Master Thesis
Electrical Engineering
September 2015

Design and Implementation of CMT in Real-time

Evaluation based on scheduling mechanisms

Hadassah Pearlyn Nagathota

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Masters in Electrical Engineering with emphasis on Telecommunication Systems. The thesis is equivalent to 20 weeks of full-time studies.

Contact Information:
Author:
Hadassah Pearlyn Nagathota
E-mail: hana14@student.bth.se

University advisor:
Prof. Dr. Kurt Tutschku
Faculty of Computing
Blekinge Institute of Technology, Sweden

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Internet	:	www.bth.se
Phone	:	+46 455 38 50 00
Fax	:	+46 455 38 50 57

Abstract

Context: Standard transport layer protocols like UDP, TCP, and SCTP use only one access technology at a time. Concurrent Multipath Transmission (CMT), has been developed for parallel use of the access technologies. The main theme of this thesis work is to implement CMT in real-time and evaluate the impact of various scheduling algorithms on its performance.

Objectives: The main objectives of this thesis are to implement a de-multiplexer at the source, re-sequencer at the receiver and to investigate some of the heuristics and analyzing their impact based on some performance metrics.

Methods: Thorough understanding on this topic is attained by literature review of related works. To implement and evaluate the different scheduling patterns an experimental test bed is set up. For the transmission of data, socket programming in Python is used. Varying various parameters that are involved in the experiment, performance metrics were measured and based on them statistical analysis is carried out for proper evaluation.

Results: CMT is implemented in real-time test bed and concurrency is validated. Weighted Round-Robin has better performance compared to that of Round-Robin when the size of the packet is large whereas both exhibit nearly same behavior for smaller packet sizes.

Conclusions: It can be concluded that Weighted Round-Robin attains higher throughput. It can be possibly due to more load of fragmentation when large packets are transmitted on the high reliable path and hence better performance than Round-Robin. There is need for further evaluation of other metrics like delay, jitter and using other scheduling mechanisms and in other environments as well.

Keywords: Concurrency, Multi-homing, Performance, Scheduling

Acknowledgements

I would like to thank Almighty God who is the pillar that held my life throughout. I would not have garnered strength in many difficult situations without His wisdom and guidance.

I am indebted to my supervisor Prof. Dr. Kurt Tutschku for his academic guidance and time. His ideas and suggestions were very much thought-provoking and showed me the right direction whenever in doubt.

I would also like to thank my thesis partner Tedla Sukesh Kumar for his assistance during the course of the project.

I would like to express my deepest gratitude and love for my parents and my grandmother who drove and stood by me at every step ever since my childhood. I have been fortunate enough to be a part of great family. I also thank my friends and people at Karlskrona for their continuous support.

I will go before thee, and make the crooked places straight.

Isaiah 45:2

Contents

List of Figures

Abbreviations

BAS	Bandwidth Aware Scheduler
CMT	Concurrent Multipath Transmission
ICMP	Internet Control Message Protocol
IGMP	Internet Group Message Protocol
IP	Internet Protocol
LAN	Local Area Network
MPTCP	Multi-path Transmission Control Protocol
NTP	Network Time Protocol
NV	Network Virtualization
ODS	On-Demand Scheduler
OS	Operating System
OSI	Open System Interconnection
OSR	One-hop Source Routing
P2P	Peer-to-Peer
QoS	Quality of Service
RIP	Routing Information Protocol
RPDB	Routing Policy Database

RRB	Round-Robin
SCTP	Stream Control Transmission Protocol
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
TV	Transport Virtualization
UDP	User Datagram Protocol
USB	Universal Serial Bus
WLAN	Wireless Local Area Network
WRRB	Weighted Round-Robin

Chapter 1

Introduction

When a device has more than one network interface it is called multi-homing. At present all the devices are multihomed. This feature is very advantageous for the reason that when one of the networks fail the other network can still provide the desired service. Computing devices like desktop computers and laptops connect to the Internet through an Ethernet Local Area Network, Wireless Local Area Network (WLAN) or through any of the USB modems that are largely available today. In the miniaturized world, smartphones are playing a very important role in satisfying the needs of an individual round the clock and from anywhere with its large extent of services. Smartphones connect to the Internet through WLAN or Cellular 3G or 4G network. The underlying technology used for each of these for communications is distinct and hence separate network interfaces are required for each of them.

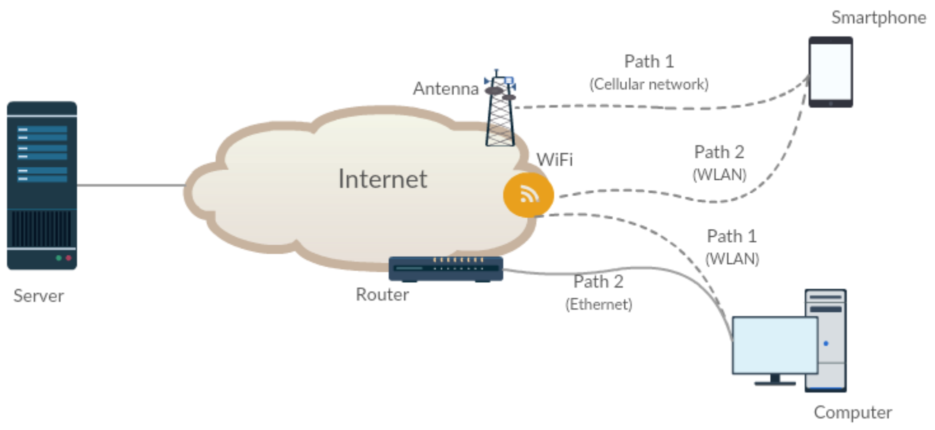


Figure 1.1: Multiple access technologies available for multihomed devices

The main purpose for the integration of these different types of access technologies is to merge all means of communications to access the Internet [1]. Number of technologies are combined to increase the bandwidth availability and the user has options to choose from any of them. There are many variations in the parameters for each network interface such as performance, cost and reliability. There are standard protocols for transport like UDP, TCP and SCTP. One of the approaches to obtain a resource in today's network is to open a single TCP connection with a single source where the resource is available and receive all the required data through this single connection. This results in slow downloads [2]. Multi-homing feature is supported by SCTP protocol. But the main drawback lies in the utilization of this feature. In SCTP, when a connection is lost, then in order to revive from that failure the secondary IP addresses are used. Due to the architectural constraints only one access technology can be used at a time.

Concurrent Multipath Transfer (CMT) which is an extension of the existing standard protocol SCTP, has been developed to exploit and enhance data communications. CMT provides a framework such that transport layer resources are used efficiently when transmitting to same destination that is multi-homed [3]. Network traffic load is shared among multiple paths which is known as Load Sharing. In this protocol, the data is split into number of chunks and is transmitted through several paths concurrently. Bandwidth may be limited depending on the access technology that is being used but when multiple interfaces are connected to different links the aggregate bandwidth is increased.

With the advantages that CMT possess, there are many more challenges that arise. Over the last decade, many researches have been conducted and there are ongoing investigations with regard to CMT. One of the main challenge that arise is the re-sequencing buffer occupancy. It is evident that when packets are sent through multiple paths they arrive out-of-order which in turn degrades the performance of the application. Hence, the packets are to be re-sequenced before delivering them to the application. A finite re-sequencing buffer is used for this purpose which gives rise to many other complications as well.

The multipath approaches share the fact that different path capacities may lead to performance degradations due to buffer blocking or additional delays due to re-sequencing mechanisms. This might be reduced by appropriate selection strategies in case of different path characteristics or appropriate scheduling mechanisms on different paths [4]. Various scheduling mechanisms have been developed by researchers to minimize re-ordering. The scheduling mechanisms could have huge impact on the performance.

This thesis mainly focuses to implement and evaluate the impact of different scheduling patterns in CMT on a real-time test bed depending on the capacity of links to find out which scheduling pattern serves better for CMT and also to investigate the feasibility of implementing CMT such that when exact copies of a packet are sent on multiple paths only the packet which arrives first is used and the others are discarded. It also includes measuring of performance metric throughput by varying different parameters.

1.1 Problem Statement

The main challenges of CMT are minimizing the re-sequencing buffer occupancy and path selection. Furthermore, different scheduling mechanisms on utilized paths may also influence the performance of the system. Many analytical and simulative models for the addressed problems in multipath transmissions exist in literature.

Various multipath approaches converge at the same point stating that different path capacities or latencies shows impact on the performance of application because of buffer blocking and delays due to re-sequencing mechanisms. They may be reduced by adopting appropriate scheduling mechanisms on various paths. Hence, implementing CMT in real-time and investigating the performance of different scheduling mechanisms is necessary.

1.2 Research Questions

1. How can scheduling mechanisms for CMT be implemented on real time test bed?
2. Is it possible to implement concurrent multipath transmission such that when same packet is sent on multiple paths only the one which arrives first is used?
3. What is the effect of different scheduling mechanisms on throughput when size of the packet is varied?

1.3 Contribution

There are many standard scheduling mechanisms which are developed to overcome the challenges and to enhance the performance of Concurrent Multipath

Transfer. This thesis mainly focuses on designing and implementing CMT for a multihomed source (with three interfaces) and a single-homed destination on a real-time test bed setup. This work is implemented in collaboration with another thesis “Performance Evaluation of Concurrent Multipath Transmission – Measurement and Analysis” [5] since both of them have similar objectives of achieving throughput in different scenarios of CMT. Hence, implementation of testbed is quite similar. However, this work is more focused on design and implementation of CMT whereas Software programming was of more priority in the other thesis.

In this thesis, implementation and concurrent transfer of data across all the paths is validated. Certain experiments were conducted in order to evaluate the main advantage for aggregating multiple paths and transmitting the data. Throughput is evaluated when different scheduling mechanisms (Round-Robin, Weighted Round-Robin) were used for scheduling packets at the source by varying the size of each chunk that data is fragmented.

1.4 Outline

This report consists of 6 chapters. Chapter 1 deals with the introduction to the thesis, problem statement, research questions and the contribution in the thesis. Chapter 2 deals with the fundamentals and related work. The fundamentals required to understand the main idea of the thesis such as the transport layer protocols, concepts of multi-homing, virtualization, scheduling patterns and also the previous works in this regard. Chapter 3 deals with the methodologies used during the course of completion of this thesis. Chapter 4 describes the implementation and technical details of the experimental setup. Chapter 5 is an analysis of the parameters observed during implementation and the results obtained through the experiments conducted. Chapter 6 contains the conclusions that are derived after analysis and the future work possible in this area.

Chapter 2

Fundamentals and Related Work

Data communication is known as exchange of data between two devices in any form of transmission medium [6]. The OSI model is a framework in network systems that allows communication between various types of computer systems. OSI model has seven layers namely physical layer, data link layer, network layer, transport layer, sessions layer, presentation layer and application layer as shown in Figure 2.1. All the layers are related to each other. Each layer has a specific functionality for transmitting information across a network.

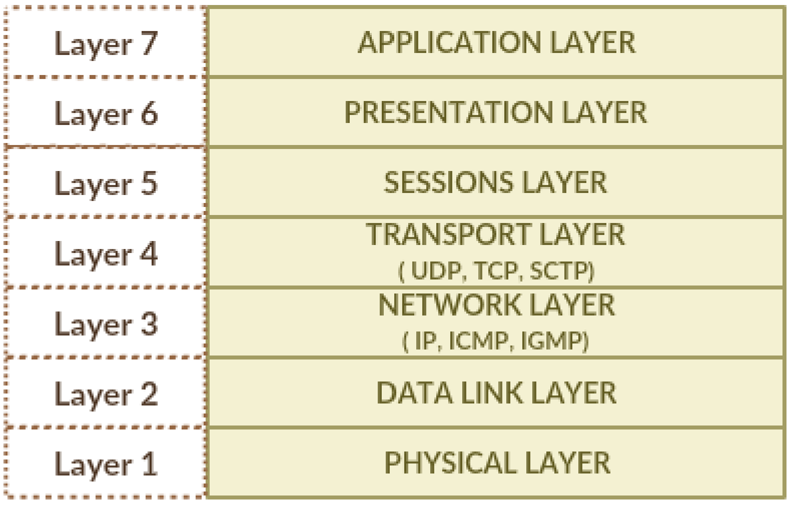


Figure 2.1: OSI Model - A framework in Network Systems

Protocol is a set of rules that are essential for communication in a network. There are number of fundamental protocols in each layer like Internet Protocol (IP), Internet Control Message Protocol (ICMP), Internet Group Message

Protocol (IGMP) are present in Network Layer whereas, User Datagram Protocol (UDP), Transmission Control Protocol (TCP), Stream Control Transmission Protocol (SCTP) are present in Transport Layer. The main emphasis of this work is on Transport Layer protocols. Hence, the features of the transport layer protocols are briefly described in the following section. Multi-homing, Virtualization concepts, Concurrent Multipath Transfer, Scheduling algorithms and related work are described in brief in this chapter.

2.1 Principal Protocols of Transport Layer

Number of processes belonging to different applications will be running at the sender and the receiver hosts simultaneously. There is necessity for a mechanism that can deliver data from any of these processes at the sender host to the corresponding process running at the receiver [6]. The main functionality of the transport layer is process-to-process delivery of the entire message. As mentioned, there are some standard protocols and their extensions that function in the Transport Layer such as UDP, TCP, SCTP, MPTCP, CMT etc. The features of these protocols are described in brief for better understanding in this section.

Transport layer protocols can be classified based on various criteria. Firstly, data can be transmitted from a source to a destination through a single path or through multiple paths. Therefore, protocols can be classified based on the ability to establish the number of paths as Single path or Multi path. Secondly, when using multi-path transport data can be transmitted one after other or concurrently. Therefore, they can be classified based on the ability of utilizing the multiple paths [7]. This classification is shown in Figure 2.2. The features, problems and challenges of the existing major protocols is reviewed below.

