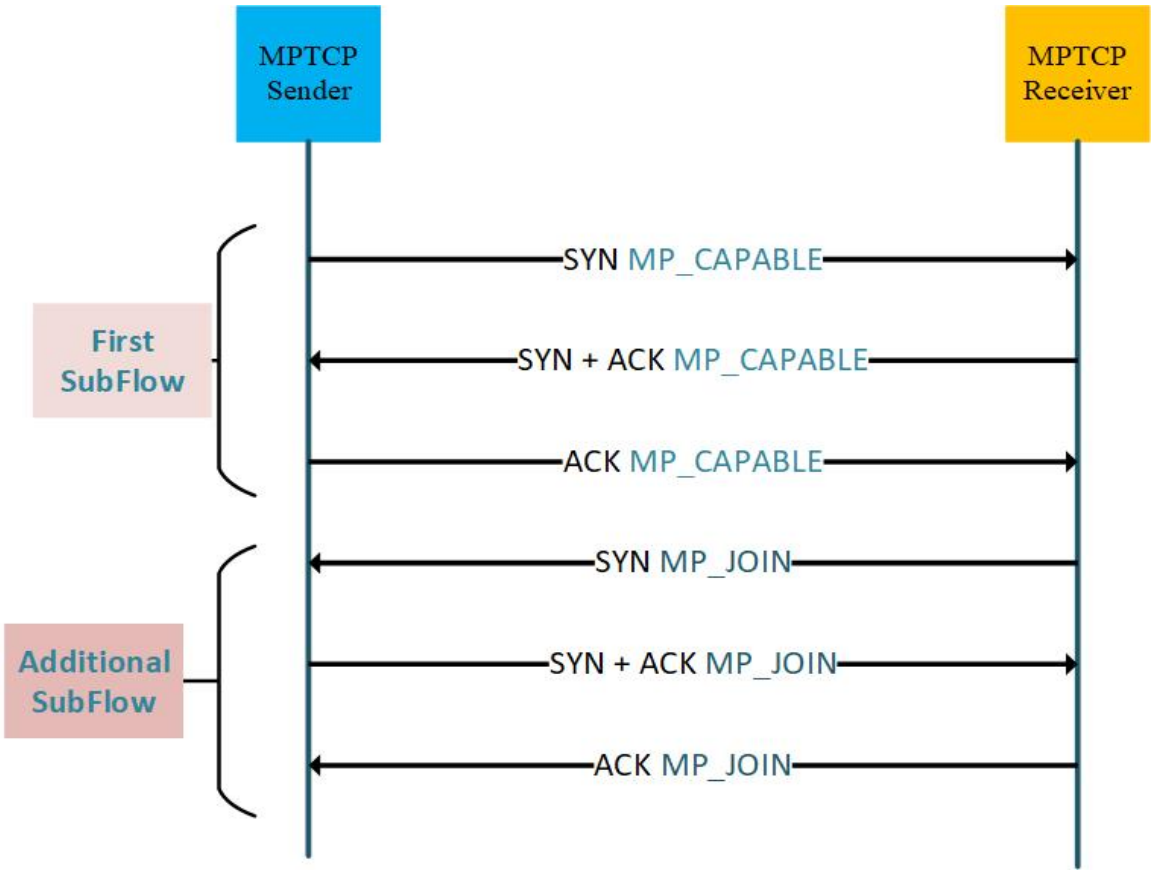

Figure 1.7: Three way Handshake MPTCP.

receive buffer to store data that may arrive in a different order than the subflow level. In effect, each TCP subflow receiver must deliver the data in order to the MPTCP receive buffer. When filling its MPTCP send buffer with bytes, each byte is numbered with a DSN. Then a scheduler decides on which subflow(s) to send the data. The bytes are transmitted on the selected subflows, where they are encapsulated in packets containing MPTCP information placed in the TCP option field. When a packet is received in sequence at the subflow level, the data is immediately delivered to the MPTCP receive buffer. The ACK number of the MPTCP level, called DATA ACK (DA), advances if the delivered data is also in order at the MPTCP level. The delivered data is acknowledged by the subflow receiver with a standard TCP ACK, and the current DA is entered into the TCP option field. The application then consumes the data in order from the MPTCP receive buffer. Currently, the MPTCP sender only releases data from the MPTCP send buffer when it has been acknowledged by DAs received on any subflow .

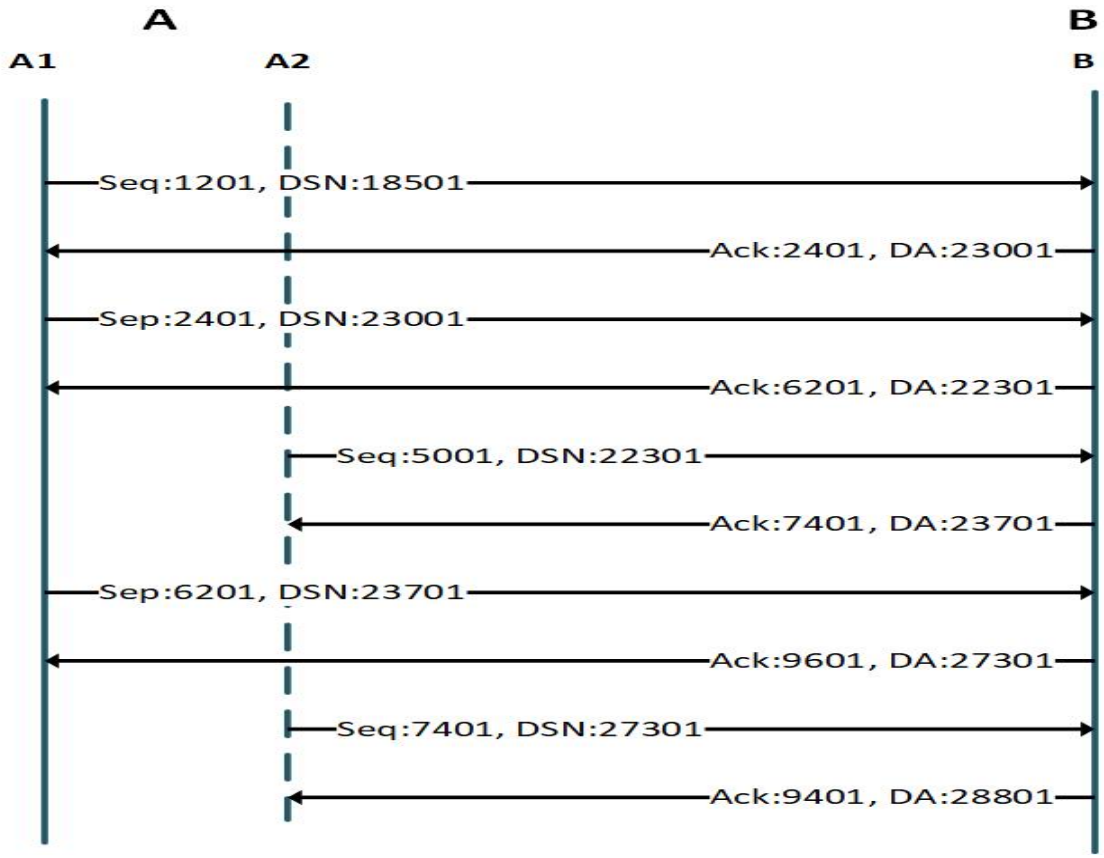

Figure 1.8: MPTCP Data Transport.

1.12 The Data sequence mapping

Data sequence mapping in MPTCP is a fundamental operation that ensures the integrity of data transmitted between the different ends of an MPTCP connection. This operation takes place in several steps: first, the data to be sent is divided between the available subflows and divided into segments which are then sent to the other end. Once the segments are received by the receiver, each subflow checks the sequence numbers and other information to ensure that the data received is consistent. Afterwards, there is a reordering of the data segments by the multipath layer to find out which data sequence numbers are associated with each packet, in order to send them to the application layer. The data sequence mappings consist of a 32-bit subflow sequence number and a 64-bit data sequence number, which are transmitted to the other end of the connection along with the segments.

1.12.1 Multipath TCP Acknowledgement

Multipath TCP introduces two levels of acknowledgement to ensure reliable data transmission: - A subflow level acknowledgement is sent on each regular TCP connection to confirm receipt of each segment. - A connection-level acknowledgement is used to provide cumulative Multipath TCP ACKS at the data sequence level. The first level of acknowledgement is triggered when a segment in a given subflow is received by the other end, which will respond with a subflow ACK. The second level of acknowledgement occurs when data sent on several subflows has been received from all subflows, and cumulative ACKs are provided. However, if a subflow with a large travel time is used, this may cause rescheduling and the data will remain in the rescheduling queue at the receiver for a long period of time. If a path fails, MPTCP automatically detects this situation and can use other available paths to continue transmitting data. This feature allows for more reliable and efficient data transmission by using multiple paths simultaneously.

1.12.2 Receive window:

One feature that allows a single receive window to be shared between all MPTCP subflows is the MPTCP receive window. This window allows data to be sent from any subflow as soon as the receiver is ready to accept it, thus improving the efficiency and reliability of data transmission [4].

1.12.3 Retransmission:

When data is lost, MPTCP uses a retransmission strategy to ensure reliable transmissions. Therefore, if a segment is lost on a given path, it can be retransmitted on the same subflow or on another available path. For example, if the loss was detected on the fastest path, it would be retransmitted on the second fastest available path. This functionality improves the quality and reliability of transmissions by using multiple paths simultaneously.

1.13 MPTCP congestion control algorithms

The congestion control algorithms for MPTCP must meet three objectives: improve throughput, do not interfere with other flows and balance congestion. Each of these algorithms has its advantages and disadvantages depending on the network conditions and application requirements. It is important to choose the appropriate congestion control algorithm based on

the specific needs of each use case.

The following is a detailed explanation of the most congestion control algorithm for MPTCP:

1.13.1 LIA (Linked Increases Algorithm):

In order to handle congestion in very high speed networks, the LIA algorithm is specifically built using a dynamic congestion window that calculates the available bandwidth on each transmission path. By adjusting congestion windows in real time based on available bandwidth and traffic load estimates, LIA is ideal for datacenters and very high speed networks.

1.13.2 CUBIC:

The CUBIC algorithm, released in 2005, was developed to avoid queuing and control congestion in high-speed networks. Through the use of a mathematical function to automatically adjust the size of the congestion window, CUBIC maintains optimal throughput while controlling packet loss rates. The cubical function used by CUBIC computes the variation of the congestion window over time, which in turn makes it ideal for high-speed networks, such as datacenter networks.

1.13.3 Westwood+:

Originally introduced in 2007, the Westwood+ algorithm was specifically created for wireless networks. Based on bandwidth measurements, Westwood+ seeks to optimise data flow by adaption of the transmission rate to variations in available bandwidth. The congestion control method estimates available bandwidth and measures network congestion, which allows Westwood+ to maintain a dynamic congestion window that adapts in real time to changes in available bandwidth.

1.13.4 Balia:

This congestion control technique was published in 2011 and is intended for low-bandwidth networks. Balia employs congestion control strategies to avoid package losses. To adjust the size of the congestion window, he employs a congestion control approach based on the available bandwidth and the detection of packet losses. on practice, this technique employs a dynamic congestion window to avoid traffic congestion and packet losses on low-bandwidth networks.

1.13.5 OLIA:

This congestion control technique was published in 2015 and has been adapted for high-speed networks. To regulate the flow of data, OLIA employs a control theory-based approach. To adjust the size of the congestion window, he employs a congestion control approach that is based on the available bandwidth and traffic charge calculation. Concisely, it employs a dynamic congestion window that evaluates the available bandwidth for each transmission path and regulates the size of the congestion window in real time. OLIA is particularly well-suited for datacenter networks and high-speed networks.

1.13.6 HyStart++:

This congestion-control technique was published in 2017 and is used to detect packet losses and reduce transmission rates to prevent congestion from worsening. HyStart++ employs a congestion-control approach based on packet loss detection to reduce transmission rates and adjust the size of the congestion window. Concisely, HyStart++ employs a congestion window that detects packet loss and reduces the size of the congestion window to prevent the congestion from worsening. This technique is well suited to low-bandwidth networks and mobile networks.

1.14 Congestion control algorithms comparison

The table 1.2 provides a description and comparison of several congestion control algorithms used in Multipath TCP. These algorithms were designed to address the challenges of transmitting data over multiple paths efficiently and reliably.[79, 86]

Algorithm	Goal	Cons	Pros
CUBIC	Prevent queuing and congestion on high-speed networks	May cause congestion on limited bandwidth networks	Optimal throughput, especially for datacenters
Westwood+	Control data rate in wireless networks	Packet loss in high-speed networks	Designed for variable bandwidth wireless networks
Balia	Avoid packet loss in low-bandwidth networks	May cause congestion on high-speed networks	Prevention of congestion and packet loss, suitable for bandwidth-constrained networks
OLIA	Detect losses on each subflow using exceptional loss indications	Packet loss in limited bandwidth networks	Assess bandwidth availability for each path in high-speed networks and datacenters
HyStart++	Minimize congestion window length to prevent traffic congestion in mobile and low bandwidth networks	Congestion problems in high-speed networks	Minimize congestion window size for mobile and low bandwidth networks
LIA	Use separate sending windows for each subflow	Potential disadvantage in detecting exceptional losses compared to OLIA	Traffic regulation in high-speed networks

Table 1.2: Congestion algorithm Descriptions

1.15 MPTCP Applications:

1.15.1 MPTCP application videoconference:

Using mptcp to enhance video quality during video conferences over unreliable networks on heterogeneous networks mptcp can be used to enhance video quality during video conferences mptcp can prevent packet losses and delays that could impair video quality by using multiple connections at once to distribute traffic along various paths experimental research has shown

that using mptcp during video conferences over hybrid networks improved the video quality and increased the success rate of calls

1.15.2 MPTCP application online gaming

Applying MPTCP to improve online gaming performance: MPTCP can be used to enhance online gaming performance by distributing traffic across many connections to prevent delays and packet losses. According to a study, using MPTCP increased data throughput while decreasing latency . This improved the performance of online games.

1.15.3 MPTCP application mobile networks

Implementing MPTCP to increase Internet connectivity on mobile devices: MPTCP can be used to increase Internet connectivity on mobile devices by using several connections at once to distribute traffic across various paths. This could benefit in avoiding dead zones and enhancing quality of service (QoS) when using bandwidth-intensive food-related applications. According to a study, using MPTCP helped applications run more smoothly on mobile devices, particularly in terms of debit and response time [9].

1.15.4 MPTCP application in data centers:

MPTCP can be used in data centers to enhance load balancing and boost network resilience. MPTCP can help prevent jitter and distribute the load of the traffic across several links by using several connections simultaneously to distribute the traffic along multiple routes. According to a study, using MPTCP helped to improve the charge balancing in a data center .

1.15.5 MPTCP application in network industry

To increase network performance, a number of networking companies, including Cisco, Huawei, and Nokia, have included MPTCP into their products. MPTCP can enhance load balancing and boost network resilience by using several connections at once to distribute traffic over multiple paths. These businesses have also created tools to help network administrators monitor and control MPTCP connections. It is possible to get online setup guidelines for Cisco, Huawei, and Nokia routers (Cisco Systems, Inc., Huawei Technologies Co., Ltd., and Nokia Corporation, 2018). Additionally, Apple included MPTCP to its iOS products to enhance

mobile devices' Internet connectivity. MPTCP can increase application performance and prevent dead zones by using multiple connections at once to split traffic across several paths. Additionally, Apple has put security measures in place to shield users from potential attacks related to the use of MPTCP (Apple Inc., 2013). On their Android handsets, Samsung, LG, and Huawei have employed mptcp to blend wi-fi and 4g for enhanced network speed and reliability. Smartphones such as the Samsung Galaxy S7 and LG G5 use mptcp to offer faster download rates and better video streaming. Furthermore, the 3rd generation partnership project 3gpp, which is in charge of defining mobile telecommunications standards, has highlighted mptcp as a favored strategy for boosting network performance in its release. This will enable mobile network operators to use mptcp in their networks to increase load balancing and failover capabilities.

1.16 Challenges and Opportunities for Enhancing Multi-Path TCP Performance

While MPTCP has many advantages, it also has flaws and concerns that must be addressed. The performance of existing MPTCP congestion management techniques may suffer if the network architecture deviates from a basic topology with a common bottleneck and pathways with comparable characteristics. Furthermore, due to the utilization of numerous subflows and potential attacks such as subflow hijacking and denial-of-service, MPTCP brings new security issues. Furthermore, while MPTCP is rapidly being employed in mobile devices where energy efficiency is crucial, much more study is required to minimize energy usage while retaining optimal performance. The most significant research question is how to increase MPTCP performance over complicated networks such as the Internet. This necessitates the development of novel congestion management algorithms that can adapt to changing network circumstances and efficiently employ numerous pathways. Furthermore, additional study is required on how to improve MPTCP packet scheduling to guarantee fairness, congestion control, security, energy usage, and network architecture are all taken into account.

