# FMTCP: A FOUNTAIN CODE-BASED MULTIPATH TRANSMISSION CONTROL PROTOCOL

Duda Prasad[1], E.Amareswar[2]
[1]Asst Professor, St. Martin's Engineering College, Hyderabad
[2]Asst Professor, MLR Institute of Technology, Hyderabad.

**Abstract**

**Ideally, the throughput of a Multipath Transmission Control Protocol (MPTCP) connection should be as high as that of multiple disjoint single-path TCP flows. In reality, the throughput of MPTCP is far lower than expected. The results indicate that a sub flow experiencing high delay and loss severely affects the performance of other sub flows, thus becoming the bottleneck of the MPTCP connection and significantly degrading the aggregate good put.**

**Fountain code-based Multipath TCP (FMTCP), which effectively mitigates the negative impact of the heterogeneity of different paths. FMTCP takes advantage of the random nature of the fountain code to flexibly transmit encoded symbols from the same or different data blocks over different sub flows. Moreover, we design a data allocation algorithm based on the expected packet arriving time and decoding demand to coordinate the transmissions of different sub flows. Quantitative analyses are provided to show the benefit of FMTCP. FMTCP achieves high stability under abrupt changes of path quality.**

**Keywords: MPTCP (Multipath Transmission Control Protocol), FMTCP -Fountain code-based Multipath TCP**.

## INTRODUCTION

Currently, the majority of data transmissions go through Transmission Control Protocol (TCP). In a network with high loss and delay, such as a wireless network, the performance of TCP degrades significantly due to frequent retransmissions of lost or erroneous packets. In addition, a user may want to transmit data at a higher aggregate throughput when having multiple access links to the network. However, conventional TCP cannot enjoy the multi homing feature. In order to solve these problems, Multipath Transmission Control Protocol (MPTCP)[1]-[3] has been proposed to transmit TCP simultaneously over multiple paths to improve the good put and reliability. However, if the paths have high diversity in quality (i.e., with different loss or delay), the good put of MPTCP degrades sharply. When a receiver waits for a packet sent on a low-quality path, the receiver buffer may be filled up. Thus, even if other paths have good quality, they cannot send more packets, and the low-quality paths become the bottlenecks of MPTCP. To solve the bottleneck problem, some attempts have been made to improve the throughput over lossy networks.

Some studies show that the good put of MPTCP can be even worse than that of an ordinary TCP in some cases. To solve the bottleneck problem, some attempts [5], [6] have been made to improve the throughput over lossy networks. However, if a large number of packets need to be retransmitted due to the high loss rate, a high overhead to schedule packet retransmissions would be incurred. It is also very difficult to coordinate transmissions among all paths.

Fountain code is a linear random code for channels with erasures, and its elemental data unit is symbol formed with a certain number of bits. An encoding symbol is generated based on random linear combination of the source symbols in a data block. With its low complexity and redundancy, different fountain codes are considered for use in different transmission standards. The most advanced version of a practical fountain code has been standardized in IETF RFC6330 to provide reliable delivery of data objects. As a key advantage, the original

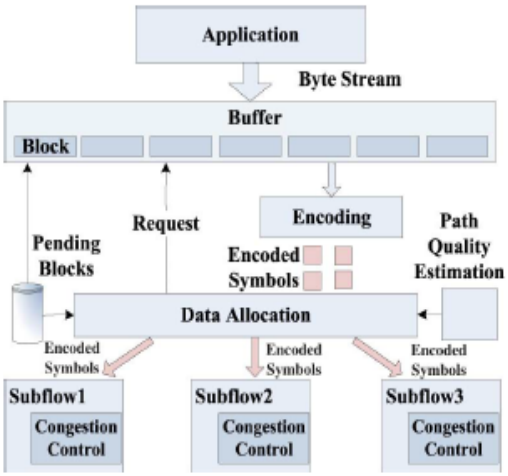
data can be encoded into an arbitrary number of symbols based on the transmission quality and the estimated number of symbols to use for data recovery. The receiver can recover the original data with high accuracy after obtaining enough number of the encoded symbols.

Taking advantage of fountain codes, a sender in FMTCP generates *new encoded symbols* for a block based on the remaining number of symbols needed for reliable decoding at the receiver. Instead of retransmitting the lost packets along the same path on which packet loss is detected, new symbols from the same or different blocks are put into one or multiple packets. Packets are flexibly allocated to different TCP sub flows for transmissions based on the packet arrival time estimated according to the transmission quality of each flow; as a result, the low-quality paths will no longer be the bottlenecks of the overall multipath TCP transmission. As only randomly generated symbols are needed for decoding, there is no need for FMTCP to coordinate transmissions on different paths, which not only significantly reduces the complexity of scheduling, but also reduces the gap in transmission time on diverse paths. This will in turn significantly improve the TCP performance