2.1.1 UDP and TCP

User Datagram Protocol (UDP):

User Datagram Protocol known as UDP is a connectionless protocol i.e, packets are transmitted from sender to destination with no need of connection establishment or release. Data is sent through a single path in UDP. It is also an unreliable protocol since the packets are unnumbered and can be delayed or they may be lost or even can arrive out of order. The delivered packet doesn't receive an acknowledgement either. UDP is almost similar to that of the Internet Protocol (IP) of the network layer except that it provides process-to-process

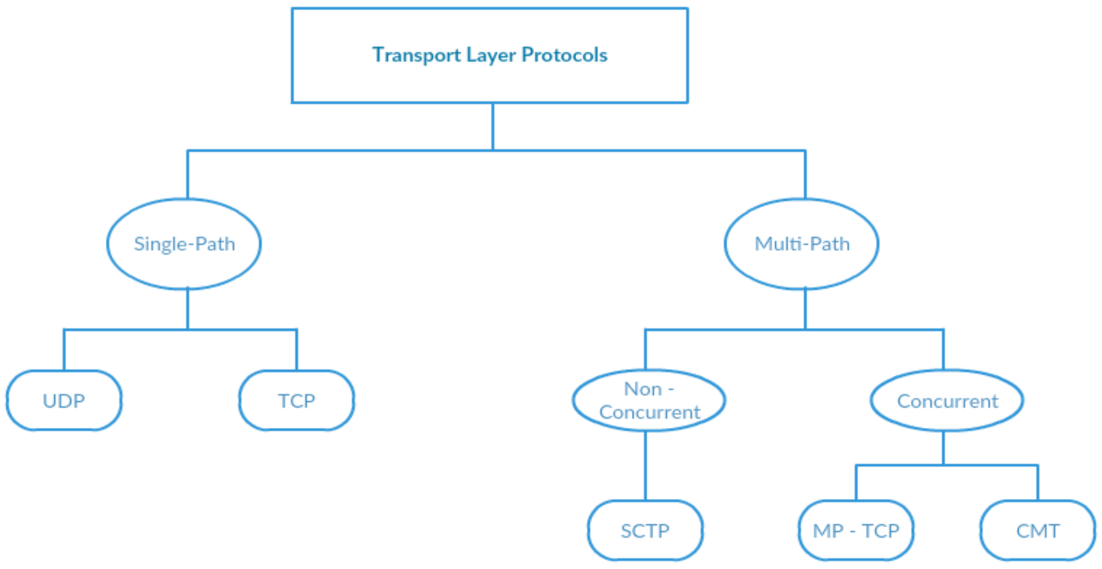


Figure 2.2: Classification of Transport Layer Protocols

communication instead of host-to-host communication [6]. It is a congestion less protocol and there is limited error checking compared to other protocols.

It is a simple protocol with minimum overhead. If an application doesn't care about reliability and wants to send a small message to a destination then UDP can be used. It is very much suitable for processes which have flow and error control mechanisms internally. It is used for broadcasting, multicasting, for management processes such as Simple Network Management Protocol (SNMP) and also for protocols like Routing Information Protocol (RIP) which updates the routes.

The main limitations of UDP are:

1. No guarantee for delivery of message
2. Data must be manually broken into packets before sending. It is not done internally.

Transmission Control Protocol(TCP):

Transmission Control Protocol known as TCP is a connection-oriented protocol. It has single source and single destination IP address. A virtual connection is created to send data. It is also a stream oriented protocol which means it enables delivery and reception of data in sending and receiving processes respectively as stream of bytes. The speed of the data transmission and reception at the sending and receiving side may not be the same always. For this purpose, buffers are required for storage of data [6]. Since, IP sends data in the form of packets but not stream of bytes TCP divides and clubs number of bytes into a segment. It is a reliable protocol. It makes use of acknowledgment mechanism to keep a check on the arrival of data packets correctly. Retransmission of packets is the primary mechanism used for flow and error control in TCP. It keeps retransmitting until the lost or corrupted packet is received at the other end.

The main limitations of TCP are:

1. TCP connections timeout leading to application recovery. This leads to undesirable delay and recovery overhead.
2. TCP is a single path protocol. Single connection is established between a TCP client and TCP server. So if any problem arises stream is blocked and has to wait till it is revived. This kind of transmission is not desirable when transmitting real time audio or video.

2.1.2 Stream Control Transmission Protocol (SCTP)

Stream Control Transmission Protocol known as SCTP is a message-oriented, reliable protocol. SCTP has the best features of both UDP and TCP. It keeps message boundaries and also has a check on flow control, error control and congestion control mechanisms. SCTP supports multi-stream service which means multiple streams in one connection where each stream is known as an association [6]. Hence, it supports multipath transmission of data. If one of the streams is blocked, the other streams will still be available for transmission of data.

It supports multi-homing service. Therefore, hosts at the sending and receiving end can define multiple IP addresses and utilize them for different associations unlike TCP. This is a fault tolerant approach which implies when one of the path fails the other paths can be used to deliver data without any hindrance for transmission. This kind of an approach is very much useful while transmitting and receiving real-time payload data such as Internet telephony.

The main limitations of SCTP are:

1. In existing SCTP, only a pair of IP addresses can be chosen for normal communication. Other addresses can be used only if the primary address fails. So, SCTP doesn't support load sharing between different paths.

2. SCTP doesn't support concurrent transfer of data.

2.1.3 Multi-path Transmission Control Protocol (MP-TCP)

Multi-path Transmission Control Protocol known as MP-TCP, is an effort for simultaneous use of multiple paths at the transport layer [8]. The main drawbacks of using either SCTP or TCP on multiple paths is that their specifications do not solve the problems which arise due to re-ordering. There are three main reasons for TCP to be chosen for implementing multipath capabilities.

1. TCP is widely deployed for transmission in today's Internet. It must be as usable as regular TCP for existing applications [9].
2. Sequence space plays a vital role and MP-TCP uses dual sequence number which helps in eliminating the problems of re-ordering.
3. Due to the way shared bottlenecks are dealt in TCP.

MP-TCP was designed to make it transparent to both application and network. It is a reliable protocol. When it is diagnosed that packets are lost over a high delay path then it is evident to retransmit the packets on path with less delay and it is done by MP-TCP. Using multiple paths for transmission has many advantages such as efficient resource utilization, obtaining better throughput etc. Coupled congestion control is an important innovation in the designing of MP-TCP. It includes packet scheduling, sub flow interface and congestion control.

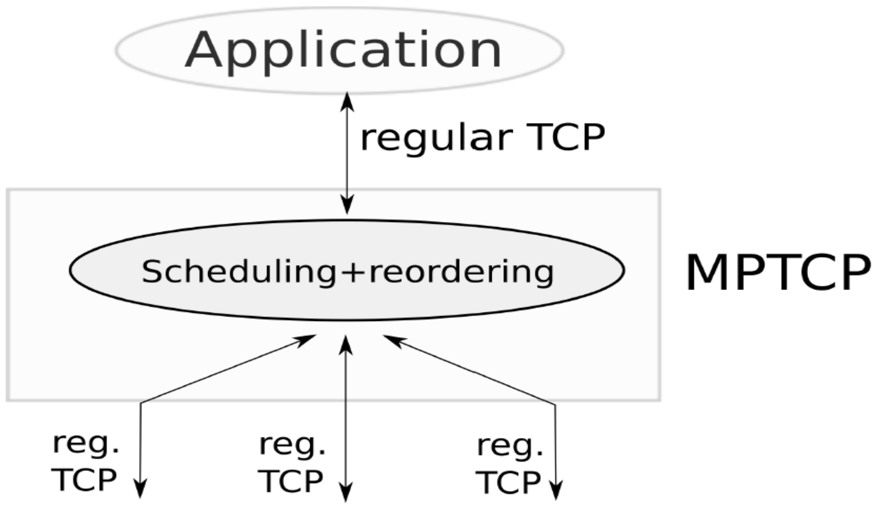


Figure 2.3: Operation of MP-TCP [8]

Each interface has a sub flow. The byte stream is split across these sub flows. There is a congestion control mechanism for MP-TCP. It can handle paths that have different bandwidths. The mechanism helps in moving the traffic to a less congested path. Hence, it can adapt load balancing depending on the load of traffic on various paths [10].

2.2 Drawbacks of Current Internet Model

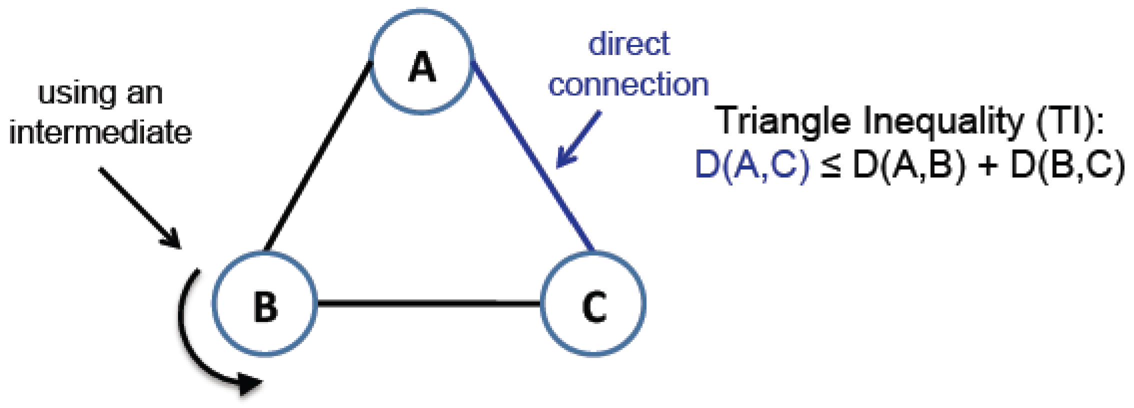


Figure 2.4: Triangle Inequality [11]

One of the major shortcomings of the present internet is the control of end-to-end Quality of Service [11]. The reliability of the current system is very poor. The present internet protocols discover the shortest route to a destination. As shown in Figure 2.4, data is to be transmitted from node A to node C which is the direct connection between them. But data is received faster when it is sent through an intermediate node B than the direct connection. This is violation of Triangle Inequality. In 25% of the cases the triangle inequality is violated. Present internet routing is far from being optimal and it is also evident that better routes exist [11].

2.3 Multi-Homing

One of the main challenges of the Future Internet is the support for multi-homing between various access networks which aims at providing services to users anywhere and anytime [12]. It is a functionality where a host system with multiple interfaces can be connected to multiple networks simultaneously through different Internet Service Providers (ISP's). This also indirectly helps in increasing resilience to path failures. A single network has two or more connections to the Internet and hence the hosts can be accessed even if any one of the IP addresses is unreachable at that moment. It enhances the fault tolerance of a host which means switching of data streams from one network technology to other without any interruption to the application. Currently, SCTP supports multi-homing as a backup service i.e, only when a primary connection fails SCTP makes use of the secondary IP address to recover.

2.3.1 Network Topology

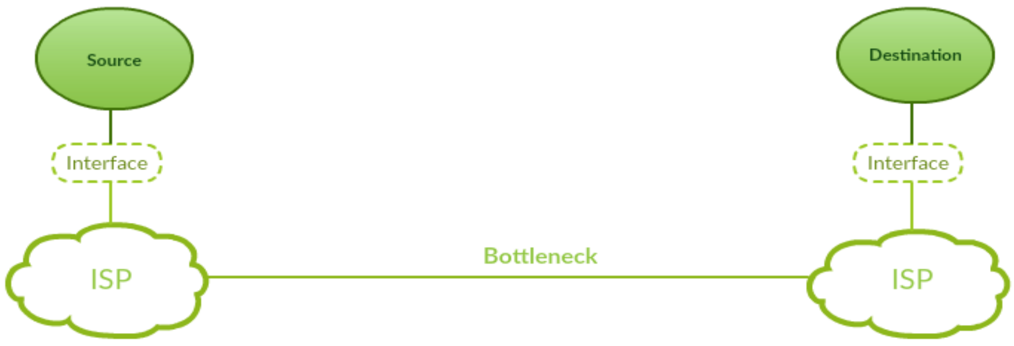


Figure 2.5: Single-homed Network Topology

In a computer network, network topology can be defined as an arrangement of network elements such as nodes, links etc. Basically, transport layer is modelled as dumbbell topology. One bottleneck link divides the sender and receiver. As depicted in Figure 2.5, there is only one sender/receiver pair. Only one network interface connected to Internet Service Provider (ISP) is present at the sender and the receiver and so it is known as Single-homed and can't support multi-homing. ISP may or may not be same at source and destination.

When an end-point has multiple network interfaces (multi-homed) it supports multi-homing as explained in previous chapter and it can be modelled as series of dumbbells. In such topology, more than one path exists between the source and destination. As shown in Figure 2.6, each interface can use a different access technology and are connected to ISP's which also may or may not be the same for all the interfaces.

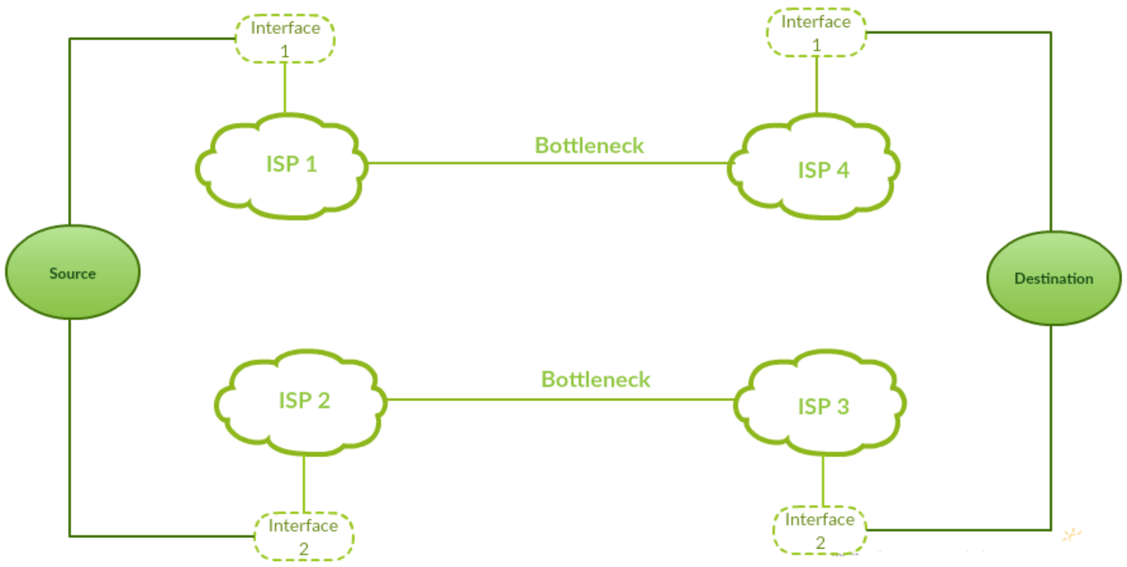


Figure 2.6: Multi-homed Network Topology

There are many similarities between multi-homed and single-homed topologies. Prior to placing data in the buffer it is copied from application layer and is transformed into packets [3]. The difference is in how the packets are transmitted among many destination addresses.