1.17 Conclusion

In this chapter, we have examined the concept of multihoming in the context of communications networks, focusing on the ability of terminals to operate all their interfaces and to have multiple IP addresses, allocated to each of the interfaces available on a node. This approach

meets a range of needs and offers a number of potential advantages.
Multihoming offers a number of advantages that have not yet been fully exploited. As well as guaranteeing continuous access to the Internet, it allows each application in use to be assigned the network interface or connection that best meets its specific requirements.
We chose the MPTCP protocol because of its status as the future reference protocol for the Internet. This protocol deals robustly with the issues of load balancing, availability, scalability, and ensures optimum performance in all aspects, including security, confidentiality, addressing, and many others. Crucially, this solution remains transparent to the upper layers of the communications model, to ensure continuous connectivity to the network, without disruption or reconfiguration of ongoing applications.

The next chapter will take a closer look at the MPTCP protocol scheduling, detailing its various features and highlighting the crucial importance of scheduling, which is the essential component for ensuring optimal load balancing and meeting the requirements of next-generation networks.

Chapter 2

MPTCP Scheduling Component

2.1 Introduction

In networking, scheduling refers to the process of allocating network resources, such as bandwidth, buffer space or processing time, to different traffic flows or applications in a fair and efficient manner. Scheduling plays a critical role in computer networks by allocating network resources, including bandwidth, buffer space, and processing time, to different traffic flows or applications in a fair and efficient manner. The main goal of scheduling is to optimize the use of available resources while ensuring that each flow receives a fair share of resources according to its Quality of Service (QoS) requirements. Scheduling algorithms are crucial in solving resource management problems in computer networks by allowing an efficient and fair allocation of resources between different traffic flows or applications, guaranteeing an optimal level of QoS for each flow. These algorithms can be implemented at different layers of the OSI model to meet the specific needs of applications and network protocols. Scheduling is a highly challenging problem that can be addressed at different levels of decision making, from tactical to operational. At the tactical level, scheduling can be used to plan resource allocation, determine traffic control policies, and prioritize different types of traffic. Meanwhile, at the operational level, scheduling can be used for queue management, load balancing, congestion control, and error management. The main objective of scheduling is to maximize the efficiency and performance of the network by balancing resource constraints and user demands.

2.2 Scheduling notion:

The problem of scheduling entails the optimization of task execution by organizing them over time, while satisfying various temporal constraints such as deadlines and sequencing requirements, as well as constraints pertaining to the utilization and availability of the required resources. In computer networking, scheduling plays a vital role in allocating network resources efficiently and fairly among different applications and traffic flows. Proactive and reactive scheduling are two different approaches that can be used to manage network resources.

2.3 Scheduling behaviour process :

2.3.1 Proactive scheduling

Proactive scheduling involves predicting future network demands and allocating resources in advance to meet those demands. This approach is used when a high level of predictability is expected, such as in long-term planning. For example, a network administrator may use proactive scheduling to allocate bandwidth for a conference or event that is scheduled to take place in the future.

2.3.2 Reactive scheduling

Reactive scheduling, on the other hand, involves dynamically adjusting resource allocation in response to changes in network conditions and demand. This approach is often used in real-time applications, such as online gaming or video streaming, where network conditions are unpredictable. For instance, in video streaming, reactive scheduling can adjust the amount of bandwidth allocated to a stream in real-time based on changes in network conditions or the number of users.

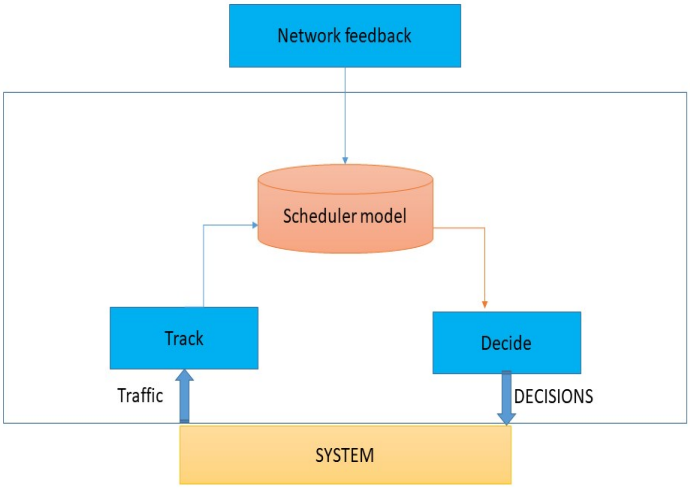

Figure 2.1: Reactive scheduling workflow.

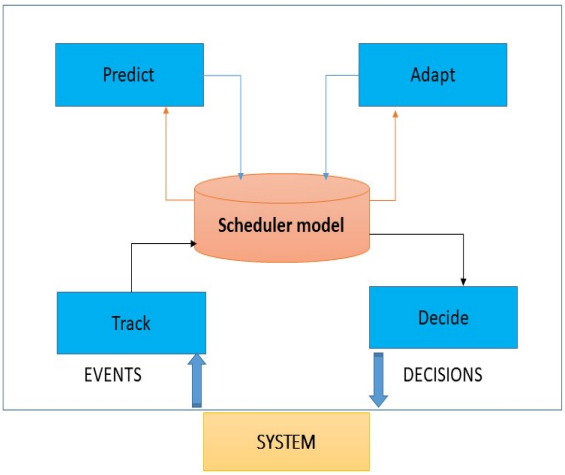

Figure 2.2: Proactive scheduling workflow.

2.4 MPTCP Scheduling

The scheduling component of Multi-Path TCP is in charge of spreading data among several subflows to increase transmission efficiency. The scheduler selects which subflow should be utilized to transport packets depending on numerous parameters such as available bandwidth, delay, and congestion levels. The scheduler used can have a considerable impact on MPTCP speed since it impacts how well MPTCP can use numerous subflows and react to changes in network conditions.

MPTCP packet scheduling algorithm research has focused on enhancing fairness, congestion control, security, energy consumption, and network topology awareness. Some suggested algorithms try to increase fairness by guaranteeing that each subflow receives an equal amount of the available bandwidth. Others want to improve congestion control by dynamically altering the data transfer rate of each subflow based on network scenarios. Furthermore, several algorithms try to increase security by preventing attackers from exploiting weaknesses in the scheduling process.

Overall, improving the scheduling in MPTCP is crucial for attaining optimal performance across a variety of network topologies while also reacting to the network changes.

The MPTCP protocol was created to address the challenges posed by complex networks,

where the default MPTCP algorithm is less efficient. To address these issues, the MPTCP dedicates an entire feature to the packet scheduling. Many optimization algorithms have been developed in this regard to improve the performance of MPTCP in various use scenarios. This study delves into these algorithms and their effects on MPTCP performance.

2.5 State of the Art in Multi-Path TCP (MPTCP) Schedulers

The utilization of redundant network resources in Multi-Path TCP (MPTCP) offers significant advantages, including bandwidth aggregation, fault tolerance, and load balancing. MPTCP outperforms single-path protocols, particularly in terms of throughput [22-23].

Since its initial adoption in 2013, extensive research has been conducted to explore various MPTCP schedulers. These investigations have focused on comparing schedulers based on path characteristics, emphasizing the need for reliable scheduling to ensure optimal performance, especially in asymmetric scenarios [24-25-35]. Several schedulers, such as Lowest RTT First (LRF) [8], Largest Window Space (LWS), Largest Time Space (LTS), and Highest Send Rate (HSR) [26], have been designed to maximize throughput and maintain multipath performance. Load balancing has also been enhanced through approaches like Weighted Round Robin (WRR) [27], which builds upon the Round Robin (RR) mode [8]. Moreover, specific attention has been given to optimizing scheduling for video applications [32-33] and introducing cross-layer information to improve scheduler efficiency [34].

To address the challenge of Head-of-Line (HoL) blocking, several scheduling policies have been proposed. These include DAPS-1 [10], DAPS-2 [11], Lowest Latency with Retransmission Penalization (LL/RP) [9], and Block Level Estimation Scheduler (BLEST) [13]. Notably, BLEST has demonstrated effective reduction of retransmission caused by out-of-order segments. Real-time transmission has also been considered with the introduction of Earliest Completion First (ECF) [34]. Another strategy, known as Shortest Transmission Time First (STTF) [29], provides similar benefits to ECF with added precision.

Recent advancements in MPTCP schedulers have explored adaptive techniques and learning-based approaches in heterogeneous networks. These techniques leverage machine learning and deep learning to optimize network performance in unpredictable conditions. In [77] proposed a federated learning (FL) technique to evaluate coordinated TCP (C-TCP) flows for IoT applications over WiFi networks. Chung et al. [46] introduced MPTCP-ML, a path management scheme utilizing machine learning mechanisms to assess active path quality based

on collected patterns and heuristics. Additionally, researchers have developed a throughput-based learning algorithm for identifying optimal signal quality rates[43].

The evaluation of durable MPTCP flows over heterogeneous networks, including WiFi and cellular networks, has been extensively investigated by [73]. Their learning algorithm exploits route heterogeneity and examines its impact on overall MPTCP performance. These techniques have potential applications in coordinating drone convoys and in city railway systems [61, 72]. Furthermore, Chiariotti et al. [45] introduced LEAP, a latency-controlled end-to-end aggregation protocol that ensures Quality of Service (QoS) requirements for delay-sensitive systems.

In recent years, sophisticated techniques based on deep learning and reinforcement learning have emerged to enhance MPTCP protocols. ReLeS, a reinforcement learning-based scheduler for MPTCP, utilizes deep reinforcement learning to train a neural network capable of determining the optimal MPTCP distribution policy [87]. A learning-based multipath QUIC (MPQUIC) scheduler, dynamically selects the most suitable scheduling strategy based on path impact analysis and throughput optimization [85]. Furthermore, game theory has been applied to define utility functions for cooperative MPTCP, resulting in improved throughput and responsiveness [78]. Distributed deep reinforcement learning-based congestion control algorithms have also been proposed to address Head-of-Line blocking and enhance throughput [54].

DeepCC introduces advanced deep reinforcement learning techniques to improve the performance of existing TCP schemes in cellular networks [73]. Additionally, blockchain-based federated learning (BFL) designs have been proposed to reduce system delay and ensure privacy in on-vehicle machine learning processing [66].

Despite considerable research efforts, discussions surrounding bandwidth requirements and scheduling obstacles persist. Further advancements are necessary to achieve a satisfactory level of performance in MPTCP scheduling.

2.6 MPTCP packet scheduling algorithms

As part of the throughput improvement and better use of the network capacity. MPTCP packet scheduling is the process that ensures the quality of service of data flows at a bottleneck by massive control related to the available bandwidth. The following part, an explanatory review of the different approaches mentioned beforehand, including the most popular MPTCP schedulers.

2.6.1 Round Robin Scheduler (RR)

The Round Robin Scheduler is a scheduler that ensures a more equitable distribution of data across multiple subflows. Its traditional operation is to select a subflow from those established in the MPTCP session when the sender has data to transmit [8]. Packets are then sent on this subflow as long as its congestion window allows. Thereafter, the scheduler moves on to the next subflow by applying the same forwarding mechanism. This process is repeated cyclically for all subflows with an available congestion window. In order to determine the amount of data to send on each subflow, the Round Robin uses an equation. This equation determines the usable part of the sender's congestion window by deducting the number of packets that have not yet been acknowledged in the relevant subflow. This value is then compared to the receiver's receive window (RWND), and the minimum between the two is transmitted. Thus, before sending data on each subflow, this operation is performed to optimise the use of available capacity. However, this approach has limitations when dealing with heterogeneous paths, where the differences in delay between the subflows of the session are significant. The disordered arrival of packets in the buffer leads to buffer blocking problems and higher application delays, as packets are not delivered to the application layer in the expected order. Despite its simplicity and common use in academic contexts to illustrate the behaviour of a packet scheduler, Round Robin may be less effective in situations where the characteristics of the subflows vary in terms of the bandwidth offered or the delay.

2.6.2 The Delay-Aware Packet Scheduling (DAPS1)

Distributed Adaptive Packet Scheduling [10] is an approach that aims to solve the receive buffer blocking problem often encountered with round robin scheduling. Its fundamental concept is to strategically distribute segments to ensure that they are received in the desired order across all available subflows. The main objective of DAPS is to optimise the use of the receive buffer to reduce the risk of blocking. DAPS operates in a well-defined three-phase process. First, it identifies the order in which the subflows are available for transmission. Based on this order, the second phase determines when to send the segments to the subflows . Finally, the third phase is dedicated to the actual sending of the segments to the recipient, taking into account the previously defined deadlines. The DAPS algorithm expertly exploits the ratio between the Round Trip Time and the Congestion Window Size to accurately determine the amount of data to be sent on each path. It assigns high priority to packets with the lowest sequence numbers, associating them with the path offering the best performance, including packets requiring retransmission, regardless of their initial path. The DAPS takes

a proactive approach, which means that it only reacts to the next data flow to be sent. As a result , DAPS significantly improves the order of arrival of packets at the receiver, especially in scenarios where the RTT differences are considerable. However, when the differences in delay between paths are negligible, it can work similarly to conventional round robin. It is important to note that DAPS requires a higher level of computation, as it generates a complete schedule rather than simply sending packets in sequence.

2.6.3 Low-RTT-First Scheduler (LRF)

The Low-RTT-First is a default scheduler in the Linux kernel [8]. LRF is a strategy that gives priority to the subflow with the smallest RTT as long as its CWND is open, In other words, the segment will be directed on the subflow that has the lowest smoothed round trip time. LRF was designed to maximize throughput for each subflow. This reactive algorithm leads the subflow with good path performance to handle more data and achieve a certain load balancing efficiency .

2.6.3.1 LRF extentions:

In order to overcome the difficulties inherent in the LRF scheduler, two extensions were developed, namely LowRTT with PR (Opportunistic Retransmission and Penalisation) and LowRTT with BM (Opportunistic Retransmission and Marking). These extensions are based on the same operating principles as the LRF, but differ in that their behaviour changes when a buffer block occurs [9].

2.6.3.2 The PR extension of the LRF :

This aims to resolve blocking situations by immediately re-injecting the segments responsible for the buffer blockage into an available subflow with both a sufficiently large congestion window and a lower RTT than the subflow causing the blockage. This approach makes it possible to quickly overcome head-of-line blocking and to compensate for the disparities in RTT between the various subflows. As for the penalty mechanism, it reduces the CWND of the subflow causing the blockage by half, thus limiting its use of the congestion window on the subflow characterised by a high RTT. This reduction of the congestion window helps to reduce the data sending rate and to mitigate the bufferbloat effect on the subflow concerned. It should be noted that the PR extension alleviates the constraint of limiting the receive window. However, in the long term, it can have negative repercussions on the overall performance of the

system. Indeed, the penalised subflow will require a considerable amount of time to increase its congestion window, which will lead to underutilisation of the subflow and suboptimal capacity aggregation. Furthermore, this mechanism only solves the buffer blocking problem temporarily, as the congestion window of the penalised path will continue to increase until the blocking is repeated.

2.6.3.3 The BM LRF :

The Bufferbloat Mitigation algorithm with LRF (Bufferbloat Mitigation BM) presents an alternative to the RP mechanism by providing an innovative approach to mitigate the adverse effects of bufferbloat. Bufferbloat refers to the excessive accumulation of data in network buffers, which leads to significant delays and performance degradation. The main objective of this algorithm is to control and reduce these delays by limiting the amount of data sent on each subflow. The central idea of this algorithm is to continuously monitor the difference between the minimum smoothed RTT (SRTTmin) and the smoothed RTT (SRTT) on the same subflow. When SRTT deviates from SRTTmin due to data being sent, this indicates the presence of bufferbloat. To remedy this situation, an upper limit, called CWNDlimit, is set for the congestion window of each subflow. The formula used to calculate CWNDlimit is as follows:

$$CWND_{\text{Lim}} = \frac{\lambda \cdot SRTT_{\min}}{SRTT} \tag{2.1}$$

2.6.4 Out-of-Order Transmission for In-Order Arrival Scheduler (OTIAS)

The Out-of-Order Transmission for In-Order Arrival (OTIAS) algorithm is a proactive scheduler designed for real-time data transfer with the objective of reducing completion time and jitter.[12] One of the main challenges of multipath data transfer is the possibility of messy arrival of data at the receiver, which can lead to significant performance issues for real-time applications. OTIAS addresses this challenge by directing each data segment to a subflow that has the minimum transfer time to reach the receiver, regardless of whether the subflow has space in its congestion window (CWND). This approach ensures that disordered data segments are transmitted through different subflows, allowing them to reach the MPTCP receiver in order. The OTIAS algorithm achieves this by dynamically selecting the subflow

with the shortest transfer time for each data segment, based on factors such as the available bandwidth and the congestion level of each subflow. This approach allows OTIAS to continuously adapt to changing network conditions, to optimise the performance of the MPTCP connection and to ensure that data arrives in the right order at the receiver. This approach allows OTIAS to continuously adapt to changing network conditions, optimise the performance of the MPTCP connection and ensure that the data arrives in order at the receiver. By directing each data segment to a subflow with the minimum transfer time, OTIAS ensures that the data arrives in order at the receiver and dynamically adapts to changing network conditions to optimise performance.