**Advantages:**

1. Reduces the gap in transmission time on diverse paths.
2. Improve the TCP performance.
3. Higher and more stable performance

## BLOCK DIAGRAM OF THE SYSTEM




**Modules Description:**

1. FMTCP ARCHITECTURE
2. ENCODING MODULE
3. CODE SELECTION FOR FMTCP MODULE
4. DATA ALLOCATION MODULE

## FMTCP ARCHITECTURE

The sender-side architecture of FMTCP is proposed here. We introduce the fountain code into the transport layer and transmit encoded data via multiple paths. A byte stream from applications is divided into blocks, which are taken as the input of the encoding module inserted on top of the data allocation module. After the encoding, each block is converted to a series of encoded symbols, which are carried in packets and transmitted to the receiver. On the receiver side, encoded symbols are converted back to the original data through a decoding module appended on top of the data aggregation module. Once decoded, the data can be transmitted to the application layer, and the corresponding symbols can be removed from the receiving buffer

## ENCODING MODULE

We focus on the packet scheduling part that breaks the byte stream received from the application into segments to transmit on different available sub flows. Before transmission, the coding module encodes the segment, and the data allocation module will determine which sub flow the segment will be assigned to based on the path quality estimation

## CODE SELECTION FOR FMTCP MODULE

We are transmitting data via different paths and these paths may have very different quality, it is possible that the low quality paths block the high-quality ones, causing increasing transmission delay and low good puts. Therefore, we introduce Forward Error Correction (FEC) code for channels with erasures into the transport layer to alleviate this problem. Here, FEC is not used to correct bit errors, but to recover data in lost packets that will be introduced later. In this way, even if packets are lost on some low-quality paths, the receiver may still be able to recover the original data, and thus the low-quality paths will not block the high-quality ones. There are basically two categories of FEC codes: fixed-rate and rate less.

5. Fixed Rate Coding
6. Rate less Coding

For a fixed-rate code, a block containing kb symbols is encoded into kb^ encoded symbols, where kb^/kb is the coding rate. It is guaranteed that any of the encoded symbols can
Recover the original file. Therefore, the receiver can successfully decode the data if the number of symbols lost is no larger than kb^-kb .To solve the problem of fixed-rate coding, rateless coding scheme is introduced. In rateless coding, an arbitrarily number of symbols can be generated for a block. If the receiver does not receive enough number of symbols due to dynamics of the channels, additional new symbols can be generated to formal overrate encoding instead of retransmitting the lost ones. Therefore, rateless coding has no fixed-rate and its rate adapts between 1 and 0.

**DATA ALLOCATION MODULE:**

FMTCP inherits the other functions of MPTCP. All MPTCP operations are signaled using optional TCP header fields, so does our FMTCP. To support coding, we can design a new MPTCP option in place of the Data Sequence Signal (DSS) option in MPTCP [3], where an 8-bit source block number and a 24-bit encoding symbol ID are used according to RFC6330 to identify the data in the packet instead of the data sequence number used in MPTCP
Intuitively, to improve the efficiency of FMTCP, data allocation is expected to achieve the following objectives.

• Low redundancy: In order to increase the good put, the sender should not transmit redundant encoded symbols to receiver once the receiver can recover the original block.

• Decoding in order: In order to ensure the delivery order, if a block is sent before a block, should be recovered by the receiver before. This is because only by recovering the first pending block can the receiver release its buffer in time and thus avoid the restriction on the aggregation data rate caused by the flow control.

• High decoding probability: In order to reduce the delivery delay for a block, the receiver should be able to recover the original data with a high probability. If the receiver is not able to recover the data, it needs to inform the sender to send more encoded symbols of the block.

**IMPLEMENTATION**

Implementation is the stage where the theoretical design is turned in to working system.
The most crucial stage is achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively.
The system can be implemented only after through testing is done and if it found to work according to the specification. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over and an evaluation of change over methods a part from planning. Two major tasks of preparing the implementation are education and training of the users and testing of the system.

The more complex the system being implemented, the more involved will be the systems analysis and design effort required just for implementation. The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The System may require some hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.
Implementation is the process of having systems personnel check out and put new equipment in to use, train users, install the new application, and construct any files of data needed to it.

Depending on the size of the organization that will be involved in using the application and the risk associated with its use, system developers may choose to test the operation in only one area of the firm, say in one department or with only one or two persons. Sometimes they will run the old and new systems together to compare the results. In still other situations, developers will stop using the old system one-day and begin using the new one the next. As we will see, each implementation strategy has its merits, depending on the business situation in which it is considered. Regardless of the implementation strategy used, developers strive to ensure that the system's initial use in trouble-free.

Once installed, applications are often used for many years. However, both the organization and the users will change, and the environment will be different over the weeks and months.