- A scheduling pattern need to be implemented because it plays a crucial role when performance characteristics vary in different paths.
- Each destination needs to have a send queue. It is helpful in determining if a packet is scheduled to that particular destination and is transmitted or not.

2.4 Virtualization

Virtualization can be described in this context as abstraction of compute resources. The virtualization technique makes the resource memory independent of its location. The actual location of the data doesn't matter and is hidden. There are many kinds of Virtualization techniques and enormous benefits in each of them. This section deals with some concepts beginning with one of the innovations in the current Internet model Peer-to-Peer Systems, Network Virtualization and Transport Virtualization which form the basis of Concurrent Multipath Transmission Protocol.

2.4.1 Peer-to-Peer Systems

Peer-to-Peer systems known as P2P systems are one of the distributed systems which work on the principle of multi-source download. Here, the abstraction of storage resources is introduced. It consists of equal entities known as peers, which share resources on application layer by direct end-to-end connections. The peers implement both client and server functionality and most of the system's state and tasks are dynamically allocated among the peers [13]. Here, one peer downloads multiple parts of the file from different peers in parallel. Hence, the downloading peer doesn't depend on one single peer which in turn increases the reliability.

The main advantage of this system is to optimize the throughput. So, the peers are selected in such a way. Some of the examples of P2P systems are eDonkey and BitTorrent. Earlier, P2P systems were mainly used only for file sharing applications but eventually P2P systems are being used and include the distribution of data, software, media content, Internet Telephony and scientific computing [13]. P2P systems form two different overlays. One overlay is dedicated to distribution of query information and the other for user data exchange [11].

2.4.2 Network Virtualization

The Future Internet consists of many interconnected subnets and nodes but the availability of these resources is constrained to a point. So, leasing of these resources is required. Networks that are based on the model of temporal leasing of resources are known as Federated Networks and the technologies that enable safe sharing of such resources are denoted as Network Virtualization (NV) mechanisms [14]. NV is a technology that allows simultaneous use of multiple logical networks on a single platform [11]. Network Virtualization can also be defined by decoupling the roles of the Internet Service Providers into two independent entities Infrastructure providers who manage the physical infrastructure and service providers who create virtual networks by aggregating resources from multiple infrastructure providers and offer end-to-end services [15].

Network Virtualization is built on the foundations of virtual network structures such as peer-to-peer overlays. Overlays do not require or cause any changes to the underlying network [15]. Overlays are capable to achieve higher performance and are very reliable because of their ability of forming network structures specific to the application. The availability of various network resources and OS Virtualization also form the basis of Network Virtualization. NV stands as a motivation for routing overlays. They are virtual network structures whose functionality is to forward random data. Various approaches have been proposed for routing overlays and a promising approach among them is the concept of one-hop source routing (OSR).

In OSR, when a given physical path suffers from the link failure or performance degradation data is forwarded through single intermediate node which relays traffic to the destination using ordinary IP routing [16]. IP system is segregated into two virtual overlays. One overlay is used for signaling and the other one for forwarding. The incoming traffic from tunnels is accepted by the software routers and the traffic is forwarded using IP routing protocols to the specified destination. The main advantages of OSR are performance control by selecting intermediate node and scalability. It is very effective in network failures. The drawback of OSR is that it is not capable to achieve high throughput data transmissions.

2.4.3 Transport Virtualization

P2P systems stands as a motivation for abstraction of data transport resources leading to Transport Virtualization (TV). Here, as long as the resource is accessible the physical location doesn't matter. TV enhances the capabilities of the Future Internet. It aggregates multiple physical or virtual networks and selects the best resources for unshared or concurrent use [14]. Abstract data transport resource can be a leased line, an IP forwarding capability to a certain destination or overlay link.

One of the challenges faced is the selection of appropriate paths among all the available set of paths. The characteristics of the paths has direct impact on reliability, security and QoS (Quality of Service) parameters between source and destination of the transmission. Upsurge in reliability and throughput are the assets of transmitting data concurrently over multiple paths. There should be a mechanism for distributing data among used paths and also for selection of appropriate set of paths. The set of used paths need not belong to one service provider. So, data transport is not bound to any one of the providers anymore [14]. Inexpensive and fast resources can be pooled together to form a high capacity pipe. Pooling is achieved in TV by a mechanism which is placed on end hosts that are multihomed.

2.5 Concurrent Multipath Transfer

Stream Control Transmission Protocol (SCTP) supports multi-homing. When the primary address fails the secondary address comes into play till the former is revived. There is no concurrent transfer of data over multiple paths. Concurrent Multipath Transmission known as CMT, is an extension of SCTP. The problem of achieving high throughput data transmissions in one hop source routing is resolved due to CMT. CMT mechanism combines multiple overlay paths into one virtual high capacity pipe [11]. The concept of CMT is to make use of multiple paths which are being ignored by the current routing system. The combined paths are used to send data packets from source to destination concurrently on different paths in parallel. This mechanism is known as Stripping [17]. The use of parallel transmissions is corresponding to the principle of P2P content distributed systems i.e multi-source download. In CMT, peer which requests the transmission service selects the best paths that gives highest combined throughput [11].

2.5.1 Stripping Mechanism

Two or more overlay paths are combined and used in parallel by sending data packets concurrently and this principle is known as stripping. The ingress router selects the paths which form the virtual pipe out of all the available paths. The data stream is then divided into segments which are split into 'm' parts. These segments are transmitted to the destination through 'm' parallel paths. The packets can arrive out-of-order at the receiver side due to the stochastically varying delay present in each of the paths. This degrades the performance of the application which is undesirable. In order to minimize this problem, the receiving router retains a finite re-sequencing buffer. But, when the re-sequencing buffer is full and there are packets yet to arrive then part loss may occur which also deteriorates the performance of the application. Handling of re-sequencing buffer increases the waiting time and hence the end-to-end delay.

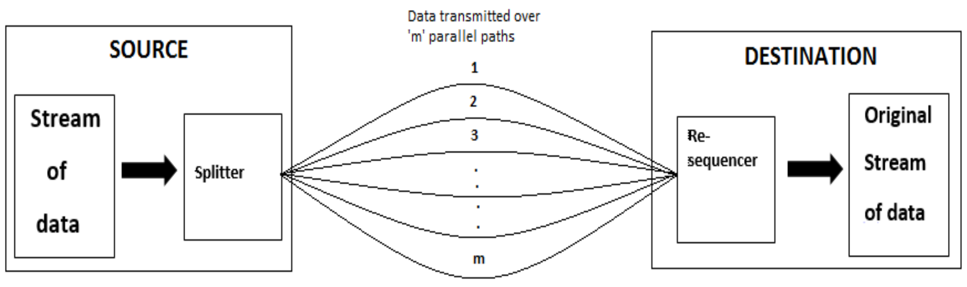


Figure 2.7: Concurrent Multipath Transmission

2.5.2 Advantages of Concurrent Multipath Transfer

Unlike SCTP, CMT features enables use of multiple paths for transmission of data. The main advantages of CMT are:

1. Heterogeneous access networks prompt the bandwidth aggregation for Concurrent Multipath Transfer to enhance transmission throughput and reliability [18].
2. More resilience for the user since the system doesn't rely on single path.
3. Higher network utilization for operators
4. Facilitates inter domain traffic management

2.5.3 Challenges of CMT

Though there are many advantages of Concurrent Multipath Transfer, there are number of challenges that are to be completely addressed. Some of the main challenges of CMT are:

1. To minimize the re-sequencing buffer occupancy: The packets are transmitted through parallel paths. Due to the stochastical delay present in these paths packets arrive out-of-order. Hence, a finite re-sequencing buffer is used at the destination.
2. Selection of paths: There are number of paths joining any two hosts. Hence, selection criteria for paths plays a vital role in the performance of CMT.

2.6 CMT Scheduling

Assuming that destination has an infinite receiving buffer is not always realistic, especially in smart phones with limited memory. When multiple paths have different path characteristics throughput can reduce to the capacity of the slowest path [3]. Therefore, for limited buffering data need to be transmitted reliably and in sequential order to a destination. Various multipath approaches converge at a same point stating that different path capacities or latencies lead to degradation of application performance because of buffer blocking and delays due to re-sequencing mechanisms [14]. The main responsibility of a scheduler is to pair packets in the send buffer with proper destination addresses so that re-ordering at the receiver is minimized.

2.6.1 Scheduling Algorithms

Round-Robin:

The round-robin scheduling is a simple mechanism. Each process is assigned a time slice which is equal and in circular order. Whenever there is space in the receiving buffer of a particular destination, packet from send buffer is assigned and is passed to the network layer for transmission to the destination. When multiple destination addresses have space in their respective receiving buffers, packets are assigned at the send queue in a round robin fashion each time to another destination address. P1, P2, P3 and P4 packets are assigned to the send queue in round – robin fashion as shown in Figure 2.8. P5 will be assigned to Queue 2 as per the order. Similarly, P6 to Queue 3, P7 to Queue 1 and so on.

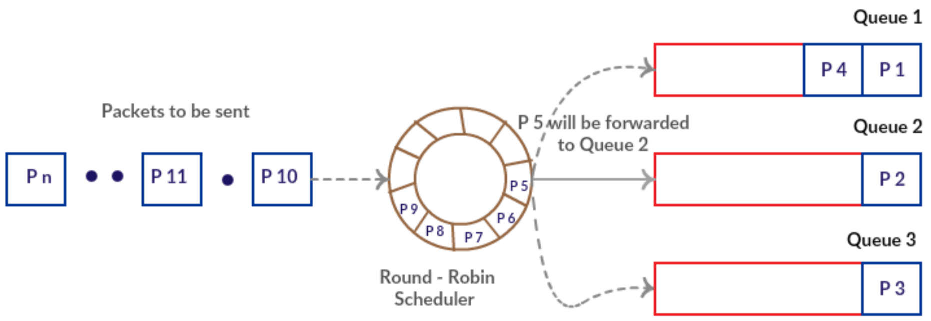


Figure 2.8: Round – Robin scheduling mechanism

This mechanism is not based on priority. Due to this, the aggregated performance decreases because of the disparity in the path characteristics which leads to situation where no destination can deliver packets faster than the slowest path irrespective of delay.

Weighted Round-Robin:

The weighted round-robin is a priority based scheduling mechanism. Each process is assigned a time slice similar to that of round-robin mechanism. But the difference lies with the priority. After the data is split into segments, the paths are given priority and the packets are sent accordingly in order of the weights.

More packets are sent on the high priority paths than the less priority paths. As depicted in Figure 2.9, Queue 2 has higher priority in this case. Hence, when one packet is forwarded to Queue 1 and Queue 3 two packets will be forwarded to Queue 2 which connects to the high reliable path.

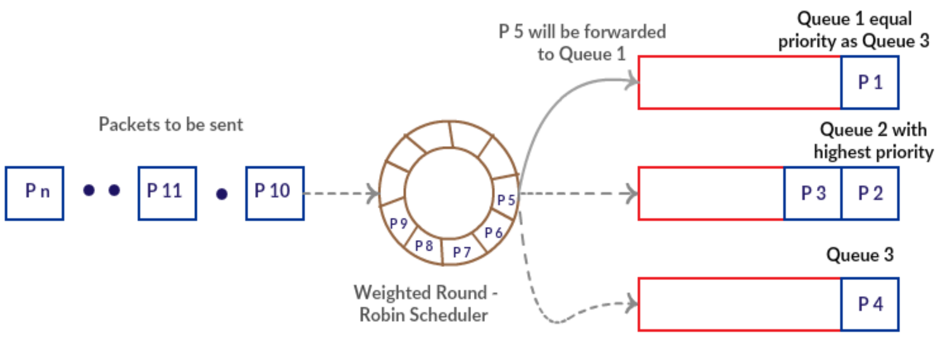

P 5 will be forwarded to Queue 1
Queue 1 equal priority as Queue 3
P 1
Packets to be sent
Queue 2 with highest priority
P n
P 11
P 10
P 5
P 3
P 2
P 9
P 6
P 8
P 7
Queue 3
P 4
Weighted Round - Robin Scheduler

Figure 2.9: Weighted Round – Robin scheduling mechanism

2.7 Related Work

Several research works have been conducted on Concurrent Multipath Transfer in previous years.

As described in earlier sections, re-sequencing the packets at the destination is one of the major challenge in CMT. To notice the main drawbacks of re-ordering introduced by CMT and to find out how it can be managed before parallel transmission itself, a research study was conducted by Janardhan R. Iyengar, Paul D. Amer, and Randall Stewart [19]. The three drawbacks that were reported are unnecessary fast retransmissions by sender, overly conservative congestion window and increased traffic caused due to acknowledgements which are delayed by a receiver. The authors propose some policies which augment and modify current SCTP to eliminate these drawbacks. They are evaluated varying the receive buffer sizes. They compared CMT against AppStripe which is a data stripping application. They conclude stating that retransmission policies which consider loss rate performs better.

One of such research works is on scheduling, modelling and congestion window management in concurrent multipath transfer by Thomas Daniel Wallace [3]. In this work the author focuses on many issues surrounding CMT and multi-homing. One of which is evaluating the performance capabilities of CMT when different scheduling algorithms are used under different network conditions. The author proposed a new scheduling algorithm known as On Demand Scheduler (ODS) to eliminate all the disparities caused due to Bandwidth Aware Scheduler (BAS) and Naïve Round Robin scheduler. It is stated that ODS offers improvement over the other scheduling approaches even when delay as its highest [3] under delay-based disparity. Whereas, when bandwidth-based disparity is considered naïve scheduler is equal to ODS when RBUF is small. But at higher RBUF sizes, ODS outperforms naïve scheduler. Under loss conditions, naïve scheduler performs better compared to ODS irrespective of the RBUF sizes. From this paper, the study of scheduling mechanisms and the basic implementation of them was understood and adopted in this thesis.