2.6.5 Block Estimation scheduler (BLEST)

The BLEST algorithm is a proactive approach that makes decisions when scheduling each segment. It is based on a new metric that estimates whether sending a segment in a slow subflow will cause a buffer block or not. This assessment is made using the send window at the MPTCP session level [13].
To better understand how BLEST works, let us consider the existence of two subflows, namely the F (fast) subflow corresponding to the fastest, and the S (slow) subflow corresponding to the slowest. These subflows are capable of transmitting data with round-trip latencies, referred to as RTTf and RTTs respectively. In the case where the data of a subflow is not acknowledged and occupies space in the MPTCP session sending window (MPTCPsw), a blocking situation may occur, resulting in the blocking of the whole multipath connection.
The BLEST algorithm assumes that if a packet is transmitted on the slow subflow S, it will occupy space in the MPTCP send window for a minimum duration of RTTs. This means that there is still space available for the fastest subflow F to use. Therefore, blocking will occur if F is not able to send data, for example due to lack of space in the sending window occupied by S. In order to maximise the sending of data in F, the BLEST algorithm estimates a quantity X which represents the amount of data that can be sent on F without causing blocking during a period of rtts. This estimate is calculated according to the following formula:

$$rtts = \frac{RTT_s}{RTT_F} \tag{2.2}$$

$$X = MSS_F \cdot \left(CWND + \left(\frac{rtts - 1}{2}\right) \cdot rtts\right) \tag{2.3}$$

Subsequently, a careful check is made of the relevance of this data to the MPTCP sending window. The estimation of the value of X is based on a fundamental assumption that for each rtts, the CWND congestion window is incremented by 1, following the principles of the congestion avoidance strategy. However, it should be noted that this estimate may have some inaccuracy, and in order to correct this approximation, an adjustment factor is introduced. This adjustment factor is progressively increased when blockages are observed, indicating excessive congestion in the subflow S, and thus a potential reduction of the available space in the MPTCP sending window. After evaluating the relevance of these data for the MPTCP sending window, we proceed to a detailed analysis.

The estimation of the value of X is based on the fundamental principle that for each rtts, the CWND congestion window is increased by 1, thus following the principles of the congestion avoidance strategy. However, it should be noted that this estimate may have some approximations, and to correct for these deviations, an adjustment factor is introduced. This adjustment factor is modulated according to the occurrence of blockages, indicating congestion in the subflow S, and decreases when these blockage situations are absent.
If :

$$X \cdot \lambda > |M| \cdot MSSs \cdot (inflightS + 1) \tag{2.4}$$

The next segment will not be transmitted via the S subflows. This mechanism helps to reduce the risk of Hol-Blocking and therefore minimises the number of potential retransmissions.

2.6.6 The Earliest Completion First (ECF)

ECF algorithm has been developed to efficiently manage path heterogeneity in the context of the MPTCP protocol. Using a multi-criteria approach, ECF aims to minimise the idle time of the fast subflow by taking into account parameters such as round-trip time, bandwidth availability (congestion windows) and the amount of data ready to be transmitted. The underlying logic of ECF is based on the presence of two distinct subflows: the fast F subflow (RTTF) and the slow S subflow (RTTS). When the F subflow has an available congestion window, the pending data is routed through this path. However, if subflow F is not available, a decision must be made whether to send packets through the slow subflow or wait for the fast subflow to become available. To determine this decision, the ECF scheduler estimates the transmission time of a set of packets K in subflow F. Since MPTCP manages two data

buffers, one at the MPTCP session and one at each subflow established for that session, ECF takes into account the unscheduled packets in subflow F and evaluates the availability of space in its congestion window. Using this evaluation, ECF determines the necessary waiting time and decides to route packets through subflow F based on an accurate estimate.[34]

$$RTTF + \frac{K}{\mathrm{CWND}_F} \cdot RTTF \tag{2.5}$$

Taking into consideration that the k packets will require RTTs to arrive at the receiver, whether they are all transmitted or not, the ECF algorithm considers the following inequality:

$$RTTF + \frac{K}{\mathrm{CWND}_F} \cdot RTTF < RTTS \tag{2.6}$$

When this condition is satisfied, a decision is made to wait and reuse subflow F once it is available again, as it offers a shorter transfer time. On the other hand, if the condition :

$$RTTF + \frac{K}{\mathrm{CWND}_F} \cdot RTTF \geq RTTS \tag{2.7}$$

This means that the MPTCP buffer contains enough packets to allow them to be sent instantly on the subflow S, without the need to wait. The method exploits the bandwidth aggregation of the two paths, which leads to a significant reduction in transfer time. By optimising the use of available resources, the overall efficiency and performance of the MPTCP system is maximised.

2.6.7 The Shortest Transmission Time First (STTF)

The Shortest Transmission Time First scheduler is a scheduler that uses an estimate of the transfer time of packets to decide which subflow to send them on. This method takes into account the characteristics of each available path in the MPTCP session, allowing each data packet to be assigned to the subflow with the shortest transfer time.[29]
The STTF scheduler approach is similar to that of other schedulers such as ECF and OTIAS, but it also incorporates other important optimisations. It takes into account the congestion state of the network (Slow Start or Congestion Avoidance) and allows the rescheduling of already planned packets in case of transmission interruption. In addition, STTF implements minor optimisations such as resetting RTT values to avoid using expired RTT values. The operation of the STTF scheduler is as follows: when packets are ready to be sent, STTF

analyses all available paths to predict the transfer time of each packet if it were transmitted on each subflow. Thus, each unscheduled packet is associated with the subflow that has the lowest estimated transmission time. This estimate is calculated by taking into account parameters such as the smoothed RTT, the number of packets waiting in the considered subflow, as well as the state of congestion, be it Slow Start, Congestion Avoidance or a transition from Slow Start to Congestion Avoidance after an RTT.
The congestion state requirement is a distinct feature of the STTF scheduler. Unlike other schedulers that assume that the MPTCP connection is in Congestion Avoidance mode, STTF takes into account the different congestion phases of the network. This allows more accurate decisions to be made, especially in the case of short flow applications that may start and end before leaving the Slow Start phase. Once packets are scheduled, the STTF scheduler delegates the management of their transmission to the MPTCP output engine. In the event of a transmission interruption, for example if an acknowledgement is received before all packets have been transferred, a rescheduling mechanism is activated. STTF then removes the planned scheduling for the relevant subflow and updates the transfer time estimates for the packets not yet sent. This update takes into account the changes in sending conditions resulting from the event that caused the transmission interruption. [24]

2.7 MPTCP scheduling challenges and limitations

The scheduling of traffic in MPTCP is an exciting topic of study. Scheduling plays a key role in the optimal distribution of flows across the multiple path availability, thus offering the possibility to fully exploit the benefits of multipathability. However, this field is not without inherent constraints and limitations that deserve a thorough analysis. Understanding these issues is an essential preliminary step to understanding the challenges associated with designing an efficient scheduling module for the MPTCP protocol. In the following sections, we will discuss the main limitations and problems that arise in the area of MPTCP scheduling. By appreciating these constraints, we will be able to explore the various challenges that are faced and identify potential directions for improving the efficiency and performance of scheduling in the context of the MPTCP protocol.

2.7.1 Head-of-Line phenomenon:

The Head-of-Line phenomenon occurs when packets are transmitted in sequential order across multiple paths. If a packet is lost or delayed on a path, all subsequent packets in the queue

cannot be transmitted until the packet at the head of the queue is successfully transmitted.

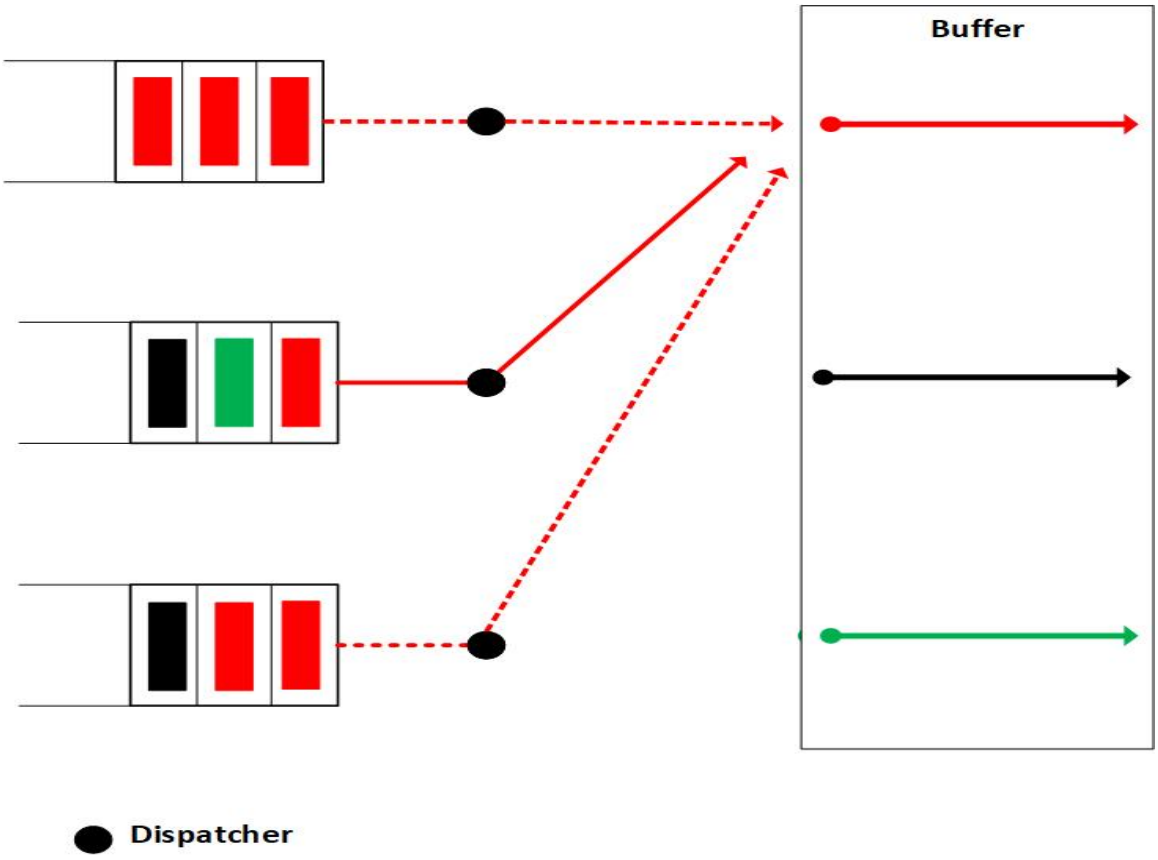

Figure 2.3: Head-of-line blocking .

This phenomenon has been addressed in research by using methods to mitigate the HoL phenomenon, such as the use of reserve buffers and modifying the scheduling to avoid congestion.

2.7.2 Path heterogeneity:

Networks may have a variety of links with different characteristics and technologies such as capacity, latency, throughput, reliability, etc. The selection of the best links for each data stream depends on the quality of each link and the available capacity. Research has explored that this heterogeneity poses a problem for the scheduling component in terms of performance.

2.7.3 Retransmissions:

In addition, retransmissions are another important constraint for MPTCP scheduling, as they can consume valuable resources such as bandwidth and energy. Scheduling algorithms must be able to minimise the number of retransmissions required to ensure efficient data transmission, while avoiding network congestion.

2.7.4 Processing complexity:

MPTCP scheduling can be expensive in terms of processing time and computational power, especially for large-scale networks. The literature has investigated how to deal with processing complexity, such as using heuristic methods, optimising scheduling algorithms and reducing the amount of data to be processed.

2.7.5 Energy aspect:

Energy is an important constraint for mobile and wireless sensor networks. Research work has explicitly focused on methods to mitigate energy consumption by dynamically adapting link selection to changing network conditions and by using energy-efficient scheduling strategies to extend the battery life of mobile devices and sensors.

2.7.6 Traffic security:

Traffic security is an important concern for MPTCP scheduling constraints. Potential security risks such as packet interception and denial of service attacks must be taken into account. Encryption techniques can be used to protect sensitive data transmitted over the different flows. In addition, MPTCP scheduling can be used to avoid congestion points that are potentially vulnerable to DDoS attacks.

2.7.7 The out-of-order phenomenon

The out-of-order is a major challenge in MPTCP scheduling, as it can lead to delays and packet losses, which can negatively impact network performance. Scheduling algorithms must be designed to take this issue into account and ensure efficient data transmission.

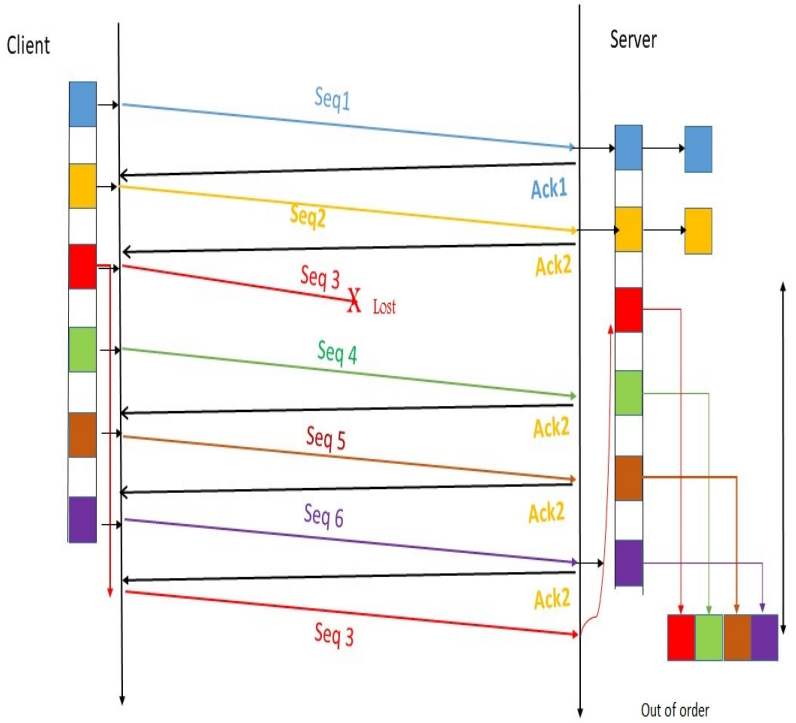

Figure 2.4: Out of order exemple case.

2.8 Scheduling comparaison

The table 2.1 serves as an informative resource, detailing how each scheduler operates and the specific features it utilizes to make decisions. It provides a clear overview of the functioning principles and criteria employed by each scheduling algorithm in MPTCP. Moreover, the table also highlights the suitable use cases or scenarios where each scheduler is most effective.

Scheduler	Description	Input Algorithm
LRF [8]	Prioritizes subflows with the lowest smoothed RTT (sRTT).	RTT
WRR [27]	Extends Round-Robin with subflow weight for optimal load balancing.	CWND, Data segment size
ESPC [35]	Estimates subflow capacity and maintains it at saturation, considering path congestion.	RTT, CWND, Loss
BLEST [13]	Evaluates potential head-of-line blocking and dynamically adjusts scheduling to prevent it.	RTT, CWND, packet in-flight, MSS
OTIAS [12]	Estimates packet arrival times across subflows and selects earliest arrivals for scheduling.	RTT, CWND, packet in-flight
DAPS1 [10]	Schedules packets based on sequence numbers and path delays, ensuring in-order delivery.	RTT
DAPS2 [11]	Enhanced version of DAPS-1 considering maximum receive buffer blocking time.	RTT, CWND, packet in-flight
ECF [34]	Identifies if scheduling a slow subflow will idle faster subflows.	RTT, CWND
STTF [29]	Estimates transfer time over subflows and selects the one with the shortest time.	RTT, CWND, packet in-flight
LTS [22]	Chooses subflows with the lowest latency-to-window space ratio, improving throughput.	CWND, packet in-flight, RTT
LWS [26]	Prefers subflows with the largest window congestion space, improving throughput.	RTT, CWND, packet in-flight
HSR [26]	Chooses the subflow with the highest send rate, improving throughput.	RTT, CWND, MSS

Table 2.1: Scheduler Descriptions

2.9 Conclusion

In this chapter, we have conducted an in-depth study on MPTCP scheduling and provided a comprehensive overview of the state-of-the-art in this field. We have explored various scheduling algorithms, their functionalities, and their suitability for different network scenarios. The challenges posed by MPTCP scheduling have been thoroughly examined, highlighting the need for innovative and efficient scheduling techniques.