In [20], authors introduced two new protocols based on SCTP which add CMT support to the original SCTP. The performance of these protocols were evaluated by implementing them on the Linux Kernel. The authors considered the scenarios which eliminates packet re-ordering by estimating the bandwidth and Round Trip Time on each path and by using appropriate scheduling algorithm so that the packets reach the destination in order. Out of the two protocols that were introduced Westwood SCTP and Sender Based Packet Pair SCTP, it is concluded from the results that the former performs better when there is interfering traffic.

In [**?**], authors evaluated CMT on paths which vary in bandwidth, delay, error rate. Authors used receiver and sender buffer splitting which splits the buffer resources equally to eliminate the buffer blocking problem. The faster path sends more data when buffer is used dynamically.

Research study has been conducted to compare the two multipath approaches which are MultiPath TCP (MP-TCP) which is an extension of TCP and Concurrent Multipath Transfer (CMT) an extension of SCTP in [22]. The authors implemented the two protocols in lab and also on intercontinental testbed between Europe and China and compared the results. It is reported that there is significant impact on their performance with respect to the path management strategies. Measurements were taken in a scenario where four address-pairs are taken to initiate the connections. In this scenario it is concluded that throughput obtained in MPTCP connection exceeded the maximum throughput and is higher than all CMT-SCTP in all the cases. CMT-SCTP has influence depending on the initial address pair that is used for setting up the connection whereas MPTCP does not.

A new transmission control scheme Concurrent Multipath Transmission Scheme was proposed by Ming-Fong Tsai, Naveen K.Chilamkurti [23] to minimize the packet re-ordering at the destination during concurrent transfers. The authors implemented a feedback control mechanism to maximize the amount of data that is to be delivered without out-of-order packets. By sending a video, the proposed scheme was compared to the original scheme where odd packets are sent on one path and the even packets on the other path. It is observed that as the buffer size increases the frame loss is low giving better video quality and high Peak Signal to Noise Ratio.

In a research study done by Jingyu Wang, Jianxin Liao, Tonghong Li, the authors proposed a new traffic scheduling mechanism called Out-of-Order Scheduling for In-order Arriving (OSIA) [24] where the original packet sequence was disordered. Estimating the delays on each of the paths, the sending sequence is ordered such that packets at slower path are transmitted prior to that of the faster path. The proposed scheduling mechanism was compared with three different existing mechanisms that are Round Robin scheduling mechanism, CWND-aware scheduling mechanism and flow level scheduling mechanism. Varying the number of packets, the aggregate application level throughput obtained by Round Robin increases much slower. CWND-aware scheduling can increase the aggregate level throughput by increasing the CWND but when paths have different delays the end-to-end application level throughput is reduced.

One of the research study is conducted to evaluate the performance Delay Aware Packet Scheduling algorithm (DAPS) that aims to reduce the receiver's buffer blocking time to enhance the QoS in wireless environments [25]. The authors developed a model for maximum blocking time i.e, maximum time the packet stays in the receiver's buffer for in-order delivery. They evaluated the round robin mechanism which is generally used in CMT with DAPS in ns-2 simulation environment. It is reported that DAPS enables lower occupancy of the receiver's buffer which in turn enhances the utilization of aggregate link capacity and also reduces end-to-end transmission delay.

Paasch et al. , Ferlin, Ozgu Alay and Olivier Bonaventure evaluated and reported various schedulers for Multipath TCP [26]. The authors considered bulk data transfer and limited traffic as well to identify different metrics and evaluate the scheduler's performance. The four schedulers that were chosen and tested are Round-Robin, Lowest-RTT-First, Retransmission and Penalization and Bufferbloat Mitigation. Authors have used a Modular scheduler selection framework to experimentally evaluate the schedulers in number of environments for both emulation and real-world experimentation. They discovered that scheduling decision has two main effects which are head-of-line blocking and receive-window limitation. It is shown that when a simple strategy is used to schedule data on subflow with lowest RTT aids in reducing the application delay-jitter compared to that in Round-Robin scheduler. Similarly, Retransmission and Penalization extension mitigates receive-window limitation but when the delay-difference is high because of high bufferbloat the penalization cannot the congestion window properly.

Chapter 3

Methodology

This chapter presents the various research methods employed to achieve all the aims and objectives of this work. A literature review is conducted to gain more knowledge in this topic. As a part of experimental research, to implement CMT in real time scenario a testbed is designed and to transmit data from a source to destination software programming was required. Statistical analysis is done to observe the behavior of metrics and analyze them in different scenarios. These methods are described in detail in the sections below.

3.1 Literature Study

In order to extend knowledge and be familiarized about the background in the topic which is chosen for thesis, Concurrent Multipath Transfer literature study of the previous works which are related is very much required. To integrate and synthesize the key aspects its mechanisms, advantages and the main challenges in this research area are essential. It aids in the identification of the problems, implementation and configurations of the experimental setup and analyzing the results thoroughly. It offers a benchmark in the assessment of our own results.

Literature Study is a synthesis of the previous work which helps in understanding and framing the steps how to implement and approach further to achieve the expected outcomes of this work. It is useful for elaborating the basis of research questions and to figure out what research has already been done on this topic. Research papers, journals previous thesis and dissertations were chosen from databases such as IEEE Explore, Google Scholar, Inspec, ACM digital library etc. There are many research works conducted in various aspects of this topic. Conceptual sources which provide a framework for approaching the identified problem, which are relevant and those that help in the implementation and analysis of the results are considered. Detailed review of some of these papers was described in Section 2.7.

3.2 Experimental Research

For evaluating the available scheduling patterns and to find out which one can be used for concurrent multipath transmission, an experimental setup has to be designed and implemented. Experiment is an investigation using tests controlled according to the parameters considered [27]. Implementation, observation, decision and modification are the steps in an experimental research. An experiment is to be conducted to find out the impact of various scheduling mechanisms on performance metrics like throughput and how strong its influence is on various links.

First step of an experimental research is to design and implement the experimental setup properly. Second step is to conduct the experiment by varying different parameters. Number of iterations of the experiment is conducted to acquire accurate and reliable results. The next step is to take a proper decision about the results obtained. The results obtained may not be the same as expected in the theoretical deductions always. Various factors in an experiment influence the parameters involved. Hence, after analyzing the results properly a decision has to be taken if they are accurate or not. The next step is modification. In case the results obtained are completely variant from the actual behavior there should be some modification in the way the experiment is being conducted. Accuracy and reliability may be directly proportional to the number of iterations. The more the number of iterations more the accuracy and reliability.

3.2.1 System Design and Architecture

To implement and conduct experiments on Concurrent Multipath Transmission in real time scenario a testbed is designed. It is designed based on the measurements that are chosen to analyze and also depending on the resources that were available. It is designed assuming that the source has three interfaces and the destination has only one active interface.

As depicted in the Figure 3.1, the total data that has to be transmitted to the destination is split into ‘n’ number of parts. These packets that are split are given to a scheduler where according to the scheduling mechanism packets are scheduled from which interface they have to be sent and are forwarded to that assigned interface. The packets are sent in three different paths from the interfaces. A re-sequencer is placed at the destination to assemble the packets which arrive out-of-order into the right sequence.

At the source, an Ethernet connection and two 4G USB modems were considered to use at three interfaces respectively whereas at destination only one Ethernet connection was considered.

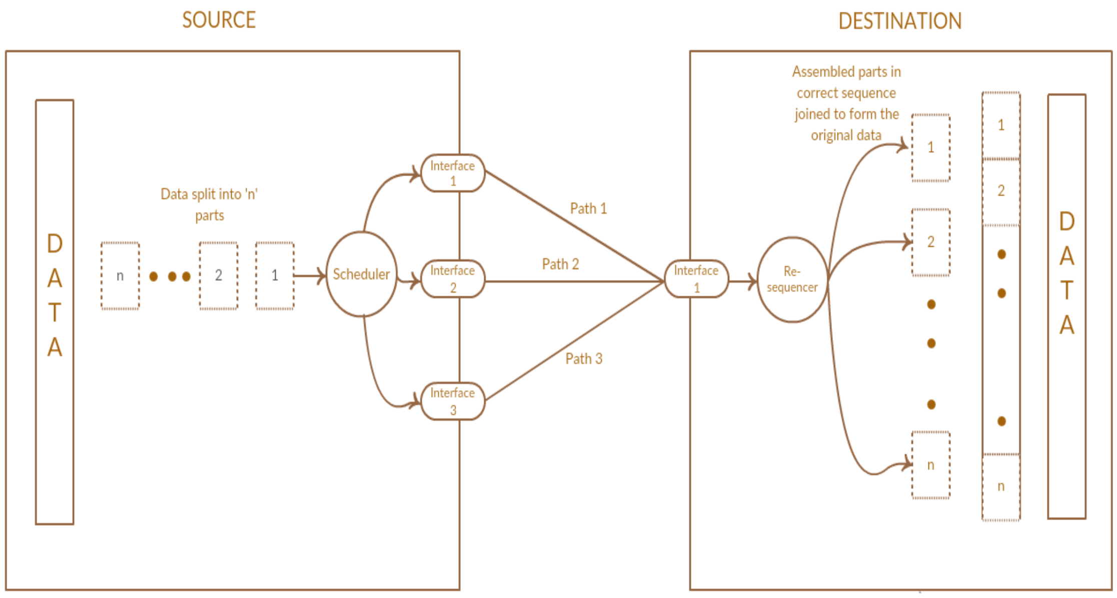


Figure 3.1: Architectural design for implementation of CMT

3.2.2 Devices and Tools used for Implementation

There are some devices and software tools that are used in the implementation of this work. They are described in brief.

1) **Mobile Broadband modem:**

Three interfaces are used at the source to transmit data through different paths. One interface is connected to Ethernet connection whereas the other two are connected to wireless 4G USB modems which are otherwise known as data cards or connect cards. Instead of any wired connection, they are used to receive Internet access through a mobile broadband connection. The IP addresses of these modems are usually private and depends on the Internet Service Provider.

2) **Eclipse IDE:**

Socket programming is executed in Python to fragment the data, transmit them concurrently on three different paths and also to re-arrange the packets at the destination after receiving. For that purpose, Python is installed in Eclipse IDE and is used to run the programs.

3) **Network Time Protocol(NTP)**

For calculating the metrics, start time and end time of transmission is to be measured at source and destination respectively. Hence, time synchronization is required for both the systems. For this purpose, NTP is used. It is a network protocol that is designed for clock synchronization of computers in variable latency and packet switched networks [28]. Time is fetched using NTP at source and destination and the metrics are calculated.

4) **Wireshark:**

Wireshark is a software that is used to analyze the packets. It is an open source packet analyzer. In this work, packets that are transmitted and received at the source and the destination are analyzed to identify and validate the concurrent transfer of data.

3.2.3 Software Programming

In this thesis, Concurrent Multipath transmission which is a transport layer protocol as described in the previous chapters is implemented. It is intercommunication of programs which are running on different computers in the network. In network-based systems, there exists a client and a server and a medium that connects both of them as shown in Figure 3.2. An interface for programming networks at the transport layer is known as a Socket. Communication between sockets is independent of the programming language used at the client and the server. The server program runs on a computer that has a socket which binds a specific port. The server listens to the socket for a client to make a connection request [29]. Python is a high-level programming language which is used on the Linux Kernel for the implementation of the thesis.

Programming involved in the thesis can be categorized into three main parts. Firstly, data is to be split into chunks before transmitting. After the data is fragmented, they must be scheduled properly for transmitting on multiple paths with different characteristics. The second step is transmission of these chunks on multiple paths concurrently using CMT. Parallel Processing modules such as "Asyncoro" in Python have been used for this purpose. The last step is to receive the chunks and re-order them at the receiver side. Because of the delay

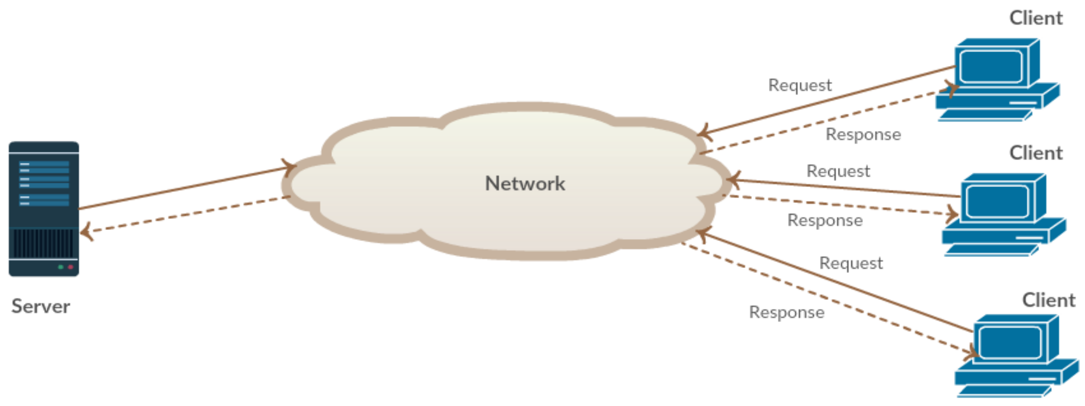


Figure 3.2: Client/Server Model for Software Programming

distributions on various paths they arrive out-of-order. So, before delivering them to the application layer they need to be properly re-ordered. For all the three parts programming is done and executed using Python scripting language modules on Linux Kernel. All these steps that are implemented are depicted in the flow diagram Figure 3.3. This shows the sequential order of steps involved in the program.