Through the literature review, we have identified the key issues and research gaps in MPTCP scheduling. The existing algorithms have shown promising results, but there is still room for improvement in terms of performance optimization, congestion control, and fairness among flows. Addressing these challenges requires the development of novel scheduling approaches that take into account the heterogeneous nature of MPTCP networks and provide efficient resource utilization.

Overall, this chapter has provided a solid foundation for further research on MPTCP scheduling. The insights gained from the literature review and analysis of existing algorithms will guide the development of our proposed scheduling approach in the subsequent chapters. By addressing the identified challenges and contributing to the field of MPTCP scheduling, this research aims to enhance the performance and reliability of MPTCP networks, ultimately providing optimal connectivity and quality of service for end users.

Chapter 3

Contribution and Method

3.1 Introduction

Multi-Path TCP schedulers play a pivotal role in managing data allocation across multiple network paths within a single communication session. Their primary objective is to intelligently distribute data, optimizing throughput, reducing latency, and enhancing the overall performance of MPTCP connections. These schedulers are crucial for load balancing, improving resilience against network failures, and enabling seamless multipath communication. However, despite the numerous advantages offered by MPTCP, certain weaknesses persist, specifically head-of-line blocking in asymmetric scenarios and the impact of out-of-order packet delivery. Head-of-line blocking occurs when packets on one path experience delays or get lost, causing subsequent packets to be held back at the receiver until the delayed or lost packets are retransmitted. This phenomenon significantly reduces throughput and hampers the overall performance of MPTCP. In scenarios where network paths have different characteristics, such as varying latency or bandwidth, head-of-line blocking becomes more pronounced, posing a substantial challenge for MPTCP schedulers. Out-of-order packet delivery represents another significant issue. This occurs when packets from different paths arrive at the receiver in a different order than they were transmitted. Several factors, including varying network paths, congestion, or packet reordering mechanisms, can contribute to out-of-order delivery. The improper reassembly of data at the receiver's end disrupts the seamless transmission, resulting in increased latency and inefficient utilization of available paths. In light of these existing research efforts, we propose a reliable scheduling approach that sheds light on efficient bandwidth utilization. This approach aims to overcome the weaknesses associated with head-of-line blocking and out-of-order delivery. Our approach contributes to the development of more robust and reliable MPTCP connections, enabling efficient and seamless multipath communication in diverse network environments.

3.2 Presentation

Despite the numerous advantages of Multipath TCP in networking, such as bandwidth aggregation, MPTCP has a significant weakness due to path heterogeneity, resulting in the Head-of-Line (HoL) blocking phenomenon that can decrease the overall throughput. The scheduling component in the design of MPTCP determines how data is transferred across multiple paths, with each scheduler adopting a subflow selection strategy based on metrics such as Round-Trip Time (RTT), window size, Maximum Segment Size (MSS), loss rate, completion time, and jitter, whether reactive or proactive. To address this issue, we propose

a scheduling strategy for MPTCP that alternates between a reactive scheduler called "Lowest RTT first" (LRF) and a proactive scheduler named "Block Estimation scheduler" (BLEST) [13-28] to ensure better utilization of network bandwidth while simultaneously addressing the buffer occupancy issue. The reactive scheduler LRF, while being designated as the default scheduler for handling transmissions in asymmetric paths due to its better bandwidth utilization, may still suffer from the HoL-blocking problem. This leads to the proposal of BLEST, which avoids the fastest subflow and waits for a more favorable one, potentially reducing the HoL phenomenon. However, BLEST may not effectively utilize fast idle subflows that have nothing to send. Our contribution involves the creation of a pattern model that combines the advantages of maximal bandwidth utilization with LRF, while resolving the HoL problem through the intervention of the proactive scheduler BLEST. BLEST regulates the window size by decrementing it and appropriately selecting the responsible subflows for the HoL, this adjusting allows to reduce buffer occupancy. Furthermore, LRF is reapplied in a final step to optimize the utilization of fast subflows. This cooperation between LRF and BLEST leverages the strengths of each approach while mitigating their weaknesses through this alternation.

3.3 Alternation process:

To implement this approach, we define a time interval T that corresponds to one-third of the estimated time, as indicated in Equations 1 and 2. During this time interval, we employ an algorithm to inspect out-of-order segments, marking them with a counter C1 in the reactive phase, which will reach its final value after time T has elapsed. Then, we call on BLEST (proactive phase) during another time interval, ensuring that the condition C2 = C1 is satisfied, where C2 is a counter for out-of-order segments in the proactive phase, to apply the second exchange. By using this alternation approach between LRF and BLEST, we can leverage the advantages of each scheduler while addressing HoL issues and optimizing the utilization of network resources for improved performance of the MPTCP protocol. This innovative approach aims to strike a balance between utilizing the fastest subflow and mitigating the negative impacts of HoL, ultimately improving the overall performance of the MPTCP protocol. This innovative approach aims to strike a balance between utilizing the fastest subflow and mitigating the negative impacts of HoL, ultimately improving the overall performance of the MPTCP protocol in scheduling subflows. In this thesis, we propose a scheduling approach, using an algorithm to inspect the out-of-order segments. To define the

fashion of this alternating task, our proposed algorithm is as follows: Considering a flow of elements Data = (source; seq_num; next_seq_num; DSN) including source address, receiver sequence number, expected sequence number and DSN (data sequence number). The proposed algorithm tracks the transmission conduct between the client and the server on two levels of analysis: subflow sequencing and data sequencing. We define the output counters (Out_Seq_num, Out_DSN) which detect the number of noted out-of-order segments so as to apply the sysctl command to alternate the desired scheduler.

3.3.1 The out-of-order inspection algorithm

The 'Counter' represents a numerical value that reflects the total count of segments identified as being out-of-order. This counter is derived from two sources: the sum of out-of-order sequence numbers and the sum of out-of-order data sequence numbers. To track the MPTCP traffic, we focus on the disordered segments. The implemented Algorithm compares each segment with its previous one at the subflow level by the expected and received sequence number and secondly at the data level by Data Sequence Number (DSN). The alternation in the MPTCP scheduling algorithm is presented for the first time, whose idea of switching between the two schedulers is inspired by an analysis of previous results in a single scheduling strategy where a counter is used to intervene at the right time. The alternation algorithm retains the minimum number of out-of-order segments recorded in each scheduler tested alone (in a single strategy), which gives a relevant and justified alternation. Indeed, the following is a definition of the parameters involved in the alternation scheme as well as a description of the process steps in Fig. 3.1.

Our algorithm computes first the expected transfer time is calculated using equation 1:

$$T = \frac{Data}{Bandwidth} \tag{3.1}$$

Then, it determines the execution time "Q" of the first scheduler according to formula 2:

$$Q = \frac{T}{3} \tag{3.2}$$

Where 3 is the number of steps.

During this Q interval, the algorithm counts the number of out-of-order segments and stores

Algorithm 1 : MPTCP Out-of order inspection

Input: Data = {Source ; SeqNum ; NextSeqNum ; Dsn}
Output: $\text{Out}_{Dsn}; Out_{SeqNum}$
$\text{Out}_{Dsn} = 0$
$\text{Out}_{SeqNum} = 0$
for every instance i **in** Data **do**
    if $Source_i = Source_{i-1}$
        if $SeqNum_i = NextSeqNum_{i-1}$
            if $Dsn_i < Dsn_{i-1}$
            $Out_{Dsn} = Out_{Dsn} + 1$
        else
        $Out_{SeqNum} = Out_{SeqNum} + 1$
        break
    end if
end for
The complexity (time and space) in Big O notation is O(N)

Figure 3.1: MPTCP Out Of Order Inspection Algorithm.

it in a counter named C1. The first swap takes place after the interval Q, where the 2nd scheduler is launched and the algorithm recalculates the number of the out-of-order segments of this step and saves it in another counter named C2. During this step, the algorithm calculates and verifies at the same time the condition C1 = C2. Once this condition is satisfied, the algorithm swaps to the 1st scheduler.

3.4 Setup configuration and network topology

The experimental setup used in our testbed is realized with two stations, one is considered to be a client and the other a server, in which we have implemented the Linux kernel proposed for MPTCP [8]. These machines have been interconnected by four Ethernet interfaces to the access routers as shown in Fig. 3.3.

In the context of network configuration, several measures were implemented to optimize performance and control network traffic on the client, server, and router sides. On both

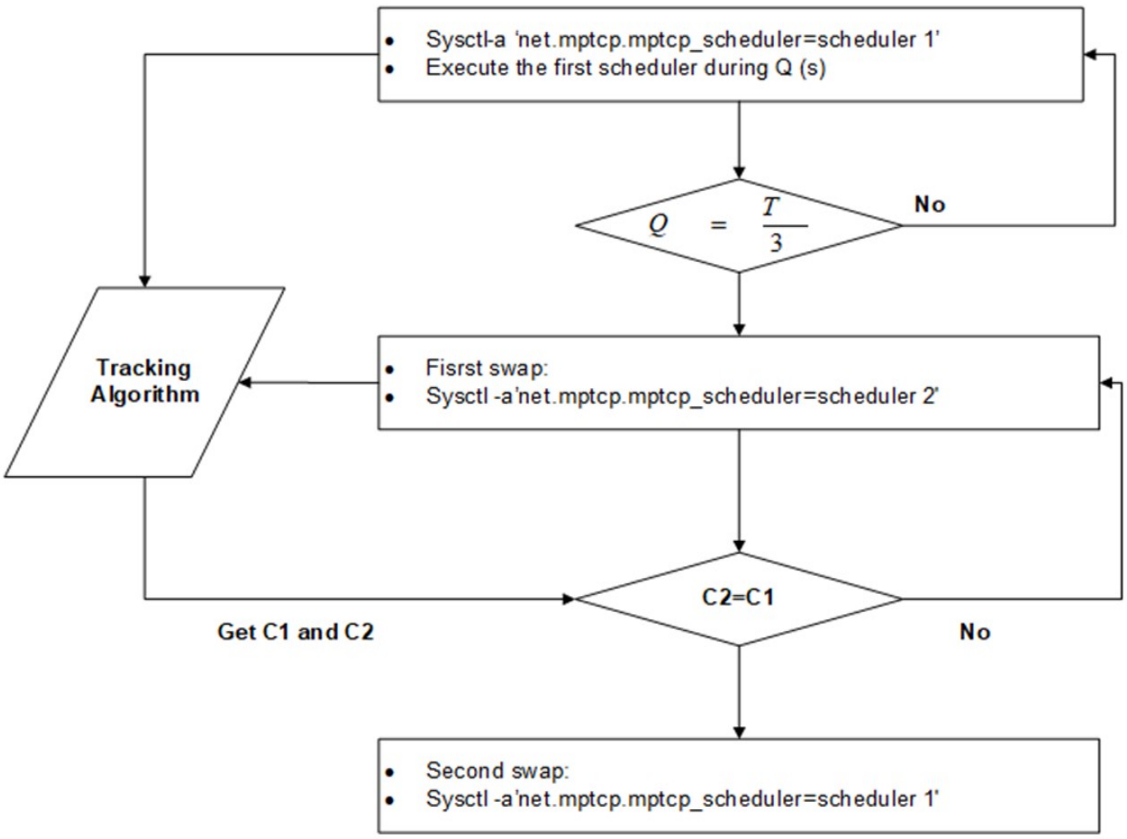

Figure 3.2: Alternation Process.

the client and server sides, the default size of the send and receive buffers was set to 65536 bytes using the 'sysctl -w' command. This adjustment ensures efficient data transmission and reception by allocating an appropriate buffer size.

Additionally, the Linux Traffic Control (TC) utility was employed as part of the interface configuration to regulate the bandwidth. By implementing TC, the network administrator was able to shape and control the flow of data on each subflow. Shaping the bandwidth allows for prioritization and allocation of network resources based on specific requirements.

On the router's side, the setup procedure involved three essential steps. Firstly, an Access Control List (ACL) was created to regulate network access and permit specific services. The ACL, named "test-services," was configured to allow only desired protocols and ports, such as ICMP (ping), HTTPS (port 443), HTTP (port 80), FTP (ports 20 and 21), TFTP (port 69), SSH (port 22), and Telnet (port 23). This level of control ensures that network resources are utilized effectively and security measures are enforced.

The second step in the router's configuration process was to create a class-map that corre-

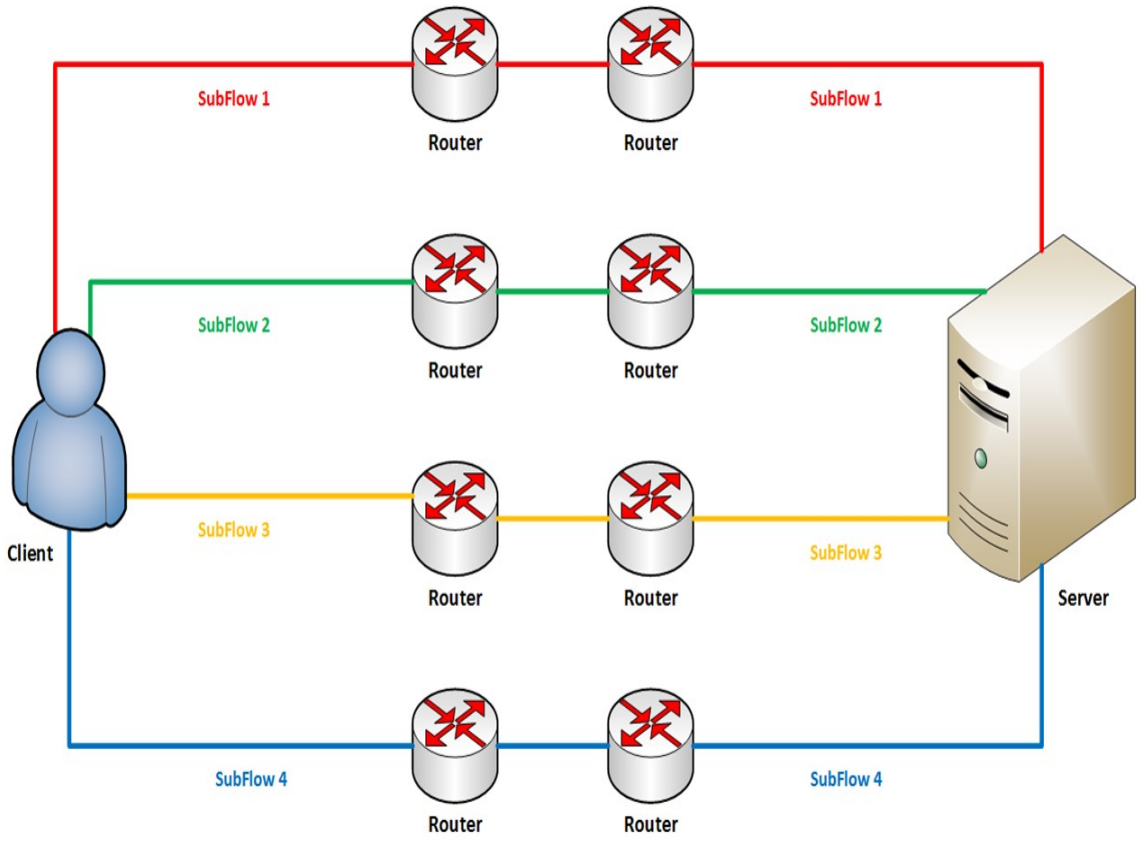

Figure 3.3: Testbed Topology.

sponds to the control list of each sub-network. These class-maps define the traffic characteristics and match the specified ACL rules for each subflow. By associating the appropriate class-map with each sub-network, traffic can be classified and treated accordingly, ensuring proper handling based on predefined criteria.
Finally, a policy-map was created to link the ACL with the respective class-map for each subflow. This policy-map acts as a policy enforcement mechanism, allowing for the application of specific actions and shaping parameters based on the matched traffic characteristics. In this case, the policy-map named "SubflowX-Shaping" (e.g., Subflow1-Shaping) was associated with the corresponding class-map and implemented the shaping parameters to limit the bandwidth of each subflow to 1 Mbps, 2 Mbps, 5 Mbps, and 10 Mbps, respectively.

The characteristics of the network equipments and the bandwidth offered in the different subflows are presented in Table 3.1 and in the section 3.5.