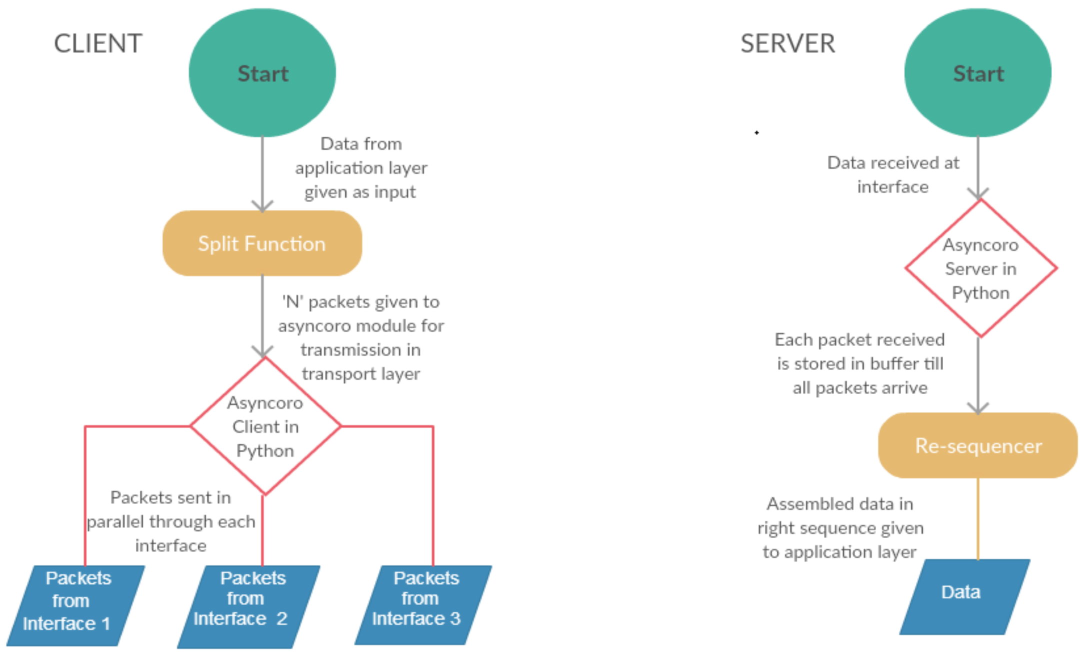

CLIENT
Start
Data from application layer given as input
Split Function
'N' packets given to asyncoro module for transmission in transport layer
Asyncoro Client in Python
Packets sent in parallel through each interface
Packets from Interface 1
Packets from Interface 2
Packets from Interface 3
SERVER
Start
Data received at interface
Asyncoro Server in Python
Each packet received is stored in buffer till all packets arrive
Re-sequencer
Assembled data in right sequence given to application layer
Data

Figure 3.3: Sequential flow diagram of program implemented

3.3 Statistical Analysis

A researcher can conclude on a theory through experiments. Statistical analysis is fundamental and plays a crucial role in most of the experiments. An experimental research includes two terms in common those are dependent and independent variables. Quantitative analysis is to record set of observations for inductive reasoning. One of the main objectives of this thesis is to analyze the performance metric throughput and to study its behavior when various scheduling mechanisms are used for scheduling packets at the source. The experiment is conducted varying different parameters in different environments. In order to achieve the desired outcomes number of experiments are conducted changing different parameters one at a time. To sustain the accuracy of the results observed, each experiment is performed number of times and the mean value of the results obtained is taken into consideration. Statistical analysis plays a key role in analyzing the outputs observed and validating them according to theoretical assumptions and results.

The parameters considered for analysis in this work are number of concurrent paths, size of each packet, number of connections between the source and destination and throughput. Experiments are conducted varying the size of packets, number of packets keeping other parameters constant. The complete description of the experiment setup and scenarios is given in the next chapter.

Chapter 4

Implementation and Experiment

The main focus of this work is to implement and evaluate different scheduling algorithms that are used for Concurrent Multipath Transmission mechanism. In order to investigate the actual performance and feasibility of different scheduling algorithms a prototype is implemented. This chapter presents an elaborate description on the design and implementation of the real time test bed setup, the changes that are made in the Linux kernel configurations and the experiment scenario that is conducted.

4.0.1 Testbed Setup

Prototype is deployed on an indoor testbed. In a real-world application for this scenario we made use of simple wireless links.

Testbed is a traditional client/server model as shown in Figure 4.1. It consists of two systems one acts as client and the other as server. Client is a multi-homed system with number of Internet connections (Ethernet and Wireless 4G modems). It is connected to one Ethernet connection and two wireless interfaces. The wireless interfaces that are used at the client side are ZTE8271 4G modems whose maximum download speed is 5.4MB/s. The two wireless interfaces at the client are connected to the same network with different IP addresses whereas the Ethernet is connected to a different network. Hence, the client has three paths.

Server is a single-homed system which has only one Ethernet connection. It is connected with an Ethernet connection which is in the same network as that of the Client Ethernet connected network.

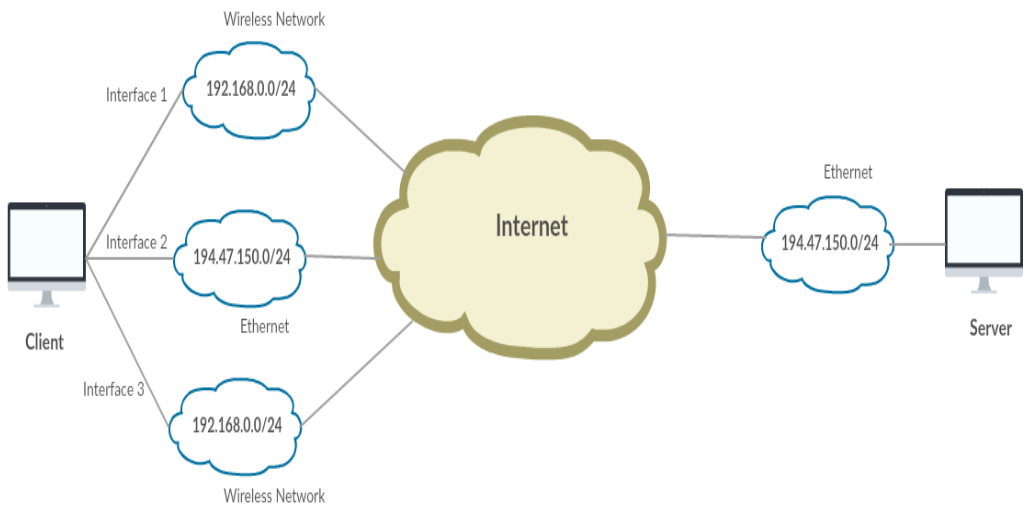


Figure 4.1: Testbed Setup

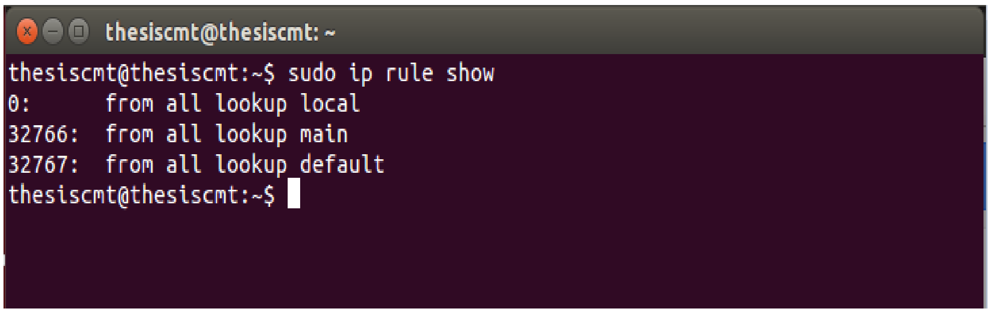
4.1 Linux Kernel Routing

Basing on the kernel's routing table, Linux Kernel supports simple IP forwarding functionality. Though it supports multi-homing, only one gateway is established for the system even if there are number of internet connections that are connected.

In order to activate and make use of multiple interfaces simultaneously some configurations need to be changed in the Linux Kernel. As shown in Figure 4.2, there are three default routes in the Routing Policy Database (RPDB). Any packet from anywhere will first be matched with routes in the local routing table and this is specified by the first rule. It is for broadcast addresses on link layers, network address translation and locally hosted IP addresses [30]. If the packet is not being sent to any of those destinations then the next entry in the RPDB is checked by the kernel.

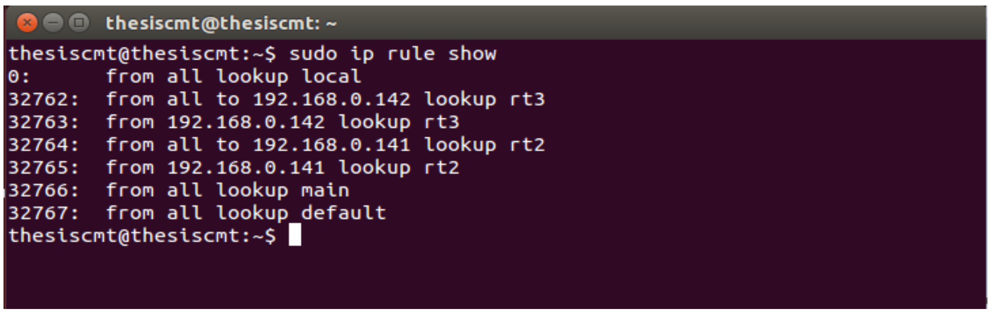
Routing tables are populated by using "ip route" command. To create and add the routing entries in tables rt2 and rt3, from terminal the following commands are used respectively.

```
ip route add 192.168.0.0/24 dev usb0 src 192.168.0.141 table rt2
ip route add default via 192.168.0.1 dev usb0 table rt2
ip route add 192.168.0.0/24 dev usb1 src 192.168.0.142 table rt3
ip route add default via 192.168.0.1 dev usb1 table rt3
```



```
thesiscmt@thesiscmt:~$ sudo ip rule show
0:      from all lookup local
32766:  from all lookup main
32767:  from all lookup default
thesiscmt@thesiscmt:~$
```

Figure 4.2: Default Routing Policy Database



```
thesiscmt@thesiscmt:~$ sudo ip rule show
0:      from all lookup local
32762:  from all to 192.168.0.142 lookup rt3
32763:  from 192.168.0.142 lookup rt3
32764:  from all to 192.168.0.141 lookup rt2
32765:  from 192.168.0.141 lookup rt2
32766:  from all lookup main
32767:  from all lookup default
thesiscmt@thesiscmt:~$
```

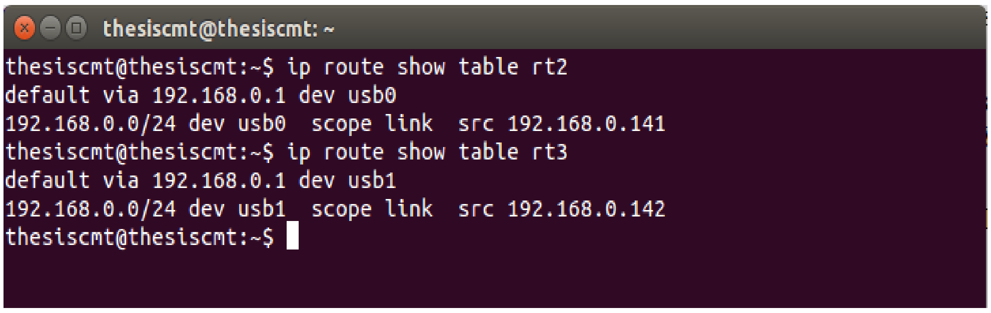
Figure 4.3: Routing Policy Database after adding tables "rt2" and "rt3"

After executing the commands, tables are created for "usb0" and "usb1" as "rt2" and "rt3" respectively as shown in Figure 4.3. The ip addresses and the gateways of these interfaces are also specified in the command. The next step is to add the rules for using these routing tables. To add rules following commands are to be executed from the terminal.

ip rule add from 192.168.0.141/32 table rt2
ip rule add to 192.168.0.141/32 table rt2
ip rule add from 192.168.0.142/32 table rt3
ip rule add to 192.168.0.142/32 table rt3

The first command adds a rule that all the incoming traffic to interface "usb0" IP makes use of "rt2" table instead of "main" table and the second command adds a rule that all the outgoing traffic from usb0 IP makes use of "rt2" table instead of "main" table. Similarly, third and fourth commands are with respect to "usb1" interface and "rt3" table.

As shown in Figure 4.4, rules are added to the tables "rt2" and "rt3" as given in the commands above.



thesiscmt@thesiscmt: ~

```
thesiscmt@thesiscmt:~$ ip route show table rt2
default via 192.168.0.1 dev usb0
192.168.0.0/24 dev usb0  scope link  src 192.168.0.141
thesiscmt@thesiscmt:~$ ip route show table rt3
default via 192.168.0.1 dev usb1
192.168.0.0/24 dev usb1  scope link  src 192.168.0.142
thesiscmt@thesiscmt:~$
```

Figure 4.4: Rules added to each interface

4.2 Implementation Details

Before sending and receiving packets to the wireless and Ethernet links packet flow must be accessed to route them to the destination through the more reliable Internet. Therefore the system consists of two core parts, the client and the server. In particular, client is multi-homed and its main job is to distribute packets efficiently through the wireless and Ethernet links to Internet. Then the packets are forwarded to the destination and from destination back to the client. The data communications on layer-3 via multiple networks is fragmented.

As described in previous chapters, Stripping mechanism is used in Concurrent Multipath Transfer. The data to be sent to destination is split into smaller packets and are sent using multiple paths that are set up through various interfaces. At the receiver, the data is arrived out-of-order due to stochastic delay present in each of the paths. Hence, the packets received are to be rearranged before delivering to the application layer. To achieve this, software programming is used in the implementation. Parallel processing module is used where the code is executed in parallel and doesn't wait for the first transmission to be completed to start the second transmission. This gives concurrency.

4.2.1 Working at Client

The outgoing packets are distributed by the client application via the Ethernet and wireless links. As described in before sections, at the client the data is split into number of smaller chunks and is transmitted in parallel via the three paths that are set up. It is observed in Wireshark Network Protocol Analyzer.

4.2.2 Working at Server

The packets that arrived at the server are observed using Wireshark. They are received out of order due to varying delays present in the paths. So, at the server the packets are re-arranged before delivering to the application layer. Code is written in Python to re-sequence the arrived packets and then deliver to the application layer. Programming is done in such a way that when multiple copies of same packet are being sent through different paths only the one that arrives first is considered and the rest are discarded.

4.3 Throughput depending on number of paths

An experiment is conducted to measure the throughput in concurrent multipath transmission varying the number of paths as an initial validation step. The setup is configured as described. After the configuration, experiment is run in three different ways which are listed below.

- A file is sent from source to destination making use of single path that is the path established through 1 USB modem.
- A file is sent from source to destination making use of two paths. These are paths established using 1 USB modem and 1 Ethernet.
- A file is sent from source to destination making use of three paths. These are paths established using 2 USB modems and 1 Ethernet.

Throughput is calculated considering the start time and end time of transmission.

4.4 Experiment Varying Chunk Size

An experiment is conducted to measure the throughput in concurrent multipath transmission varying the size of the chunk. The setup is configured as described in the previous sections. Routing tables and default gateway are configured for two USB modems and an Ethernet connection at the client/sender/-source and one Ethernet connection at the server/receiver/destination.

By varying the size of each chunk a file is transmitted from the source to the destination keeping the number of TCP connections in each path as constant. Source and destination are time synchronized using Network Time Protocol (NTP). For each chunk size throughput is calculated by measuring the start time of transmission at the source and end time at the destination. The experiment is repeated number of times for each transmission to ensure reliability. The mean values of the all the iterations are considered. Throughput is calculated as data transmitted divided by the total time taken for all the data to be transmitted from the source and received at the destination. Experiment is conducted for two scheduling mechanisms. The parameters considered during the experiment are given below.

1. **Scheduling mechanisms considered:**
    - Round-Robin
    - Weighted Round-Robin
2. **IP addresses used:**
    (a) At client
        - eth0: *194.47.150.241*
        - usb0: *192.168.0.141*
        - usb1: *192.168.0.142*

    (b) At server
        - eth0: *194.47.150.244*
3. **Number of paths:** *Three*
4. **File size used for transmission:** *104857600 Bytes (100MB)*
5. **Number of TCP connections:** *15*
6. **Size of chunks:** *Varied between 500 Bytes to 20000 Bytes*
7. **Time measured for calculating throughput:** *Used Network Time Protocol (NTP) for time synchronization of two systems*

In round-robin scheduling mechanism, the packets are split and are sent in equal number on each path. In weighted round-robin, as the name suggests one path is given more priority over the other. Hence, for usb0, usb1 and eth0 it is divided in the ratio 1:1:2 respectively. Priority is given to Ethernet in this scenario because it is the high reliable path. When 1 chunk is sent through the usb modems two chunks are sent through the Ethernet. This ratio is considered because Ethernet interface is more reliable and can transmit chunks with more speed.

Chapter 5

Results and Analysis

This chapter presents the results and the analysis of the results obtained during implementation and experimentation. For more accurate results, the experiment is repeated number of times and the mean values are considered.

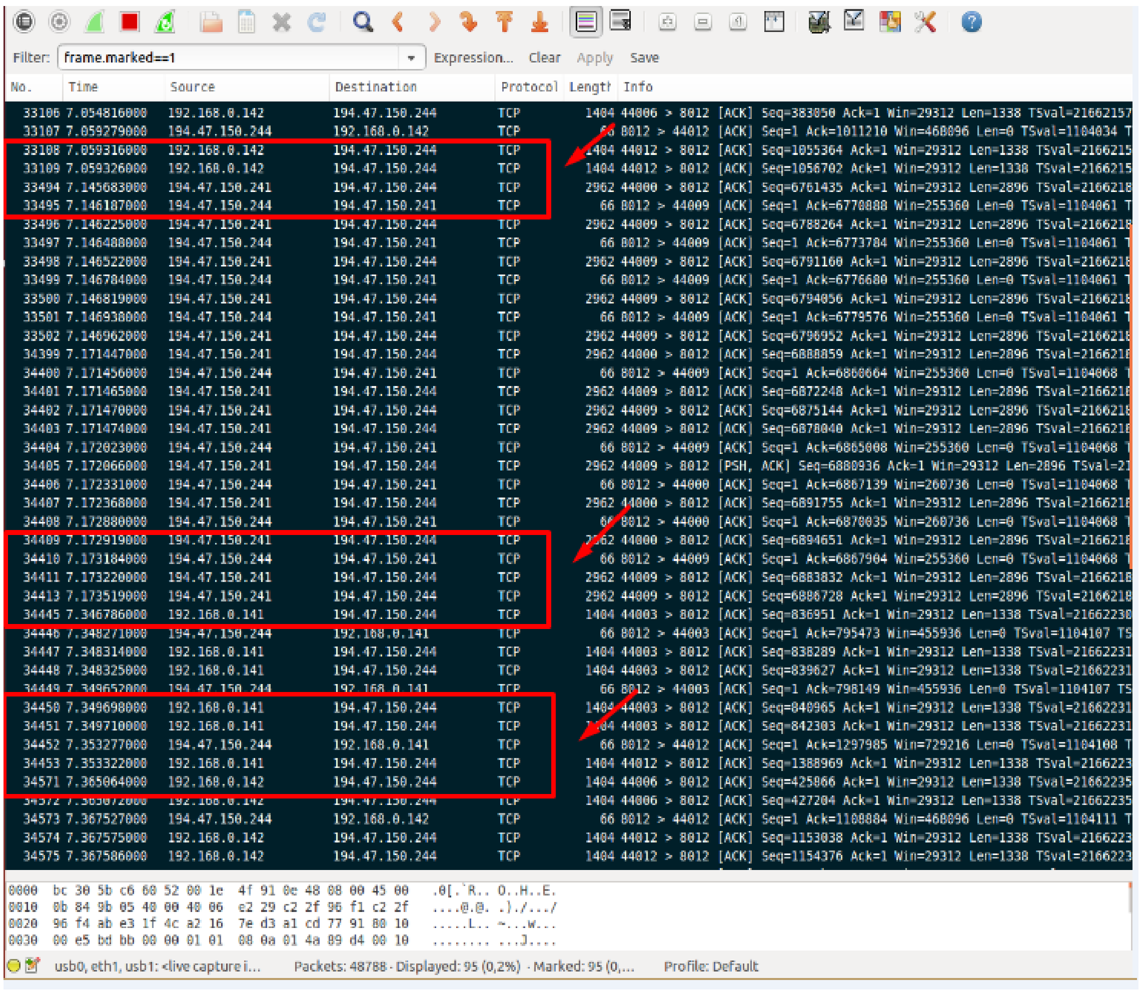
5.1 Section-1

Firstly, data is transmitted from source to the destination through the real-time test bed that is set up in order to observe that the packets are being sent concurrently across all the paths. It can be observed in two ways by analyzing the packets that are transmitted and received using Wireshark and also by measuring throughput when single and multiple paths are used.

5.1.1 Concurrent Transmission on Multiple Paths

When a file of size 100MB is sent from the source to the destination, packets were observed using Wireshark Protocol Analyzer. It helps in analyzing the packets source and destination IP addresses, time at which the packets are sent and received respectively. This is implemented on a real time testbed.

Figure 5.1 represents the packets sent from the source. As highlighted, the source IP addresses of the packets are 192.168.0.142, 194.47.150.241, 192.168.0.141 and the destination IP address is 194.47.150.244. This validates that the packets are not sent sequentially but are sent concurrently from the source.



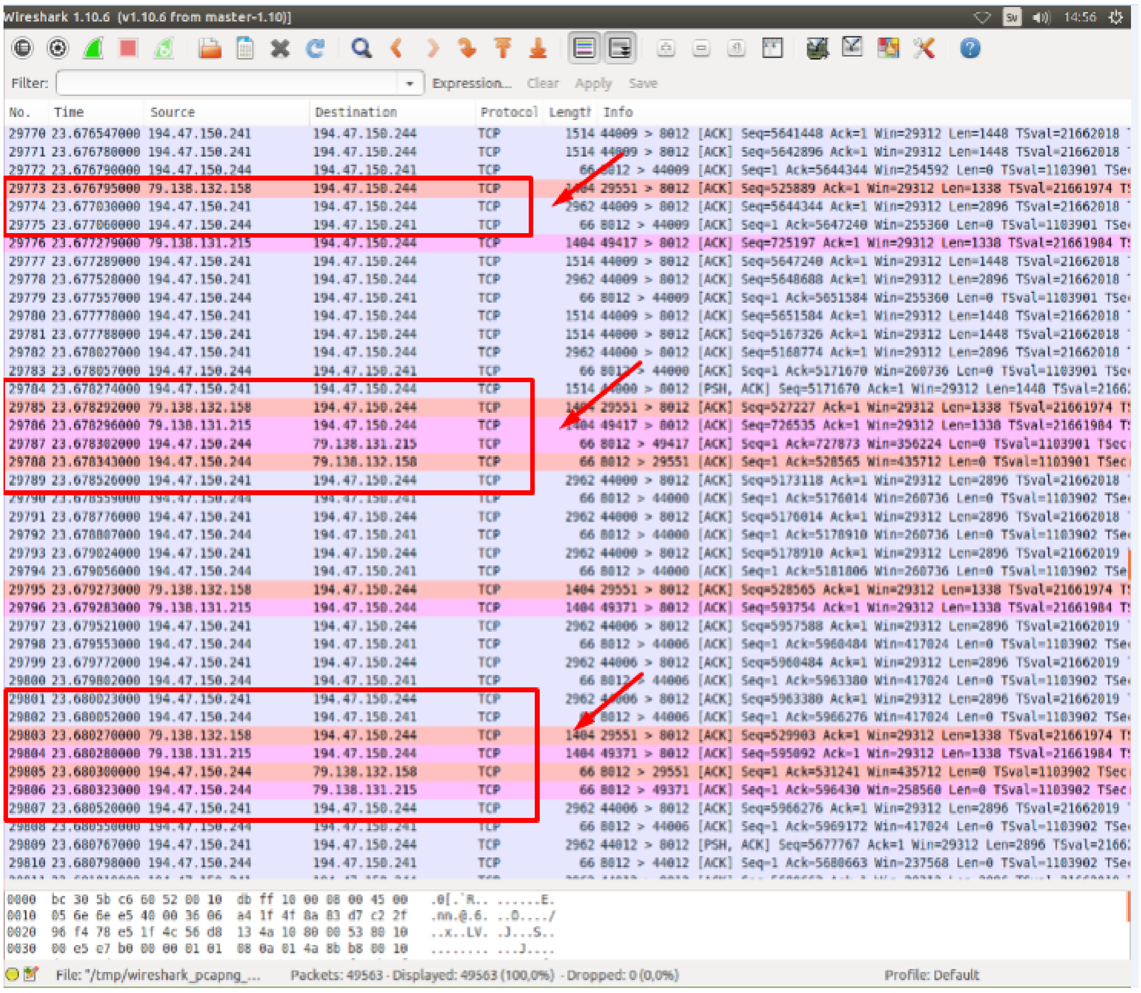
```
No.   Time         Source           Destination      Protocol Length Info
33106 7.054816000  192.168.0.142    194.47.150.244   TCP      1404 44006 > 8012 [ACK] Seq=383050 Ack=1 Win=29312 Len=1338 TSval=21662157
33107 7.059279000  194.47.150.244   192.168.0.142    TCP        66 8012 > 44012 [ACK] Seq=1 Ack=1011210 Win=468096 Len=0 TSval=1104034 T
33108 7.059316000  192.168.0.142    194.47.150.244   TCP      1404 44012 > 8012 [ACK] Seq=1055364 Ack=1 Win=29312 Len=1338 TSval=2166215
33109 7.059326000  192.168.0.142    194.47.150.244   TCP      1404 44012 > 8012 [ACK] Seq=1056702 Ack=1 Win=29312 Len=1338 TSval=2166215
33494 7.145683000  194.47.150.241   194.47.150.244   TCP      2962 44000 > 8012 [ACK] Seq=6761435 Ack=1 Win=29312 Len=2896 TSval=216621
33495 7.146187000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44009 [ACK] Seq=1 Ack=6770888 Win=255360 Len=0 TSval=1104061 T
33496 7.146225000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6788264 Ack=1 Win=29312 Len=2896 TSval=216621
33497 7.146488000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44009 [ACK] Seq=1 Ack=6773784 Win=255360 Len=0 TSval=1104061 T
33498 7.146522000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6791160 Ack=1 Win=29312 Len=2896 TSval=216621
33499 7.146784000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44009 [ACK] Seq=1 Ack=6776680 Win=255360 Len=0 TSval=1104061 T
33500 7.146819000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6794056 Ack=1 Win=29312 Len=2896 TSval=216621
33501 7.146938000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44009 [ACK] Seq=1 Ack=6779576 Win=255360 Len=0 TSval=1104061 T
33502 7.146962000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6796952 Ack=1 Win=29312 Len=2896 TSval=216621
34399 7.171447000  194.47.150.241   194.47.150.244   TCP      2962 44000 > 8012 [ACK] Seq=6888859 Ack=1 Win=29312 Len=2896 TSval=216621
34400 7.171456000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44009 [ACK] Seq=1 Ack=6860664 Win=255360 Len=0 TSval=1104068 T
34401 7.171465000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6872248 Ack=1 Win=29312 Len=2896 TSval=216621
34402 7.171470000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6875144 Ack=1 Win=29312 Len=2896 TSval=216621
34403 7.171474000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6878040 Ack=1 Win=29312 Len=2896 TSval=216621
34404 7.172023000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44009 [ACK] Seq=1 Ack=6865008 Win=255360 Len=0 TSval=1104068 T
34405 7.172066000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [PSH, ACK] Seq=6880936 Ack=1 Win=29312 Len=2896 TSval=21
34406 7.172331000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44000 [ACK] Seq=1 Ack=6867139 Win=260736 Len=0 TSval=1104068 T
34407 7.172368000  194.47.150.241   194.47.150.244   TCP      2962 44000 > 8012 [ACK] Seq=6891755 Ack=1 Win=29312 Len=2896 TSval=216621
34408 7.172880000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44000 [ACK] Seq=1 Ack=6870035 Win=260736 Len=0 TSval=1104068 T
34409 7.172919000  194.47.150.241   194.47.150.244   TCP      2962 44000 > 8012 [ACK] Seq=6894651 Ack=1 Win=29312 Len=2896 TSval=216621
34410 7.173184000  194.47.150.244   194.47.150.241   TCP        66 8012 > 44009 [ACK] Seq=1 Ack=6867904 Win=255360 Len=0 TSval=1104068 T
34411 7.173220000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6883832 Ack=1 Win=29312 Len=2896 TSval=216621
34413 7.173519000  194.47.150.241   194.47.150.244   TCP      2962 44009 > 8012 [ACK] Seq=6886728 Ack=1 Win=29312 Len=2896 TSval=216621
34445 7.346786000  192.168.0.141    194.47.150.244   TCP      1404 44003 > 8012 [ACK] Seq=836951 Ack=1 Win=29312 Len=1338 TSval=21662230
34446 7.348271000  194.47.150.244   192.168.0.141    TCP        66 8012 > 44003 [ACK] Seq=1 Ack=795473 Win=455936 Len=0 TSval=1104107 TS
34447 7.348314000  192.168.0.141    194.47.150.244   TCP      1404 44003 > 8012 [ACK] Seq=838289 Ack=1 Win=29312 Len=1338 TSval=21662231
34448 7.348325000  192.168.0.141    194.47.150.244   TCP      1404 44003 > 8012 [ACK] Seq=839627 Ack=1 Win=29312 Len=1338 TSval=21662231
34449 7.349652000  194.47.150.244   192.168.0.141    TCP        66 8012 > 44003 [ACK] Seq=1 Ack=798149 Win=455936 Len=0 TSval=1104107 TS
34450 7.349698000  192.168.0.141    194.47.150.244   TCP      1404 44003 > 8012 [ACK] Seq=840965 Ack=1 Win=29312 Len=1338 TSval=21662231
34451 7.349710000  192.168.0.141    194.47.150.244   TCP      1404 44003 > 8012 [ACK] Seq=842303 Ack=1 Win=29312 Len=1338 TSval=21662231
34452 7.353277000  194.47.150.244   192.168.0.141    TCP        66 8012 > 44012 [ACK] Seq=1 Ack=1297985 Win=729216 Len=0 TSval=1104108 T
34453 7.353322000  192.168.0.141    194.47.150.244   TCP      1404 44012 > 8012 [ACK] Seq=1388969 Ack=1 Win=29312 Len=1338 TSval=2166223
34571 7.365064000  192.168.0.142    194.47.150.244   TCP      1404 44006 > 8012 [ACK] Seq=425866 Ack=1 Win=29312 Len=1338 TSval=21662235
34572 7.365072000  192.168.0.142    194.47.150.244   TCP      1404 44006 > 8012 [ACK] Seq=427204 Ack=1 Win=29312 Len=1338 TSval=21662235
34573 7.367527000  194.47.150.244   192.168.0.142    TCP        66 8012 > 44012 [ACK] Seq=1 Ack=1108884 Win=468096 Len=0 TSval=1104111 T
34574 7.367575000  192.168.0.142    194.47.150.244   TCP      1404 44012 > 8012 [ACK] Seq=1153038 Ack=1 Win=29312 Len=1338 TSval=2166223
34575 7.367586000  192.168.0.142    194.47.150.244   TCP      1404 44012 > 8012 [ACK] Seq=1154376 Ack=1 Win=29312 Len=1338 TSval=2166223

0000  bc 30 5b c6 60 52 00 1e  4f 91 0e 48 08 00 45 00   .0[.`R.. O..H..E.
0010  0b 84 9b 05 40 00 40 06  e2 29 c2 2f 96 f1 c2 2f   ....@.@. .)./.../
0020  96 f4 ab e3 1f 4c a2 16  7e d3 a1 cd 77 91 80 10   .....L.. ~...w...
0030  00 e5 bd bb 00 00 01 01  08 0a 01 4a 89 d4 00 10   ........ ...J....

usb0, eth1, usb1: <live capture i...   Packets: 48788 · Displayed: 95 (0,2%) · Marked: 95 (0,...   Profile: Default
```