In view of setting up our Ubuntu FTP server, we first have to install the necessary soft-

Device	Description
Client	Intel(R) Core(TM) i5-5200U 2.20GHz (with SSE4.2) Linux 4.14.24+
Server	Intel(R) Core(TM) i7-3770 3.40GHz (with SSE4.2)Linux 4.14.24+
Router	Cisco 2821 w/iOS 15.1(4) M 512D/256F CME 8.6

Table 3.1: Hardware characteristics.

ware. For this, Ubuntu machines have the adequate version already with vsftpd. We use the terminal command Line:sudo apt-get install vsftpd, then we change the configuration file by setting the enable function to yes. Afterwards, an explorer has to be installed on the client-side. For this FTP traffic, we use a file size of 344 Mb . Therefore, we perform the same measurements using a file size of 540 Mb to validate our approach. We present the results found by the evaluation of the different scheduling algorithms previously listed the command that calls scheduler with sysctl -a"net.mptcp.mptcp_scheduler= ".
The traffic information is introduced through Wireshark where we specify filters to get our MPTCP dataset traffic and record it in order to contribute to the in-depth analysis of these packet traces. All graphs that will be presented in this work are created using a python program. This testbed experiment is an investigation of the MPTCP schedulers previously mentioned.

In order to set up our Ubuntu FTP server, we begin by installing the necessary software. Fortunately, Ubuntu machines come with vsftpd pre-installed, which is the appropriate version for our needs. We can easily install it using the following command in the terminal: sudo apt-get install vsftpd
Once vsftpd is installed, we proceed to modify its configuration file to enable the FTP server. We locate the configuration file and make the necessary changes by setting the "enable" function to "yes".
On the client-side, we install an FTP client software to establish a connection with the FTP server. This allows us to interact with the server and perform file transfers. There are several options available for FTP clients, such as FileZilla, WinSCP, or the command-line FTP client. For our FTP traffic measurements, we choose a file size of 344 MB. This file serves as the basis for our evaluation. Additionally, to validate the scalability of our approach, we repeat the measurements using a larger file size of 540 MB.
To evaluate the performance of different scheduling algorithms, we utilize the sysctl command-line tool. By using the sysctl -a command, we can view and modify various kernel parameters.

Specifically, we have applied sysctl -a"net.mptcp.mptcp_scheduler= "command line , which allows us to select a specific scheduler algorithm. We iterate through the list of previously mentioned scheduling

Furthermore, to ensure thorough monitoring and analysis of the network traffic, we configure a mirroring port on the FastEthernet interface of the router. This mirroring port duplicates all the traffic passing through the interface and redirects it to a monitoring station. By capturing the mirrored traffic, we gain valuable insights into the network behavior, enabling us to closely examine the performance and effectiveness of the MPTCP system.

3.5 EXPERIMENTS SCENARIOS

3.5.1 First scenario

In the first scenario, the objective was to comprehensively evaluate five Multi-Path TCP schedulers, including Round Robin, Low RTT First, Delay-Aware Packet Scheduling, Out-of-Order Transmission for In-Order Arrival Scheduler, and Block Estimation scheduler, from an algorithmic perspective. The goal was to assess their performance and effectiveness in handling MPTCP traffic under various network conditions. To ensure the reliability and robustness of the findings, each experiment was repeated ten times for each algorithm, providing a solid basis for statistical analysis and reducing the impact of random variations.

Scenario	Schedulers	Configuration	Size file
First	RR LRF DAPS1 OTIAS BLEST	Sf 1 (BW = 1 Mb/s ; Delay = 15 ms) Sf 2 (BW = 2 Mb/s ; Delay = 12 ms) Sf3 (BW = 5 Mb/s ; Delay = 2 ms) Sf 4 (BW = 10 Mb/s ; Delay = 1 ms) (10 runs for each approach)	Data = 344 Mb

Table 3.2: The first scenario description and subflow parameters.

The experiments were conducted within a controlled test bed environment, utilizing the capabilities of Linux Traffic Control (TC) to configure and simulate different network scenarios. This allowed precise adjustments of bandwidth and delay parameters in each subflow, closely resembling real-world network conditions. The meticulous design of the test bed setup aimed to provide a comprehensive and accurate assessment of the schedulers' capabilities.

Scenario	Schedulers	Configuration	Size file
First	OTIAS-BLEST-OTIAS	Sf1 (BW = 1 Mb/s ; Delay = 15 ms)	Data = 344 Mb
Alternating	BLEST-OTIAS-BLEST	Sf2 (BW = 2 Mb/s ; Delay = 12 ms)	
case	LRF-BLEST-LRF	Sf3 (BW = 5 Mb/s ; Delay = 2 ms)	
	BLEST-LRF-BLEST	Sf4 (BW = 10 Mb/s ; Delay = 1 ms)	

Table 3.3: The second scenario - Part 1 : Without cross-traffic.

During the experiments, specific performance metrics were considered to evaluate the efficacy of the schedulers. These metrics included throughput, bandwidth utilization, loss ratio, and processing time, collectively offering a holistic view of the schedulers' performance characteristics. Throughput measurements were taken for individual subflows and aggregated across all subflows.

Bandwidth utilization assessed the efficiency of each scheduler in utilizing network resources, while the loss ratio metric quantified the proportion of lost packets during transmission, providing insights into resilience and reliability. Additionally, the processing time metric evaluated the efficiency of each scheduler in packet processing and scheduling within the MPTCP framework.The whole test process is illustrated in Fig. 3.4.

To record and analyze the MPTCP traffic generated during the experiments, Wireshark, a widely-used network protocol analyzer, was employed. Carefully crafted filters were applied to isolate and capture the MPTCP dataset traffic, facilitating in-depth analysis of packet traces. This packet-level analysis provided valuable insights into the behavior of the schedulers, shedding light on performance characteristics, patterns, and anomalies.

For the presentation and interpretation of the experimental results, a Python program utilizing the Pandas and Seaborn libraries was employed. This program generated informative graphs and visual representations, derived from the captured data. The standardized visualization approach ensured clarity and consistency, allowing for effective comparisons and meaningful interpretations of the experimental results. Furthermore, crucial network parameters and configurations employed in the evaluations were included in following Tables These tables served as a reference, providing detailed information for readers to gain a deeper understanding of the experimental setup. Tables contained network parameters such as bandwidth, delay parameters for each subflow, and the amount of data used, enhancing the transparency and reproducibility of the experiments.

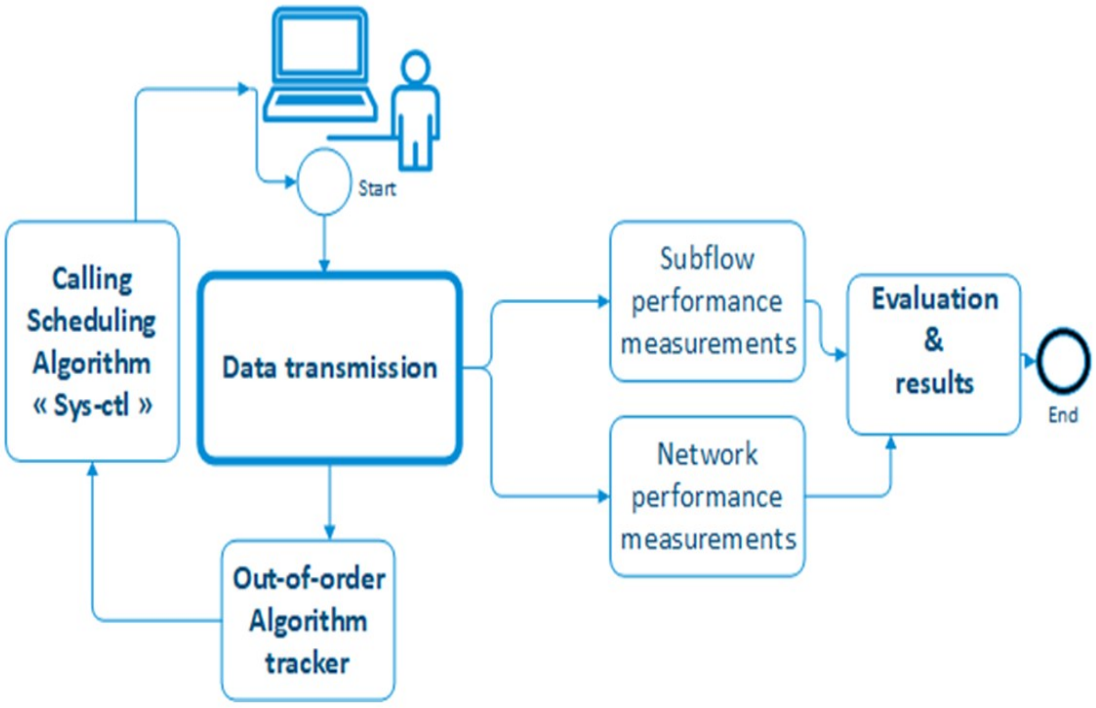

Figure 3.4: The Testbed Workflow.

3.6 The second scenario:

The second scenario is divided into three distinct parts, each focusing on specific aspects of our study. These parts will be described in sequential order, providing detailed information about the objectives and procedures of each part

3.6.1 Part 1: Alternating Case Evaluation

In the first part of the second scenario, we focus on evaluating our proposed contribution, the "Alternating" case. This part builds upon the previous configuration established in the initial scenario. Within this part, we test four different strategies without the presence of cross traffic initially. The selected strategies include LRF (Low RTT First) as a reactive scheduler,

as well as two proactive schedulers, OTIAS (Out-of-Order Transmission for In-Order Arrival Scheduler) and BLEST (Block Estimation Scheduler).
To support the alternation process, we implement an algorithm specifically designed to detect the out-of-order phenomenon, which plays a vital role in the proposed contribution. The objective of this part is to thoroughly assess the performance and effectiveness of the schedulers under varying network conditions.
Throughout the experiments in this part, we conduct multiple runs (ten times for each strategy) to ensure statistical significance and reliability of the results. The data size utilized for these experiments is set to 13 Mb, allowing us to analyze the behavior of the schedulers and their ability to handle MPTCP traffic efficiently.

Scenario	Schedulers	Configuration	Size file
Second	OTIAS-BLEST-OTIAS	Sf1 (BW = 1 Mb/s ; Delay = 15 ms)	Data = 344 Mb
Part1:	BLEST-OTIAS-BLEST	Sf2 (BW = 2 Mb/s ; Delay = 12 ms)	CT=13 Mb
	LRF-BLEST-LRF	Sf3 (BW = 5 Mb/s ; Delay = 2 ms)	
	BLEST-LRF-BLEST	Sf4 (BW = 10 Mb/s ; Delay = 1 ms)	

Table 3.4: The second scenario - Part 1 : With cross-traffic.

3.6.2 Part 2: Confirmation and Scaling

The second part of the scenario focuses on further confirming the performance of the unique and alternating strategies proposed in Part 1. We repeat the experiments while introducing cross traffic into the network. This additional traffic enables us to evaluate the performance of both strategies in the presence of competing traffic patterns.

To validate the scalability and effectiveness of the schedulers, we increase the data size to 540 Mb for these experiments. This larger data size provides more demanding conditions and allows us to assess the schedulers' adaptability and efficiency in handling increased traffic loads.

3.6.3 Part 3: Subflow 4 Parameter Variation

In the final part of the scenario, our aim is to validate our proposed contribution and investigate the behavior of the schedulers in a modified network configuration. Specifically, we focus on changing the parameters of Subflow 4 to gain insights into the impact of these

Scenario	Schedulers	Configuration	Size file
Second: Part 2	RR LRF DAPS1 OTIAS BLEST OTIAS-BLEST-OTIAS BLEST-OTIAS-BLEST LRF-BLEST-LRF BLEST-LRF-BLEST	Sf1 (BW = 1 Mb/s ; Delay = 15 ms) Sf2 (BW = 2 Mb/s ; Delay = 12 ms) Sf3 (BW = 5 Mb/s ; Delay = 2 ms) Sf4 (BW = 10 Mb/s ; Delay = 1 ms)	Data = 344 Mb CT=13 Mb

Table 3.5: The second scenario - Part 2 : Validation 1.

modifications on the performance of the schedulers. We assign the fourth subflow with 10 Mb/s a delay of 140 ms to examine profoundly the alternation strategy behaviour. We use the Linux TC by applying the following command:
sudo tc qdisc add dev eth4 root netem delay 140ms
Similar to the previous parts, we conduct multiple runs (ten times) to ensure accurate and reliable results. By examining the behavior of the schedulers under different parameter settings, we can gain a deeper understanding of their adaptability and further validate the effectiveness of our proposed contribution.
Throughout all parts of the scenario, we ensure the accuracy and precision of our results by disabling all background traffic in the network. This step minimizes any potential interference or noise that could affect the performance measurements, thus enhancing the accuracy and reliability of our findings. The experimental steps and procedures for each part are detailed in Tables " 3.4, 3.5 and 3.6", providing a clear reference and guidance for the execution of the experiments within each part of the scenario.

Scenario	Schedulers	Configuration	Size file
Second Part 3	OTIAS-BLEST-OTIAS BLEST-OTIAS-BLEST LRF-BLEST-LRF BLEST-LRF-BLEST	Sf1 (BW = 1 Mb/s ; Delay = 15 ms) Sf2 (BW = 2 Mb/s ; Delay = 12 ms) Sf3 (BW = 5 Mb/s ; Delay = 2 ms) Sf4 (BW = 10 Mb/s ; Delay = 140 ms)	Data = 540 Mb

Table 3.6: The second scenario - Part 3 : Validation 2.

3.7 Conclusion

Dans ce chapitre, nous avons présenté notre méthode proposée en détaillant les implémentations réalisées. Nous avons ensuite procédé à la phase expérimentale, qui s'est déroulée en deux scénarios avec un ordonnancement logique, dans le but de prouver et de constater les résultats afin de les valider. Nous avons mené ces scénarios dans différents contextes. Les expériences effectuées dans cette partie ont été réalisées en effectuant plusieurs exécutions pour assurer la signification statistique et la fiabilité des résultats. nous permettant ainsi d'analyser le comportement des ordonnanceurs et leur capacité à gérer efficacement le trafic MPTCP. Les résultats de ces expériences seront discutés en détail dans le chapitre suivant, où nous examinerons et analyserons les résultats afin de valider l'approche proposée. Cette analyse fournira des informations précieuses sur les performances et l'efficacité des ordonnanceurs mis en œuvre, en mettant en évidence leurs capacités ainsi que les éventuelles pistes d'amélioration. Dans l'ensemble, les expériences menées dans ce chapitre constituent une étape cruciale pour valider l'approche proposée. Cette analyse fournira des informations précieuses sur les performances et l'efficacité des ordonnanceurs mis en œuvre, en mettant en évidence leurs capacités ainsi que les éventuelles pistes d'amélioration.

Chapter 4

Discussions and Results

4.1 Introduction

The results presented in this section come from our experiments carried out on a testbed, in accordance with the schedule defined in the methodology of our study. The scenarios set up made it possible to evaluate the impact of each scheduler in an asymmetric multihoming network, taking into account both the head-of-line phenomenon and the exploitation of the available bandwidth. In order to analyse the behaviour of each scheduling strategy in detail, we carried out throughput measurements at two different levels. Firstly, throughput measurements were made at the level of each individual subflow. This approach allowed us to evaluate the efficiency of each scheduler in terms of the specific throughput of each subflow, while taking into account the particular characteristics of each of them. Subsequently, throughput measurements were taken at the level of the total aggregation throughput, encompassing all the subflows. These global measurements gave us an overall view of the performance of each scheduler in the network. The figures and tables below illustrate the results of these throughput measurements, providing a clear and comparative overview of the performance of the different schedulers in our asymmetric network scenario. However, to reinforce the relevance of our findings, we also applied several experimental variations in terms of network conditions. In particular, we introduced cross-traffic, simulating situations where the network is subjected to additional load from other data traffic. In addition, we have modified the volume of data transmitted and the characteristics of the subflows to assess the ability of the schedulers to adapt to different conditions and maintain optimal performance. Finally, we also considered the loss ratio recorded during each test, in order to evaluate the reliability of the schedulers in realistic scenarios. Alongside the throughput and loss measurements, we also assessed the transfer time of data. This metric plays a crucial role in highlighting the temporal efficiency of our investigative approach.

4.2 Results and discussion:

The results are reported from our testbed experiments following the timeline announced in the methodology. The scenarios show the impact of each scheduler in an asymmetric multihoming network taking into account the Head-of-Line phenomenon and the exploitation of the bandwidth offered by the network. In order to know the behavior of each scheduling strategy, throughput measurements are performed at each subflow level, then at the total aggregation throughput level as shown in figures underneath; this is followed by a comparative presentation of the chosen algorithms which projects on the percentage of maximum

bandwidth use, the number of out-of-order segment and loss rate marked in each test.