Figure 5.1: Packets sent by source shown in Wireshark

Figure 5.2 represents the packets that are received at the destination. Each field marked with a color represents packets received from different source IP addresses. As the IP addresses of the wireless interfaces used at the source are private IP's Wireshark displays the public IP addresses of them. As shown, packets are received from IP addresses 194.47.150.241 which represent packets that are sent through Ethernet interface at source. 79.138.132.158 and 79.138.131.215 are the public IP addresses of the wireless interfaces. This shows that the packets are received in random order irrespective of the interface through which they are transmitted due to various delays and speed present for that particular network. This proves that the packets are received concurrently at the destination and hence Concurrent Multipath Transfer is aptly working in the testbed that is setup. Even when multiple copies of the same packet are being sent through different paths only the one that arrives first is considered and the rest are discarded.



```
No.   Time        Source          Destination     Protocol Length Info
29770 23.676547000 194.47.150.241 194.47.150.244  TCP   1514 44009 > 8012 [ACK] Seq=5641448 Ack=1 Win=29312 Len=1448 TSval=21662018
29771 23.676780000 194.47.150.241 194.47.150.244  TCP   1514 44009 > 8012 [ACK] Seq=5642896 Ack=1 Win=29312 Len=1448 TSval=21662018
29772 23.676790000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44009 [ACK] Seq=1 Ack=5644344 Win=254592 Len=0 TSval=1103901 TSe
29773 23.676795000 79.138.132.158 194.47.150.244  TCP   1404 29551 > 8012 [ACK] Seq=525889 Ack=1 Win=29312 Len=1338 TSval=21661974 T
29774 23.677030000 194.47.150.241 194.47.150.244  TCP   2962 44009 > 8012 [ACK] Seq=5644344 Ack=1 Win=29312 Len=2896 TSval=21662018
29775 23.677060000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44009 [ACK] Seq=1 Ack=5647240 Win=255360 Len=0 TSval=1103901 TSe
29776 23.677279000 79.138.131.215 194.47.150.244  TCP   1404 49417 > 8012 [ACK] Seq=725197 Ack=1 Win=29312 Len=1338 TSval=21661984 T
29777 23.677289000 194.47.150.241 194.47.150.244  TCP   1514 44009 > 8012 [ACK] Seq=5647240 Ack=1 Win=29312 Len=1448 TSval=21662018
29778 23.677528000 194.47.150.241 194.47.150.244  TCP   2962 44009 > 8012 [ACK] Seq=5648688 Ack=1 Win=29312 Len=2896 TSval=21662018
29779 23.677557000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44009 [ACK] Seq=1 Ack=5651584 Win=255360 Len=0 TSval=1103901 TSe
29780 23.677770000 194.47.150.241 194.47.150.244  TCP   1514 44009 > 8012 [ACK] Seq=5651584 Ack=1 Win=29312 Len=1448 TSval=21662018
29781 23.677788000 194.47.150.241 194.47.150.244  TCP   1514 44000 > 8012 [ACK] Seq=5167326 Ack=1 Win=29312 Len=1448 TSval=21662018
29782 23.678027000 194.47.150.241 194.47.150.244  TCP   2962 44000 > 8012 [ACK] Seq=5168774 Ack=1 Win=29312 Len=2896 TSval=21662018
29783 23.678057000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44000 [ACK] Seq=1 Ack=5171670 Win=260736 Len=0 TSval=1103901 TSe
29784 23.678274000 194.47.150.241 194.47.150.244  TCP   1514 44000 > 8012 [PSH, ACK] Seq=5171670 Ack=1 Win=29312 Len=1448 TSval=2166
29785 23.678292000 79.138.132.158 194.47.150.244  TCP   1404 29551 > 8012 [ACK] Seq=527227 Ack=1 Win=29312 Len=1338 TSval=21661974 T
29786 23.678296000 79.138.131.215 194.47.150.244  TCP   1404 49417 > 8012 [ACK] Seq=726535 Ack=1 Win=29312 Len=1338 TSval=21661984 T
29787 23.678302000 194.47.150.244 79.138.131.215  TCP     66 8012 > 49417 [ACK] Seq=1 Ack=727873 Win=356224 Len=0 TSval=1103901 TSec
29788 23.678343000 194.47.150.244 79.138.132.158  TCP     66 8012 > 29551 [ACK] Seq=1 Ack=528565 Win=435712 Len=0 TSval=1103901 TSec
29789 23.678526000 194.47.150.241 194.47.150.244  TCP   2962 44000 > 8012 [ACK] Seq=5173118 Ack=1 Win=29312 Len=2896 TSval=21662018
29790 23.678559000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44000 [ACK] Seq=1 Ack=5176014 Win=260736 Len=0 TSval=1103902 TSe
29791 23.678776000 194.47.150.241 194.47.150.244  TCP   2962 44000 > 8012 [ACK] Seq=5176014 Ack=1 Win=29312 Len=2896 TSval=21662018
29792 23.678807000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44000 [ACK] Seq=1 Ack=5178910 Win=260736 Len=0 TSval=1103902 TSe
29793 23.679024000 194.47.150.241 194.47.150.244  TCP   2962 44000 > 8012 [ACK] Seq=5178910 Ack=1 Win=29312 Len=2896 TSval=21662019
29794 23.679056000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44000 [ACK] Seq=1 Ack=5181806 Win=260736 Len=0 TSval=1103902 TSe
29795 23.679273000 79.138.132.158 194.47.150.244  TCP   1404 29551 > 8012 [ACK] Seq=528565 Ack=1 Win=29312 Len=1338 TSval=21661974 T
29796 23.679283000 79.138.131.215 194.47.150.244  TCP   1404 49371 > 8012 [ACK] Seq=593754 Ack=1 Win=29312 Len=1338 TSval=21661984 T
29797 23.679521000 194.47.150.241 194.47.150.244  TCP   2962 44006 > 8012 [ACK] Seq=5957588 Ack=1 Win=29312 Len=2896 TSval=21662019
29798 23.679553000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44006 [ACK] Seq=1 Ack=5960484 Win=417024 Len=0 TSval=1103902 TSe
29799 23.679772000 194.47.150.241 194.47.150.244  TCP   2962 44006 > 8012 [ACK] Seq=5960484 Ack=1 Win=29312 Len=2896 TSval=21662019
29800 23.679802000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44006 [ACK] Seq=1 Ack=5963380 Win=417024 Len=0 TSval=1103902 TSe
29801 23.680023000 194.47.150.241 194.47.150.244  TCP   2962 44006 > 8012 [ACK] Seq=5963380 Ack=1 Win=29312 Len=2896 TSval=21662019
29802 23.680052000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44006 [ACK] Seq=1 Ack=5966276 Win=417024 Len=0 TSval=1103902 TSe
29803 23.680270000 79.138.132.158 194.47.150.244  TCP   1404 29551 > 8012 [ACK] Seq=529903 Ack=1 Win=29312 Len=1338 TSval=21661974 T
29804 23.680280000 79.138.131.215 194.47.150.244  TCP   1404 49371 > 8012 [ACK] Seq=595092 Ack=1 Win=29312 Len=1338 TSval=21661984 T
29805 23.680300000 194.47.150.244 79.138.132.158  TCP     66 8012 > 29551 [ACK] Seq=1 Ack=531241 Win=435712 Len=0 TSval=1103902 TSec
29806 23.680323000 194.47.150.244 79.138.131.215  TCP     66 8012 > 49371 [ACK] Seq=1 Ack=596430 Win=258560 Len=0 TSval=1103902 TSec
29807 23.680520000 194.47.150.241 194.47.150.244  TCP   2962 44006 > 8012 [ACK] Seq=5966276 Ack=1 Win=29312 Len=2896 TSval=21662019
29808 23.680550000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44006 [ACK] Seq=1 Ack=5969172 Win=417024 Len=0 TSval=1103902 TSe
29809 23.680767000 194.47.150.241 194.47.150.244  TCP   2962 44012 > 8012 [PSH, ACK] Seq=5677767 Ack=1 Win=29312 Len=2896 TSval=2166
29810 23.680798000 194.47.150.244 194.47.150.241  TCP     66 8012 > 44012 [ACK] Seq=1 Ack=5680663 Win=237568 Len=0 TSval=1103902 TSe

0000  bc 30 5b c6 60 52 00 10  db ff 10 00 08 00 45 00   .0[.`R.. ......E.
0010  05 6e 6e e5 40 00 36 06  a4 1f 4f 8a 83 d7 c2 2f   .nn.@.6. ..O..../
0020  96 f4 78 e5 1f 4c 56 d8  13 4a 10 80 00 53 80 10   ..x..LV. .J...S..
0030  00 e5 c7 b0 00 00 01 01  08 0a 01 4a 8b b8 00 10   ........ ...J....

File: "/tmp/wireshark_pcapng_...   Packets: 49563 · Displayed: 49563 (100,0%) · Dropped: 0 (0,0%)   Profile: Default
```

Figure 5.2: Packets recieved at destination shown in Wireshark

5.1.2 Throughput Variance in CMT

When a file of 100MB (104857600 Bytes) is sent from source to destination varying the number of paths for transmission, throughput is measured and plotted. This is shown in Figure 5.3. There is a drastic increase in the throughput measured when two paths (1 Ethernet and 1 USB) are used concurrently compared to the throughput obtained when only one path is used. However, when three paths are used there is a very slight increase in the throughput when compared to two paths. One of the major advantages of the Concurrent Multipath Transfer is that it can achieve high throughput data transmissions. This aids to the theoretical statement.

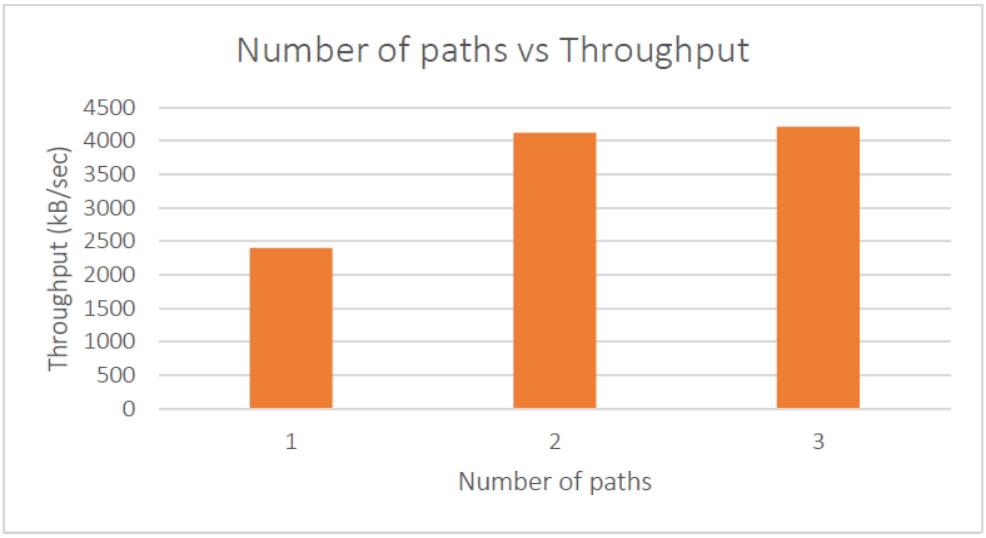


Figure 5.3: Throughput measurements depending on number of paths

5.2 Section-2

The data that is transmitted is split into smaller chunks and is sent. The test bed setup and the parameters considered for this experiment are described in detail in the previous chapter. The size of each chunk the data is split into is made configurable. Throughput is measured varying the size of the chunk. This experiment is conducted number of times and the mean of them is considered and plotted for more reliability of results. Graphs are plotted in two scenarios using Round-Robin (RRB) mechanism and Weighted Round-Robin (WRRB) mechanism. The experiment can be considered in two ways one considering the size of the chunk and the other the total number of packets being transmitted.