4.3 The first scenario

Firstly, we concentrate in this part of the discussion on the naive approach (RR), the reactive approach (LRF and DAPS), and the proactive approach (OTIAS and BLEST), on the basis of three measures: the throughput, the bandwidth utilization ratio and the number of out-of-order packets recorded in each test.

4.3.1 Test 1: RR case

Fig. 5 shows the amount of data sent in bits per second. When looking at the throughput variation in the case of the RR scheduler, this strategy behaves initially by an increasing throughput ramp-up period until the 16th second, corresponding to a fall value. Thereafter, it passes to an instability period starting from the second subflow and which is reflected unfavorably in the total aggregation throughput. RR is lacking the ability to work with heterogeneous networks because it rarely takes into account the characteristics of the path, which leads to the appearance of 'out-of-order' packets. This was also claimed in [9].

4.3.2 Test 2: LRF case

In Contrary to the naive scheduler (RR), LRF responds more to the distinctive characteristics of paths; the obtained results have shown an important throughput stabilization during transmission time precisely in the first and the second subflows which were too close to the bandwidth values of respectively 1 Mb/s and 2 Mb/s. Moreover, LRF does not mark the intersections after the scale-up period as shown in Fig. 6. In fact, this is the main objective that seeks to maximize the throughput in the asymmetric multihoming networks as stated in [9] and which has been clearly evidenced by our obtained results.

4.3.3 Test 3: DAPS1 case

Although the DAPS scheduler is dedicated to asymmetric multihoming networks, the results found in our test illustrated in Fig. 7 show that just after the initialization of the first subflow, DAPS causes a sudden failure in terms of throughput delivered, the latter is characterized by a rather high instability of variation in all subflows. This finding is due to the number

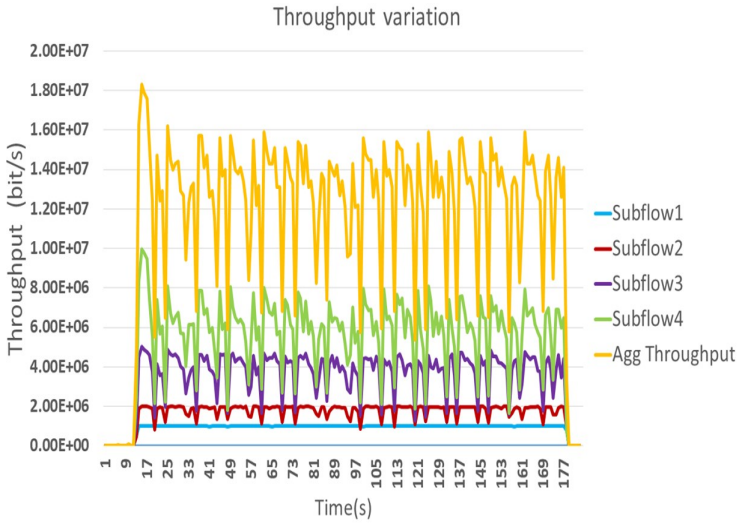

Figure 4.1: Round Robin scheduler.

Figure 4.2: Low RTT First scheduler.

of subflows as demonstrated in [10], It is to note that we have used the first implementation version of DAPS in the tested project even though an improved version named DAPS2 [11]has made its appearance but unfortunately this latter is not available in the kernel of this project [31]. After 130 seconds, another observation in this experimentation shows that the throughput offered in the 2nd subflow with a maximum capacity of 2 Mb/s gives a throughput

lower than the one offered by the first one with a maximum capacity of 1 Mb/s; the same behaviour is noted between the 3rd and the 4th subflow.

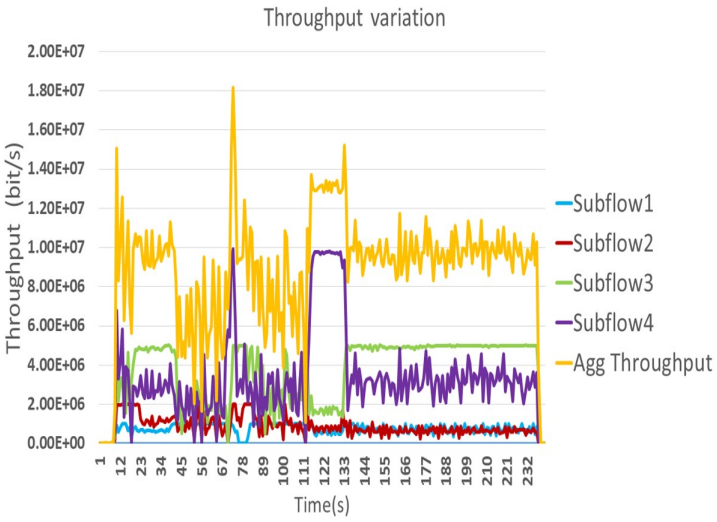

Figure 4.3: DAPS scheduler.

4.3.4 Test 4: OTIAS case

The test with the OTIAS scheduler in Fig. 8 has shown a stability of throughput in the initial subflow. However, the throughput in the second subflow has decreased after 20 seconds of testing, this is because OTIAS favors subflows with high bandwidth, namely subflows 3 and 4. When dealing with asymmetry, OTIAS takes into account the situation of the network , it can be interpreted by the ranking assigned to the best performing subflows. Looking at the right part of the graph, the greatest advantage of the OTIAS scheduler is that it outputs the highest throughput in the subflows characterized by high bandwidth, as indicated in the 5 Mb/s third subflow. Moreover, it attains the maximum data sent through the fourth subflow with of 10 Mb/s. In overall, OTIAS reaches a satisfactory level of offered throughput.

4.3.5 Test 5: BLEST case

With the test results of presented in Fig. 9 showing the behaviour of the BLEST algorithm, we can say that the throughput reaches maximum values precisely in subflows 1, 2 and 3 which gave us a better average throughput. This appears in the graphs which respect

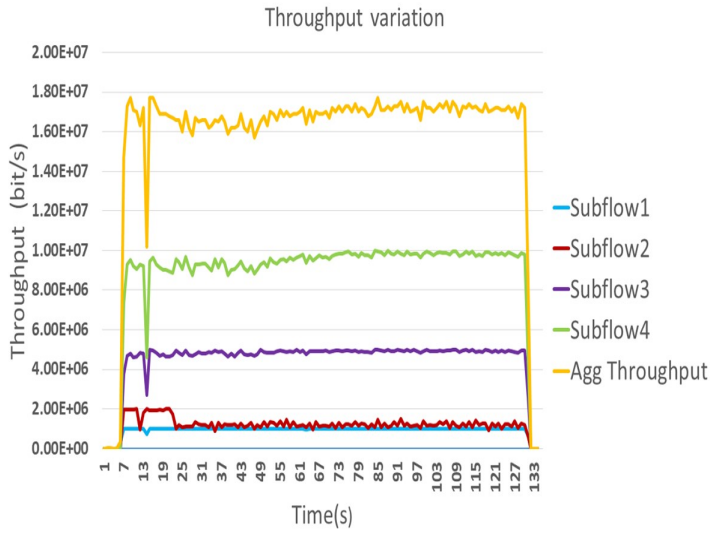

Figure 4.4: OTIAS scheduler.

on the one hand a condition of extreme stability and on the other hand a good quality of transmission.It is to note that there is indeed, some less stabilization in the fourth subflow which also shows performances but with a lesser degree compared to the first three subflows. In overall, the results prove that BLEST is a generic solution in our state of research, the benefit is observed on a large scale in subflow graphs that show the absence of intersections between them. As of the stability of the graph, the value of the minimum throughput in subflow ‘ i ’ is greater than the value of the maximum throughput in subflow ‘i-1’, and thus BLEST has reached the expected desired results [13]; expressed by an impressive aggregation of the through puts as demonstrated in this work.

4.4 The bandwidth utilization in the unique case:

The Fig. 4.6 highlights the bandwidth utilization on the subflow level. An investigation is conducted based on an average of 10 runs for each scheduler. The results show that LRF outperforms all approaches. LRF has reached a significant bandwidth utilization followed first by BLEST and then by OTIAS. An exception is however noted in the first subflow where OTIAS comes second. In fact, BLEST comes at the third place since it ignores the low bandwidth subflow in order to reduce the potential of HoL-blocking. Finally, RR and DAPS have behaved poorly compared to the other approaches.

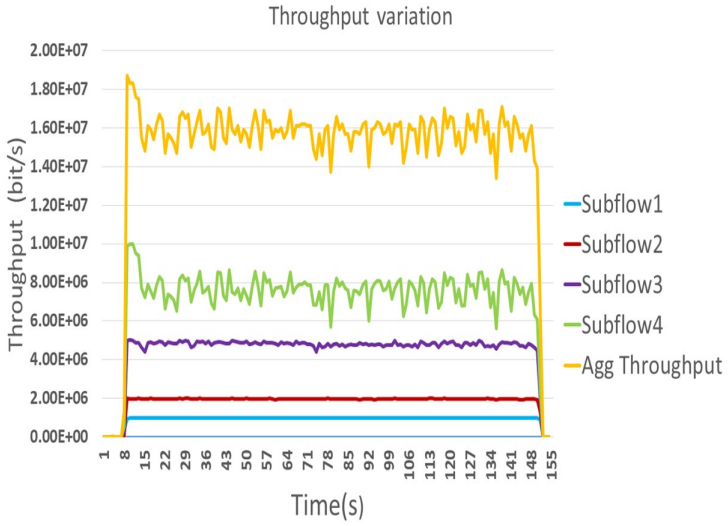

Figure 4.5: BLEST scheduler.

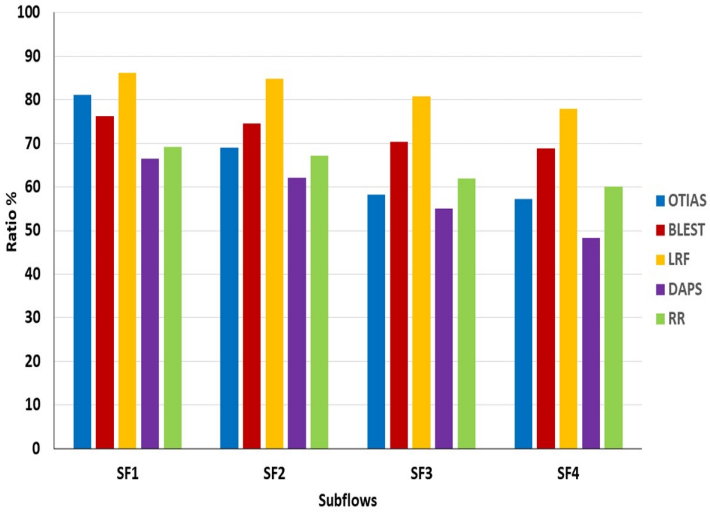

Figure 4.6: Subflows bandwidth utilization ' unique case'.

4.5 First scenario summury

For a general summary, Table 4.1 reveals that the LRF scheduler offers a better overall throughput value that reaches 15.28 Mb/s and responds to a bandwidth utilization close to 81 %İt is therefore considered to be the best scheduler of the reactive approach for all assessment parameters considered in our study, followed by the BLEST scheduler, and then

OTIAS, which also shows acceptable results that led us to adopt it for the next scenario, as well as its compatibility with the other studies [28-29].

Scheduler	Average throughput aggregation (b/s)	Global bandwidth utilization %
RR	10806282	57.2537
LRF	15286766	80.9921
DAPS	7975296	42.2546
OTIAS	12546502	66.4737
BLEST	14884588	78.8613

Table 4.1: The first scenario : Global statistics.

4.6 The second scenario

In the case of invoking 'OTIAS-BLEST-OTIAS' to alternate with this sequencing, as shown in Fig. 4.7 the global throughput exhibits such a high instability, especially in time interval [20;100], which represents a delicate period in the data transmission. After that, it starts to improve except the 2nd subflow that has poorly exploited its proper bandwidth. Indeed, the change with the strategy that calls the order 'BLEST-OTIAS-BLEST' shown in Fig. 4.8 results in a completely different behaviour compared to the previous test.The variation is now more stable in the first two subflows, which reflects well on the whole throughput.The latest tests which are intended to alternate the reactive and proactive approaches take into consideration the order of the scheduler that we will call it first.

The Fig. 4.9 : represents the result which puts the reactive scheduling as a starting point in this test, named 'LRF-BLEST-LRF'. This interaction of schedulers records better results marked by a good stability precisely seen in the first three subflows with a significant throughput towards the capacity. The same behavior that has been seen previously and with the goal to change the starting priority to the proactive scheduler BLEST as opposed to the third test has been obtained.

The results of the test shown in Fig. 4.10 named 'BLEST-LRF-BLEST' proves also that this alternation between the proactive and reactive approaches gives a good level of throughput aggregation.

The gain is widely observed from the aggregate throughput and is witnessed in the last three subflows where it appears that there is no intersection.

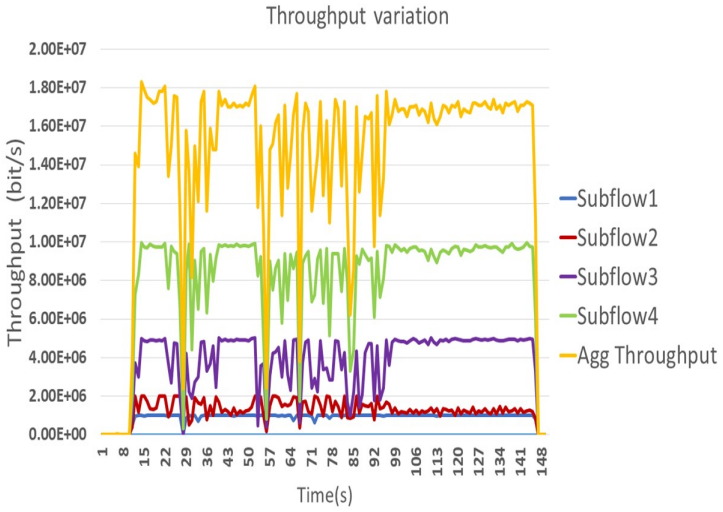

Figure 4.7: OTIAS-BLEST-OTIAS.

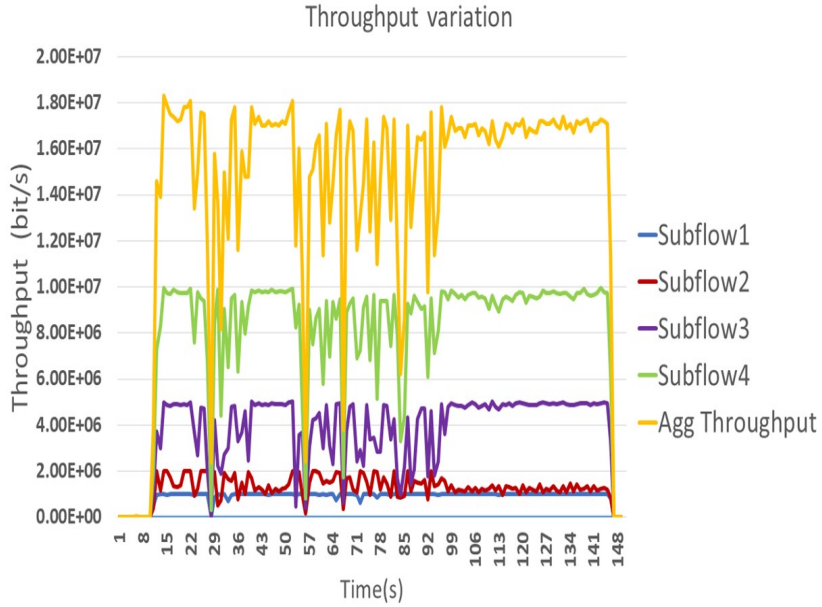

Figure 4.8: BLEST-OTIAS-BLEST.

4.7 Bandwidth Utilization in the Alternating Case

The bandwidth utilization in the alternating case was evaluated to assess the performance of our proposed alternation approach. As shown in Figure 4.1, the obtained results clearly

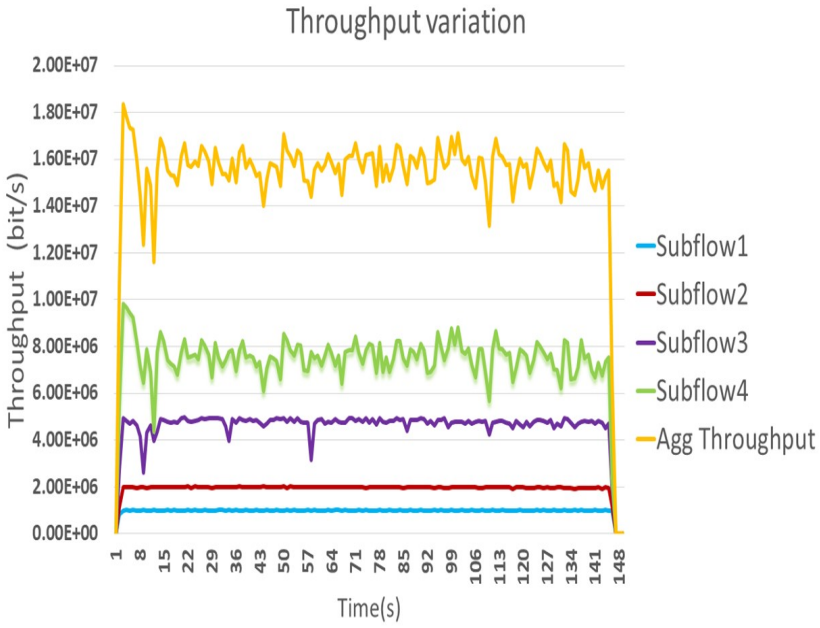

Figure 4.9: LRF-BLEST-LRF.