5.2.1 Varying size of packet

As shown in graph 5.4, RRB and WRRB mechanism exhibits linear behavior and almost similar throughput with very slight difference in the orders of tens as the chunk size is varied from 1000 Bytes to 5000 Bytes. After reaching an optimal point of 5000 Bytes throughput remains in a steady state for RRB whereas in WRRB it is linearly increasing until it reached the optimum at 8000 Bytes and is in steady state. However, there is drastic change between both the mechanisms. WRRB obtained way higher throughput when compared to RRB

This behavior is due to increase in priority of more reliable path in the case of WRRB. The difference between both the mechanisms can be clearly observed

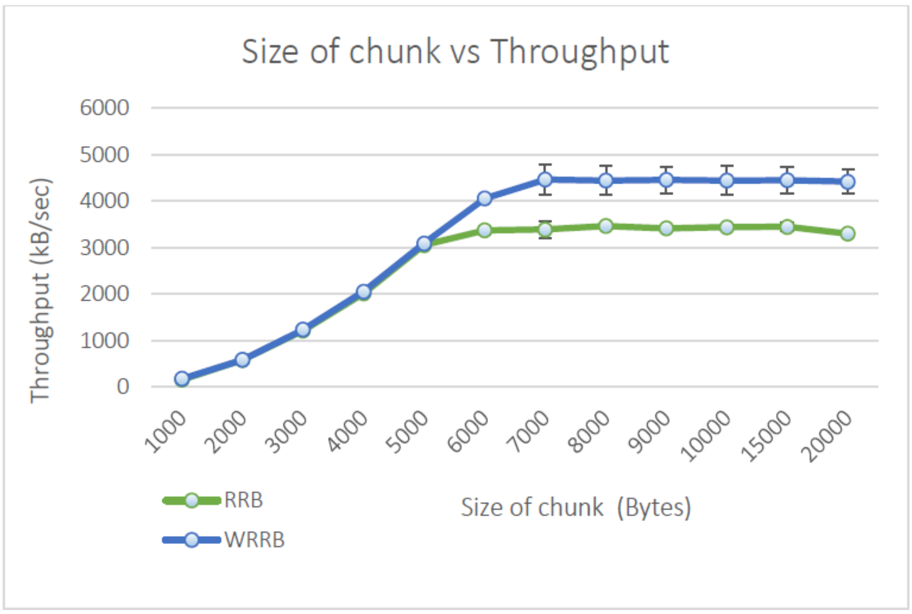


Figure 5.4: Throughput measured varying size of chunk

when the size of the chunk is higher. It may be possibly because of the fragmentation of packets in the lower layers when the size of the packet is higher which leads to higher network overhead. So, when WRRB mechanism is used priority is given to the path which is more reliable. Hence, throughput obtained is higher in this case when compared to RRB mechanism.

5.2.2 Impact due to number of packets

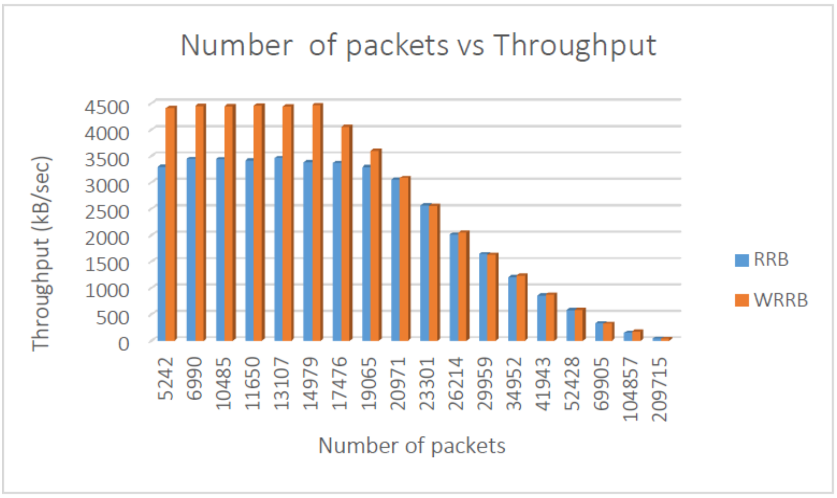


Figure 5.5: Throughput measured varying total number of packets

We can deduct from the graph below the difference in throughput when RRB and WRRB mechanisms are used for scheduling the packets before transmission. WRRB obtained comparatively very high throughput than the throughput obtained by RRB when the total number of packets is less. However, when the number of packets that are transmitted increases it is observed that there is barely any difference between the two of them.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, principal scheduling mechanisms in CMT were implemented in a real-time scenario. The main goals of this work was to implement CMT with de-multiplexer at the source to fragment data, to implement a re-sequencer at the destination for packet re-ordering, to evaluate different scheduling patterns for CMT with respect to throughput. Two scheduling mechanisms namely, Round-robin and Weighted Round-robin mechanisms were studied and their performance was analyzed.

After the implementation of CMT, the packets were analyzed on Wireshark to prove that packets are being sent concurrently from the source and are received at the destination. These results were shown and discussed in Chapter 5. It is also observed that when multiple copies of the same packet are transmitted only the one that arrives first is kept is stored in the re-sequencing buffer and the others are discarded. Performance of the scheduling mechanisms is evaluated by varying the size of the chunk and measuring the throughput. The results were discussed and included in the second part of the results section.

Certain observations can be made by analyzing the behavior of throughput. Compared to Round-Robin, Weighted Round-Robin has high throughput as the size of the packets are increased though they are almost equal for small packet sizes. In both the mechanisms they exhibit steady state behavior after reaching an optimal point. It also implies that when the number of packets are increased the throughput decreased. There is a huge difference in throughput as the range varied. It can possibly be due to network overhead caused because of fragmentation when there are number of packets being transmitted which takes more time for processing at the network layer.

It can be concluded that Weighted Round-Robin mechanism performs relatively better than that of the Round-Robin when large packet sizes are considered whereas both the mechanisms perform similar in case of small packets. It is also proved that when a packet is transmitted more than once, only the one which arrives first is considered and the rest are discarded.

When Concurrent Multipath Transfer is implemented there may be higher network utilization for network operators. Network utilization is the ratio of current network traffic to the maximum traffic. Hence, when CMT is implemented by them even when there are no signals for one network it can make use of other networks by collaboration which improves the current network traffic and in turn also increases Network utilization.

6.2 Future Work

Although we evaluated Round-Robin and Weighted Round-Robin scheduling mechanisms, there are several other mechanisms being developed by many researchers. They need to be implemented and evaluated on the real-time test bed setup. There are various other parameters like delay, jitter etc. which can also be measured and evaluated in these scenarios. In this work, 4G USB modems of the same type at the source and a single-homed destination were used for the implementation. However, investigations can be made making use of different wireless networks and multi-homed destination in real-time scenarios which might bring more challenges in configuring.

References

[1] “Goals and Benefits of Multihoming.” [Online]. Available: http://www.nautilus6.org/doc/drafts/draft-ernst-generic-goals-and-benefits-02.html.[Accessed: 30-Jul-2015]

[2] Y. Nebat and M. Sidi, “Resequencing considerations in parallel downloads,” in IEEE INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings, 2002, vol. 3, pp. 1326–1335 vol.3.

[3] T. Wallace, “Concurrent Multipath Transfer: Scheduling, Modelling, and Congestion Window Management,” Electron. Thesis Diss. Repos., Mar. 2012.

[4] T. Zinner, D. Klein, K. Tutschku, T. Zseby, P. Tran-Gia, and Y. Shavitt, “Performance of concurrent multipath transmissions #x2014; Measurements and model validation,” in 2011 7th EURO-NGI Conference on Next Generation Internet (NGI), 2011, pp. 1–8.

[5] S. K. Tedla, “Performance Evaluation of Concurrent Multipath Transmission – Measurement and Analysis,” Blekinge Institute of Technology, Supervisor: Prof.Dr Kurt Tutschku, 2015.

[6] B. Forouzan, C. Coombs, and S. C. Fegan, Introduction to Data Communications and Networking. New York, NY, USA: McGraw-Hill, Inc., 1998.

[7] A. Jayaraman, “Concurrent Multi-Path Real-Time Transmission Control Protocol,” Open Access Theses, Jan. 2007.

[8] S´. Barr´e, “Implementation and Assessment of Modern Host-based Multipath Solutions,” Universit´e catholique de Louvain, Louvain-la-Neuve,Belgium, 2011.

[9] “Multipath TCP.” [Online]. Available: https://queue.acm.org/detail.cfm?id=2591369.[Accessed: 31-Jul-2015].

[10] R. van der Pol, S. Boele, F. Dijkstra, A. Barczyk, G. van Malenstein, J. H. Chen, and J. Mambretti, “Multipathing with MPTCP and OpenFlow,” in High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:, 2012, pp. 1617–1624.

[11] K. Tutschku, T. Zinner, A. Nakao, and P. Tran-Gia, “Network Virtualization: Implementation Steps Towards the Future Internet,” p. 14.

[12] J. Liao, J. Wang, T. Li, and X. Zhu, “Introducing multipath selection for concurrent multipath transfer in the future internet,” Comput. Netw., vol. 55, no. 4, pp. 1024–1035, Mar. 2011.

[13] R. Rodrigues and P. Druschel, “Peer-to-peer Systems,” Commun ACM, vol. 53, no. 10, pp. 72–82, Oct. 2010.

[14] T. Zinner, K. Tutschku, A. Nakao, and P. Tran-Gia, “Using concurrent multipath transmission for Transport Virtualization: Analyzing path selection,” in Teletraffic Congress (ITC), 2010 22nd International, 2010, pp. 1–7.

[15] N. M. M. K. Chowdhury and R. Boutaba, “Network virtualization: state of the art and research challenges,” IEEE Commun. Mag., vol. 47, no. 7, pp. 20–26, Jul. 2009.

[16] S. Tian, J. Liao, J. Wang, J. Wang, Q. Qi, and L. Zhang, “Load-balanced one-hop overlay source routing over shortest path,” in Global Information Infrastructure Symposium, 2013, 2013, pp. 1–3.

[17] K. Tutschku, T. Zinner, A. Nakao, and P. Tran-Gia, “Re-sequencing Buffer Occupancy of a Concurrent Multipath Transmission Mechanism for Transport System Virtualization,” in Kommunikation in Verteilten Systemen (KiVS), K. David and K. Geihs, Eds. Springer Berlin Heidelberg, 2009, pp. 303–308.

[18] J. Wu, B. Cheng, C. Yuen, Y. Shang, and J. Chen, “Distortion-Aware Concurrent Multipath Transfer for Mobile Video Streaming in Heterogeneous Wireless Networks,” IEEE Trans. Mob. Comput., vol. 14, no. 4, pp. 688–701, Apr. 2015.

[19] J. R. Iyengar, P. D. Amer, and R. Stewart, “Concurrent Multipath Transfer Using SCTP Multihoming Over Independent End-to-End Paths,” IEEEACM Trans. Netw., vol. 14, no. 5, pp. 951–964, Oct. 2006.

[20] F. Perotto, C. Casetti, and G. Galante, “SCTP-based Transport Protocols for Concurrent Multipath Transfer,” in IEEE Wireless Communications and Networking Conference, 2007.WCNC 2007, 2007, pp. 2969–2974.

[21] H. Adhari, T. Dreibholz, M. Becke, E. P. Rathgeb, and M. Tüxen, "Evaluation of Concurrent Multipath Transfer over Dissimilar Paths," in 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications (WAINA), 2011, pp. 708–714.

[22] M. Becke, H. Adhari, E. P. Rathgeb, F. Fa, X. Yang, and X. Zhou, "Comparison of Multipath TCP and CMT-SCTP based on intercontinental measurements," in 2013 IEEE Global Communications Conference (GLOBECOM), 2013, pp. 1360–1366.

[23] M.-F. Tsai, N. K. Chilamkurti, S. Zeadally, and C.-K. Shieh, "A Concurrent Multi-path Transmission Control Scheme to reduce packet reordering latency at the receiver," in International Conference on Advanced Technologies for Communications, 2008. ATC 2008, 2008, pp. 437–440.

[24] J. Wang, J. Liao, and T. Li, "OSIA: Out-of-order Scheduling for In-order Arriving in concurrent multi-path transfer," J. Netw. Comput. Appl., vol. 35, no. 2, pp. 633–643, Mar. 2012.

[25] 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), 2014, pp. 1222–1227.

[26] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental Evaluation of Multipath TCP Schedulers," in Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop, New York, NY, USA, 2014, pp. 27–32.

[27] "Research Methodology in Computer Science." [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.260.529&rep=rep1&type=pdf.[Acc 09-Jul-2015].

[28] "Network Time Protocol." [Online]. Available: https://en.wikipedia.org/wiki/Network_Time_Protocol.[Accessed: 09-Jul-2015].

[29] R. Buyya, Object Oriented Programming with Java: Essentials and Applications. McGraw-Hill Education (India) Pvt Ltd New Delhi, India.

[30] "IP rule." [Online]. Available: http://linux-ip.net/html/tools-ip-rule.html.[Accessed: 28-Aug-2015].

APPENDIX

Size of packet (Bytes)	**Throughput (kB/sec)**	
	Round-Robin	**Weighted Round-Robin**
500	41.81	39.54
1000	153.128	176.597
1500	332.918	325.096
2000	578.664	582.483
2500	857.507	873.016
3000	1210.344	1234.246
3500	1643.588	1629.826
4000	2011.32	2053.257
4500	2564.395	2555.429
5000	3052.923	3087.028
5500	3294.36	3599.679
6000	3370.482	4055.893
7000	3387.114	4465.263
8000	3464.274	4441.448
9000	3414.742	4456.348
10000	3439.892	4444.296
15000	3446.825	4448.634
20000	3300.384	4415.992

Table 1: Throughput varying size of each packet

Number of paths	**Throughput Obtained (kB/sec)**
1	2396.51
2	4123.65
3	4211.22

Table 2: Throughput varying number of paths