Figure 4.10: BLEST-LRF-BLEST.

demonstrate that the bandwidth utilization in all subflows is significantly improved compared to the unique case. Notably, the 'LRF-BLEST-LRF' strategy exhibited the highest

bandwidth utilization ratios, with values of 90.94% 86.32% 81.76% and 80.32% for subflows 1, 2, 3, and 4, respectively.

Comparing these results with those of the first scenario, the benefits of our proposed approach become evident. The summarized data in Table 4.2 further emphasize the advantages of our contribution, which is based on a novel MPTCP scheduling aspect. Specifically, the 'LRF-BLEST-LRF' strategy demonstrated reliable and efficient utilization of bandwidth at the subflow level, resulting in a global bandwidth utilization ratio of 90.02% and an average throughput aggregation of 16.99 Mb/s.

These findings indicate that the 'LRF-BLEST-LRF' strategy outperforms the other scheduling strategies evaluated in the second scenario. Its ability to effectively utilize available bandwidth and achieve high throughput aggregation makes it the preferred choice for scheduling in MPTCP environments. The global statistics presented in Table 4.4 further support this conclusion, as the 'LRF-BLEST-LRF' strategy achieves the highest average aggregation of 16,991,124 b/s and the highest global bandwidth utilization ratio of 90.0222%

In summary, the results highlight the significant improvements in bandwidth utilization achieved through our alternation approach. The 'LRF-BLEST-LRF' strategy proves to be superior in effectively utilizing bandwidth at the subflow level, leading to higher overall throughput and improved performance compared to other scheduling strategies. These findings underscore the effectiveness of our proposed approach in optimizing MPTCP data transmission and recommend its adoption for efficient scheduling in real-world network scenarios.

Scheduler	Average aggregation (b/s)	Global bandwidth utilization %
OTIAS-BLEST-OTIAS	13482491	71.4328
BLEST-OTIAS-BLEST	15466356	81.9437
LRF-BLEST-LRF	16991124	90.0222
BLEST-LRF-BLEST	16442370	87.1148

Table 4.2: The second scenario part 1 : Global statistics.

4.8 Cross-traffic analysis :

The figure 4.11 presented here aims to investigate and provide evidence for the validity of our approach. By generating a cross traffic while maintaining the same traffic pattern as in

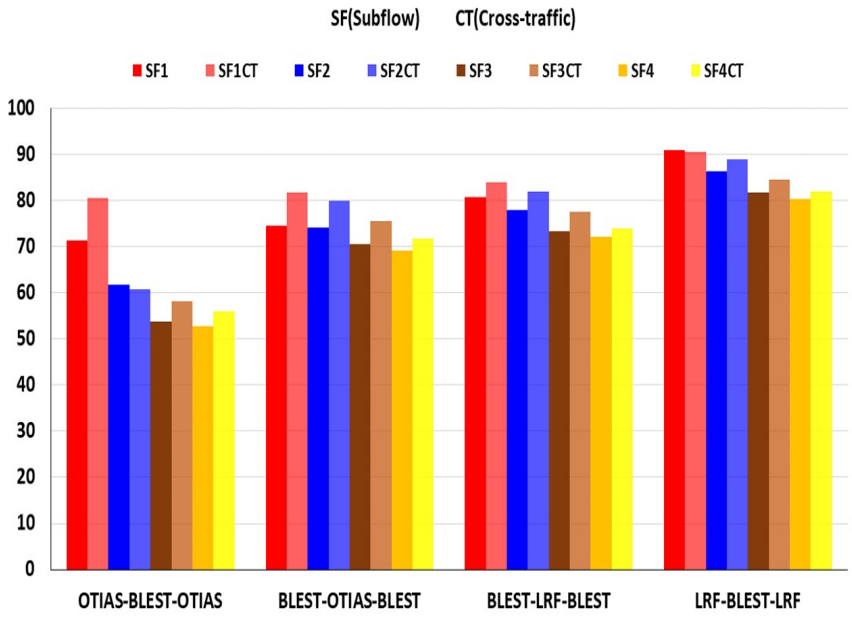

Figure 4.11: Subflows bandwidth utilization 'alternating case.

scenario 1, we can observe the performance of our proposed approach, namely "LRF-BLEST-LRF." The results clearly demonstrate its effectiveness, particularly in terms of bandwidth utilization compared to other scheduling approaches. In this test, the utilization of the available bandwidth is analyzed across all the scheduling approaches under consideration. The outcomes reveal that our approach, despite the presence of foreground traffic in this specific context, demonstrates its validity and efficiency. The findings highlight the superiority of "LRF-BLEST-LRF" in effectively managing the competing cross traffic while maintaining the desired performance for the foreground traffic. These results further validate the effectiveness of our approach in handling different traffic scenarios and its ability to adapt to challenging network conditions. The successful management of cross traffic, along with the sustained performance of the foreground traffic, reinforces the robustness and practical applicability of our proposed approach.

4.9 TCP loss ratio

In term of loss-rate (a major performance criteria), the outcomes presented in Table 4.3 show that this parameter is below 0.15% for all tests. Moreover, and when comparing these schedulers, the approach named 'LRF-BLEST-LRF' has resulted in the lowest loss-rate of

0.102 % followed by "BLEST-LRF-BLEST" with 0.106 % and then BLEST, OTIAS and LRF, contrarily to the Daps and RR schedulers which have registered a high loss-rate. These findings confirm indeed the previous comparisons conducted in this work and confirm also that the obtained results are better than those of the work in [36], thus validating our proposed 'LRF-BLEST-LRF' scheduler which offers a greater bandwidth aggregation and a lower loss rate.

4.10 Approach evaluation bulk traffic case:

In Table 4.4 The LRF-BLEST-LRF approach stands out as the most favorable approach among the tested strategies. It achieves the highest bandwidth utilization at the subflow level and overall total utilization when considering a file size of 540 Mb. Additionally, it exhibits the lowest processing time, indicating efficient packet processing and scheduling within the MPTCP framework. These results validate the effectiveness of our proposed approach in optimizing network performance and resource utilization. By combining the reactive nature of LRF and the proactive nature of BLEST, followed by another LRF stage, we have successfully achieved superior performance in terms of both bandwidth utilization and processing time. This highlights the significance of our approach in improving the efficiency and performance of Multi-Path TCP scheduling algorithms.

Scheduler	Loss(SF1)	Loss(SF2)	Loss(SF3)	Loss(SF4)	Total loss	Loss%
BLEST	111	105	65	9	290	0.1080
OTIAS	108	96	78	10	292	0.1087
LRF	110	99	76	8	293	0.1091
DAPS	104	94	90	39	327	0.1218
RR	132	125	81	53	391	0.1456
OTIAS-BLEST-OTIAS	107	94	75	35	311	0.1158
BLEST-OTIAS-BLEST	119	100	67	15	301	0.1121
BLEST-LRF-BLEST	115	101	63	8	287	0.1069
LRF-BLEST-LRF	112	96	61	7	276	0.1028

Table 4.3: Global TCP Loss segment statistics.

Scheduler	SF1 (%)	SF2 (%)	SF3 (%)	SF4 (%)	Total utilization	Time (s)
BLEST	94.61	88.67	82.97	86.52	88.82	292.097
OTIAS	90.83	65.28	73.96	71.84	74.67	316.227
LRF	98.35	94.8	89.98	85.67	90.78	252.875
DAPS	69.71	48.60	87.37	27.86	50.42	432.113
RR	70.57	67.30	63.11	65.58	67.33	358.084
OTIAS-BLEST-OTIAS	96.36	73.46	70.96	71.67	74.74	336.099
BLEST-OTIAS-BLEST	94.53	93.92	89.39	84.10	87.24	254.172
BLEST-LRF-BLEST	94.97	93.57	88.91	83.19	88.88	252.074
LRF-BLEST-LRF	97.48	96.06	91.71	87.69	92.65	245.710

Table 4.4: Validation 1 "Bandwidth utilization and processing time".

4.11 Approach validation: Network conditions

Table 4.5 presents the results of Validation 2, specifically focusing on the scenario with a data size (D) of 540 Mb and a delay of 140 ms for subflow 4 (SF4). This validation aimed to assess the performance of different schedulers under varying network conditions. Among the evaluated approaches, our proposed LRF-BLEST-LRF scheduler demonstrated exceptional performance, exhibiting both high bandwidth utilization and low processing time across different network conditions.

Under the specified conditions, LRF-BLEST-LRF achieved a remarkable bandwidth utilization of 98.39 % for SF1, 97.39% for SF2, 92.31% for SF3, and 49.64% for SF4. These utilization percentages indicate the efficiency of our approach in effectively utilizing the available network resources for data transmission. Furthermore, the total utilization, which combines the contributions from all subflows, reached an impressive 69.84%

In terms of processing time, LRF-BLEST-LRF demonstrated its superiority by achieving a significantly reduced processing time of 455.471 seconds. This indicates the efficiency and effectiveness of our scheduler in handling and processing data within the specified network conditions.

Overall, the results from Validation 2 highlight the robustness and reliability of our proposed LRF-BLEST-LRF scheduler. It consistently delivers high bandwidth utilization and low processing time, even when subjected to changes in network conditions. These findings reinforce the effectiveness of our approach and its potential for optimizing data transmission in real-world network environments.

Scheduler	SF1 (%)	SF2 (%)	SF3 (%)	SF4 (%)	Total utilization	Time (s)
OTIAS-BLEST-OTIAS	88.14	73.79	72.94	32.99	53.14	602.099
BLEST-OTIAS-BLEST	97.21	95.67	91.36	31.02	60.23	594.091
BLEST-LRF-BLEST	97.32	95.85	90.62	49.02	67.54	504.085
LRF-BLEST-LRF	98.39	97.39	92.31	49.64	69.84	455.471

Table 4.5: Validation 2 "D = 540 Mb and SF4 delay = 140 ms".

4.12 Conclusion

The results and discussions presented in this chapter demonstrate the suitability of the proposed approach for alternating between reactive and proactive planning in MPTCP networks. Evaluations and measurements have shown that the LRF-BLEST-LRF algorithm is reliable and outperforms existing algorithms in terms of various performance criteria.

Bandwidth utilization and aggregation in the proposed approach showed remarkable improvements, leading to better network performance. The LRF-BLEST-LRF scheme effectively balances load distribution, maximizes resource utilization and ensures efficient data transmission in MPTCP networks.

Moreover, the validation of our approach under changing network conditions, such as varying data patterns and available resources, further strengthens the insight and significance of our research findings. The adaptability and performance of the LRF-BLEST-LRF algorithm in such dynamic environments highlights its effectiveness and potential to meet the evolving demands of modern networks.

All in all, the results presented in this chapter provide convincing evidence of the relevance and effectiveness of the proposed approach. The LRF-BLEST-LRF algorithm offers significant improvements in terms of reliability, bandwidth utilization and overall network performance over existing scheduling algorithms. These results contribute to advancing the field of MPTCP scheduling, and provide valuable information for researchers and network administrators seeking to optimize performance and resource handling in MPTCP networks.

Bibliography

Bibliography

[1] See Kelly, Tim (2006) The 4A vision: anytime, anywhere, by anyone and anything", presentation at ITAHK luncheon, 8 December 2005, URL:http://www.cahk.hk/Event/30/images/Luncheon_Dec2005

[2] C. Sunshine, Specification Of Internet Transmission Control Program" pp. 1–70, 1974 URL: https://tools.ietf.org/pdf/rfc675.

[3] Hyo-JeongShin ; Dan Pei; M. Lad; Yanghee Choi "The impact of multi-homing on network reliability and stability, Computer Communications and Networks Conf, 2005.

[4] A. Ford, C. Raiciu, M. Handley, S. Barre, and J. Iyengar, "Architecturalguidelines for multipath TCP development," Internet Engineering Task Force RFC 6182, 2011.

[5] S. Barré, C. Paasch, and O. Bonaventure, "Multipath TCP: From theory to practice," in NETWORKING, J. Domingo-Pascual, P. Manzoni, S. Palazzo, A. Pont, and C. Scoglio, Eds. Berlin, Germany: Springer, 2011, pp. 444–457.

[6] O. Bonaventure, M. Handley, and C. Raiciu, An overview of multipath TCP, *Usenix Login*, vol. 37, no. 5, pp. 17—22, 2012.

[7] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, TCP Extensions for Multipath Operation With Multiple Addresses, document RFC 6824, Jan. 2013.

[8] C. Paasch and S. Barre, "Multipath TCP in the Linux kernel," 2013. [Online]. URL: https:// www.multipath-tcp.org

[9] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental evaluation of multipath TCP schedulers," in Proc. ACM SIGCOMMCapacity Sharing Workshop (CSWS), 2014, pp. 27–32

[10] G. Sarwar, R. Boreli, E. Lochin, A. Mifdaoui, and G. Smith, "Mitigating receiver's buffer blocking by delay aware packet scheduling in multipath data transfer," in Proc. Int. Conf. Adv. Inf. Netw. Appl. Workshops(WAINA), Barcelona, Spain, Mar. 2013, pp. 1119–1124.

[11] N. Kuhn, E. Lochin, A. Mifdaoui, G. Sarwar, and O. Mehani, "DAPS: Intelligent delay-aware packet scheduling for multipath transport," in Proc. IEEE Int. Conf. Commun. (ICC), Sydney, NSW, Australia, Jun. 2014, pp. 1222–1227.

[12] F. Yang, Q. Wang, and P. D. Amer, "Out-of-order transmission for in-order arrival scheduling for multipath TCP," in Proc. Int. Conf. Adv.Inf. Netw. Appl. Workshop (WAINA), Victoria, BC, Canada, May 2014, pp.749–752.

[13] S. Ferlin, Ö. Alay, O. Mehani, and R. Boreli, "BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks," in Proc. IFIP Netw. Conf. Workshops, Vienna, Austria, May 2016, pp. 431–439.

[14] K. Wang et al., "On the Path Management of Multi-Path TCP in Internet Scenarios based on the NORNET Testbed." 2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA) Path manager.Conf

[15] C. Raiciu, D. Wischik, and M. Handley, "Practical congestion control for multipath transport protocols," Dept. Comput. Sci., Univ. College London, London, U.K., Tech. Rep., Nov. 2009.

[16] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, "Design, implementation and evaluation of congestion control for multipath TCP," in Proc. 8th USENIX Conf. Networked Syst. Design Implement. (NSDI), Boston, MA, USA, pp. 99–112, 2011.

[17] F. Fu, X. Zhou, T. Dreibholz, K. Wang, F. Zhou, and Q. Gan, "Performance Comparison of Congestion Control Strategies for Multi-Path TCP in the NORNET Testbed." 2015 IEEE/CIC International Conference on Communications in China (ICCC).Conf ;Year: 2015.

[18] Christoph Paasch, O. Bonaventure, A generic control stream for Multipath TCP, February 2014, Internet draft, URL: https://datatracker.ietf.org/doc/draft-paasch-mptcp-control-stream/

[19] C. Raiciu, "Multipath TCP Signalling Types of Control Information used by Multipath TCP," pp. 1–5, 2010.

[20] P. D. Amer, M. Becke, T. Dreibholz, N. Ekiz, J. R. Iyengar, P. Natarajan,R. R. Stewart, and M. T uxen, "Load Sharing for the Stream ControlTransmission Protocol (SCTP)," IETF, Network Working Group, InternetDraft Version 06, Mar. 2013, draft-tuexen-tsvwg-sctp-multipath-06.txt, work in progress

[21] R. R. Stewart, "Stream Control Transmission Protocol," IETF, Standards Track RFC 4960, Sept. 2007, ISSN 2070-1721

[22] H. M. A. Hijawi and M. M. N. Hamarsheh, “Performance Analysis Of Multi-Path TCP Network,” vol. 8, no. 2, pp. 145–157, 2016.

[23] M. Sayit, E. Karayer, C. D. Phung, S. Secci, and S. Boumerdassi, “Numerical evaluation of MPTCP schedulers in terms of throughput and reliability,” Proc. 2019 11th Int. Work. Resilient Networks Des. Model. RNDM 2019, pp. 1–6, 2019.

[24] B. Arzani, A. Gurney, S. Cheng, R. Guerin, and B. T. Loo, “Impact of path characteristics and scheduling policies on MPTCP performance,” Proc. - 2014 IEEE 28th Int. Conf. Adv. Inf. Netw. Appl. Work. IEEE WAINA 2014, pp. 743–748, 2014.

[25] N. Ye, P. Dong, G. Duan, J.Wang, Research on mptcp flows scheduling algorithm based on flow characteristics, Journal of Central South University (Science and Technology) (2018) 1691–1699.

[26] B. Y. L. Kimura, D. C. S. F. Lima, and A. A. F. Loureiro, “Alternative scheduling decisions for multipath TCP,” IEEE Commun. Lett., vol. 21, no. 11, pp. 2412–2415, Nov. 2017.

[27] K. W. Choi, Y. S. Cho, Aneta, J. W. Lee, S. M. Cho, and J. Choi, “Optimal load balancing scheduler for MPTCP-based bandwidth aggregation in heterogeneous wireless environments,” Comput. Commun., vol. 112, pp.116–130, 2017.

[28] P. Dong, J. Xie, W. Tang, N. Xiong, H. Zhong, and A. V. Vasilakos, “Performance Evaluation of Multipath TCP Scheduling Algorithms,” IEEE Access, vol. 7, pp. 29818–29825, 2019.

[29] P. Hurtig, K. Grinnemo, A. Brunstrom, S. Ferlin, O. Alay, and N. Kuhn, “Low-latency scheduling MPTCP,” IEEE/ACM Trans. Netw., vol. 27, no. 1, pp. 302–315, Feb. 2019.

[30] https://www.wireshark.org/docs/man-pages/tshark.html

[31] url : https://bitbucket.org/BLEST_mptcp/

[32] T. Ojanperä and J. Vehkaperä, “Network-assisted multipath DASH using the distributed decision engine,” in Proc. Int. Conf. Comput., Netw.Commun. (ICNC), Feb. 2016, pp. 1–6.

[33] D. Jurca and P. Frossard, “Video packet selection and scheduling for multipath streaming,” IEEE Trans. Multimedia, vol. 9, no. 3,629–641, Apr. 2007.

[34] X. Corbillon, R. Aparicio-Pardo, N. Kuhn, G. Texier, and G. Simon, “Cross-layer scheduler for video streaming over MPTCP,” in Proc.7th Int. Conf. Multimedia Syst. (MMSys), Klagenfurt, Austria, May 2016,Art. no. 7.

[35] B. Y. L. Kimura, D. C. S. F. Lima, and A. A. F. Loureiro, "Packet Scheduling in Multipath TCP: Fundamentals, Lessons, and Opportunities," IEEE Syst. J., pp. 1–9, 2020.

[36] K. Xue et al., "DPSAF: Forward Prediction Based Dynamic Packet Scheduling and Adjusting with Feedback for Multipath TCP in Lossy Heterogeneous Networks," IEEE Trans. Veh. Technol., vol. 67, no. 2, pp. 1521–1534, 2018.

[37] S. Abbasloo, C.-Y. Yen, and H. J. Chao, Wanna make your TCP scheme great for cellular networks? let machines do it for you!, IEEE Journal on Selected Areas in Communications, 39 (2020), pp. 265–279.

[38] V. Adarsh, P. Schmitt, and E. Belding, Mptcp performance over heterogenous subpaths, in 2019 28th International Conference on Computer Communication and Networks (ICCCN), IEEE, 2019, pp. 1–9.

[39] E. Ahmed, I. Yaqoob, I. A. T. Hashem, J. Shuja, M. Imran, N. Guizani, and S. T. Bakhsh, Recent advances and challenges in mobile big data, IEEE Communications Magazine, 56 (2018), pp. 102–108.

[40] S. Bae, D. Ban, D. Han, J. Kim, K.-h. Lee, S. Lim, W. Park, and C.-k. Kim, Streetsense: Effect of bus wi-fi aps on pedestrian smartphone, in Proceedings of the 2015 Internet Measurement Conference, 2015, pp. 347–353.

[41] M. Bansal, I. Chana, and S. Clarke, A survey on iot big data: Current status, 13 v's challenges, and future directions, ACM Computing Surveys (CSUR), 53 (2020), pp. 1–59.

[42] F. Yang, P. Amer, and N. Ekiz, "A Scheduler for multipath TCP," in Proc. 22nd Int. Conf. Comput. Commun. Netw., 2013, pp. 1–7 .

[43] E. F. G. M. Beig, P. Daneshjoo, S. Rezaei, A. A. Movassagh, R. Karimi, and Y. Qin, Mptcp throughput enhancement by q-learning for mobile devices, in 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), IEEE, 2018, pp. 1171–1176.

[44] M. Chen, Y. Liu, and S. Mao, Big data: Asurvey mobile network and applications, (2014).

[45] F. Chiariotti, S. Kucera, A. Zanella, and H. Claussen, Analysis and design of a latency control protocol for multi-path data delivery with pre-defined QoS guarantees, IEEE/ACM Transactions on Networking, 27 (2019), pp. 1165–1178.

[46] J. Chung, D. Han, J. Kim, and C.-k. Kim, Machine learning based path management for mobile devices over mptcp, in 2017 IEEE International Conference on Big Data and Smart Computing (BigComp), IEEE, 2017, pp. 206–209.

[47] H. Fang, Z. Zhang, C. J. Wang, M. Daneshmand, C. Wang, and H. Wang, A survey of big data research, IEEE network, 29 (2015), pp. 6–9.

[48] S. Ferlin, Ö. Alay, O. Mehani, and R. Boreli, Blest: Blocking estimation-based MPTCP scheduler for heterogeneous networks, in 2016 IFIP Networking Conference (IFIP Networking) and Workshops, IEEE, 2016, pp. 431–439.

[49] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, and C. Paasch, Rfc 6824: TCP extensions for multipath operation with multiple addresses, Internet Engineering Task Force, (2013).

[50] Y. E. Guo, A. Nikravesh, Z. M. Mao, F. Qian, and S. Sen, Accelerating multipath transport through balanced subflow completion, in Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking, 2017, pp. 141–153.

[51] S. Ha, I. Rhee, and L. Xu, Cubic: a new TCP-friendly high-speed TCP variant, ACM SIGOPS operating systems review, 42 (2008), pp. 64–74.

[52] H. Han, S. Shakkottai, C. Hollot, R. Srikant, and D. Towsley, Overlay TCP for multi-path routing and congestion control, in IMA Workshop on Measurements and Modeling of the Internet, 2004.

[53] M. Honda, Y. Nishida, L. Eggert, P. Sarolahti, and H. Tokuda, Multipath congestion control for shared bottleneck, in Proc. PFLDNeT workshop, vol. 357, Citeseer, 2009, p. 378.

[54] C. Huang, J. Zhang, and T. Huang, Objective-oriented resource pooling in MPTCP: A deep reinforcement learning approach, in 2020 3rd International Conference on Hot Information-Centric Networking (HotICN), IEEE, 2020, pp. 175–181.

[55] H. Karim, IoT networking using mptcp protocol, 2020.

[56] F. Kelly and T. Voice, Stability of end-to-end algorithms for joint routing and rate control, ACM SIGCOMM Computer Communication Review, 35 (2005), pp. 5–12.

[57] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, and J.-Y. Le Boudec, MPTCP is not pareto-optimal: performance issues and a possible solution, in Proceedings of the 8th international conference on Emerging networking experiments and technologies, 2012, pp. 1–12.

[58] N. Kuhn, E. Lochin, A. Mifdaoui, G. Sarwar, O. Mehani, and R. Boreli, Daps: Intelligent delay-aware packet scheduling for multipath transport, in 2014 IEEE International Conference on Communications (ICC), IEEE, 2014, pp. 1222–1227.

[59] Y.-s. Lim, Y.-C. Chen, E. M. Nahum, D. Towsley, and K.-W. Lee, Cross-layer path management in multi-path transport protocol for mobile devices, in IEEE INFOCOM 2014-IEEE Conference on Computer Communications, IEEE, 2014, pp. 1815–1823.

[60] Y.-s. Lim, E. M. Nahum, D. Towsley, and R. J. Gibbens, Ecf: An MPTCP path scheduler to manage heterogeneous paths, in Proceedings of the 13th international conference on emerging networking experiments and technologies, 2017, pp. 147–159.

[61] Y. Liu, A. Neri, A. Ruggeri, and A. M. Vegni, A MPTCP-based network architecture for intelligent train control and traffic management operations, IEEE Transactions on Intelligent Transportation Systems, 18 (2016), pp. 2290–2302.

[62] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, nature, 518 (2015), pp. 529–533.

[63] S. C. Nguyen and T. M. T. Nguyen, Evaluation of multipath TCP load sharing with coupled congestion control option in heterogeneous networks, in Global Information Infrastructure Symposium-GIIS 2011, IEEE, 2011, pp. 1–5.

[64] Q. Peng, A. Walid, J. Hwang, and S. H. Low, Multipath TCP: Analysis, design, and implementation, IEEE/ACM Transactions on networking, 24 (2014), pp. 596–609.

[65] S. R. Pokhrel, Federated learning meets blockchain at 6g edge: A drone-assisted networking for disaster response, in Proceedings of the 2nd ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond, 2020, pp. 49–54.

[66] S. R. Pokhrel and J. Choi, Low-delay scheduling for internet of vehicles: Load-balanced multipath communication with FEC, IEEE Transactions on Communications, 67 (2019), pp. 8489–8501.

[67] , A decentralized federated learning approach for connected autonomous vehicles, in 2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), IEEE, 2020, pp. 1–6.

[68] , Improving TCP performance over wifi for internet of vehicles: A federated learning approach, IEEE Transactions on Vehicular Technology, 69 (2020), pp. 6798–6802.

[69] S. R. Pokhrel, J. Ding, J. Park, O.-S. Park, and J. Choi, Towards enabling critical mmtc: A review of urllc within mmtc, IEEE Access, 8 (2020), pp. 131796–131813.

[70] S. R. Pokhrel and S. Garg, Multipath communication with deep q-network for industry 4.0 automation and orchestration, IEEE Transactions on Industrial Informatics, (2020).

[71] S. R. Pokhrel, J. Jin, and H. Le Vu, Mobility-aware multipath communication for unmanned aerial surveillance systems, IEEE Transactions on Vehicular Technology, 68 (2019), pp. 6088–6098.

[72] S. R. Pokhrel and M. Mandjes, Improving multipath TCP performance over wifi and cellular networks: An analytical approach, IEEE Transactions on Mobile Computing, 18 (2018), pp. 2562–2576.

[73] S. R. Pokhrel, M. Panda, and H. L. Vu, Fair coexistence of regular and multipath TCP over wireless last-miles, IEEE Transactions on Mobile Computing, 18 (2018), pp. 574–587.

[74] S. R. Pokhrel, R. C. Pandey, and S. R. Joshi, Performance modelling and evaluation of V2I video surveillance system, in 2015 Twelfth International Conference on Wireless and Optical Communications Networks (WOCN), IEEE, 2015, pp. 1–4.

[75] S. R. Pokhrel, Y. Qu, and L. Gao, Qos-aware personalized privacy with multipath tcp for industrial iot: Analysis and design, IEEE Internet of Things Journal, 7 (2020), pp. 4849–4861.

[76] S. R. Pokhrel and S. Singh, Compound TCP performance for industry 4.0 wifi: A cognitive federated learning approach, IEEE Transactions on Industrial Informatics, 17 (2020), pp. 2143–2151.

[77] S. R. Pokhrel and C. Williamson, A rent-seeking framework for multipath TCP, (2020).

[78] J. Postel, Rfc0793: Transmission control protocol, 1981.

[79] C. Raiciu, M. Handley, and D. Wischik, Coupled congestion control for multipath transport protocols, tech. rep., IETF RFC 6356, Oct, 2011.

[80] M. Scharf and S. Kiesel, Nxg03-5: Head-of-line blocking in TCP and SCTP: Analysis and measurements, in IEEE Globecom 2006, IEEE, 2006, pp. 1–5.

[81] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, Industrial internet of things: Challenges, opportunities, and directions, IEEE Transactions on Industrial Informatics, 14 (2018), pp. 4724–4734.

[82] J. Sommers and P. Barford, Cell vs. wifi: on the performance of metro area mobile connections, in Proceedings of the 2012 internet measurement conference, 2012, pp. 301–314.

[83] K. Tan, J. Song, Q. Zhang, and M. Sridharan, A compound TCP approach for high-speed and long distance networks, in Proceedings-IEEE INFOCOM, 2006.

[84] H. Wu, Ö. Alay, A. Brunstrom, S. Ferlin, and G. Caso, Peekaboo: Learning-based multipath scheduling for dynamic heterogeneous environments, IEEE Journal on Selected Areas in Communications, 38 (2020), pp. 2295–2310.

[85] Y. Xing, J. Han, K. Xue, J. Liu, M. Pan, and P. Hong, MPTCP meets big data: Customizing transmission strategy for various data flows, IEEE Network, 34 (2020), pp. 35–41.

[86] C. Yu, W. Quan, N. Cheng, S. Chen, and H. Zhang, Coupled or uncoupled? multipath TCP congestion control for high-speed railway networks, in 2019 IEEE/CIC International Conference on Communications in China (ICCC), IEEE, 2019, pp. 612–617.

[87] H. Zhang, W. Li, S. Gao, X. Wang, and B. Ye, Reles: A neural adaptive multipath scheduler based on deep reinforcement learning, in IEEE INFOCOM 2019-IEEE Conference on Computer Communications, IEEE, 2019, pp. 1648–1656.

CONCLUSION AND FUTURE PERSPECTIVES

In conclusion, this thesis has made significant contributions to the field of MPTCP scheduling, aiming to enhance the reliability and throughput of the TCP Multi-Path protocol in multi-path networks. The research efforts have been devoted to addressing the challenges posed by path diversity and improving the performance of the packet scheduler, a crucial component in MPTCP.

Through comprehensive experimentation and evaluation of existing MPTCP schedulers tailored for asymmetric paths, we have gained valuable insights into their performance and efficiency in terms of bandwidth aggregation. Additionally, the issue of buffer blocking caused by unordered packets has been effectively tackled, as it has a substantial impact on the overall reliability and efficiency of MPTCP. This research has contributed significantly to the domain of MPTCP scheduling, addressing critical challenges and introducing innovative approaches. The results obtained demonstrate high scalability and compliance with bandwidth aggregation rules, laying a strong foundation for further exploration. By embracing the future perspectives of integrating machine learning for intelligent path selection and incorporating anomaly detection mechanisms, we can propel the field of MPTCP scheduling towards improved reliability and efficiency in multi-path network environments. These advancements are essential in meeting the ever-increasing demands of modern network communication and paving the way for next-generation network protocols.

Looking forward, there are several compelling perspectives for further enhancing this research. Firstly, the integration of machine learning techniques holds great promise in advancing MPTCP scheduling by incorporating intelligent path selection based on dynamic network conditions. By leveraging machine learning algorithms, we can develop sophisticated models that learn from historical data and adaptively choose the most opti-

mal paths in real-time, taking into account factors such as link quality, congestion levels, and network load. This approach would enable MPTCP to dynamically allocate network resources and optimize routing decisions, leading to improved performance and efficiency.

Furthermore, another crucial direction for future research lies in the realm of anomaly detection within the scheduling pillar of MPTCP. By integrating advanced anomaly detection algorithms, we can proactively identify and mitigate irregular network behavior, such as packet loss, link failures, or malicious attacks. Early detection and appropriate reaction to anomalies would enhance the reliability and robustness of MPTCP, enabling it to effectively handle unforeseen network conditions and maintain optimal performance.

It is important to emphasize that these future perspectives carry immense potential in advancing the field of MPTCP scheduling. By integrating machine learning for intelligent path selection and incorporating anomaly detection mechanisms, we can significantly enhance the reliability, efficiency, and adaptability of MPTCP in dynamic multi-path networks. This would open up new opportunities for more efficient utilization of network resources, improved Quality of Service, and seamless support for emerging applications and services.

Appendix A

Contribution

M. Bencheikh, C. Zouaoui, N. Taleb, A. Bouklif, A. Benaouda Chaht, M. Naimi, " A reliable alternating multipath transmission control protocol scheduling for bandwidth aggregation performance", Concurrency and Computation: Practice and Experience 1532-0626 / 1532-0634, 2023 https://onlinelibrary.wiley.com/doi/10.1002/cpe.7764