Therefore, the application will undoubtedly have to be maintained. Modifications and changes will be made to the software, files, or procedures to meet the emerging requirements.

## SYSTEM TESTING

Testing is the process of finding differences between the expected behavior specified by system models and the observed behavior implemented system. From modeling point of view, testing is the attempt of falsification of the system with respect to the system models. The goal of testing is to design tests that exercise defects in the system and to reveal problems.

The process of executing a program with intent of finding errors is called testing. During testing, the program to be tested is executed with a set of test cases, and the output of the program for the test cases is evaluated to determine if the program is performing as expected. Testing forms the first step in determining the errors in the program. The success of testing in revealing errors in program depends critically on test cases.

Strategic Approach to Software Testing:

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirements analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decreases the level of abstraction on each item

A Strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing will progress by moving outward along the spiral to integration testing , where the focus on the design and the concentration of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

Types of Testing

1. Black box or functional testing
2. White box testing or structural testing

## BLACK BOX TESTING

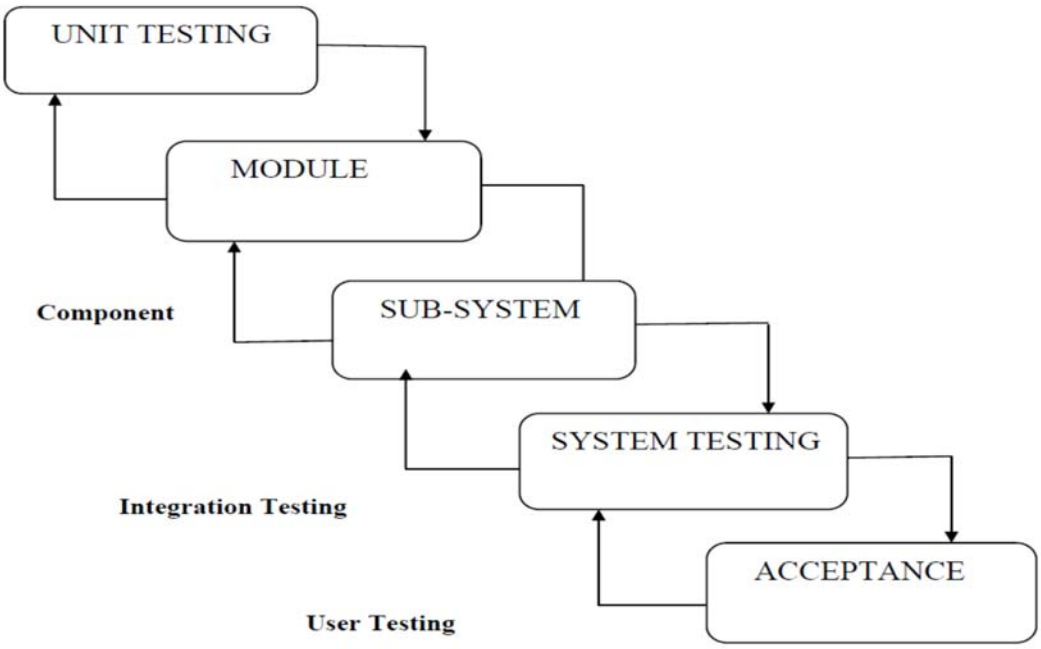
This method is used when knowledge of the specified function that a product has been designed to perform is known. The concept of black box is used to represent a system whose inside workings are not available to inspection

1. Black box testing attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structure
- Performance errors
- Initialization and termination errors

## WHITE BOX TESTING

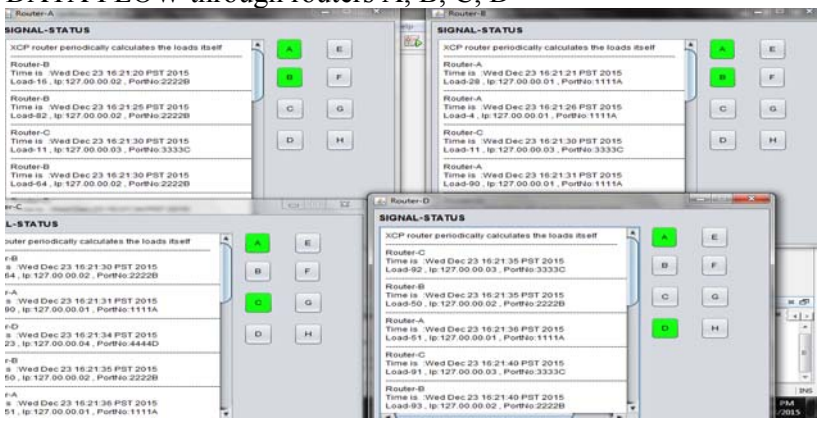
White box testing is concerned with testing the implementation of the program. The intent of structural is not to exercise all the inputs or outputs but to exercise the different programming and data structure used in the program.



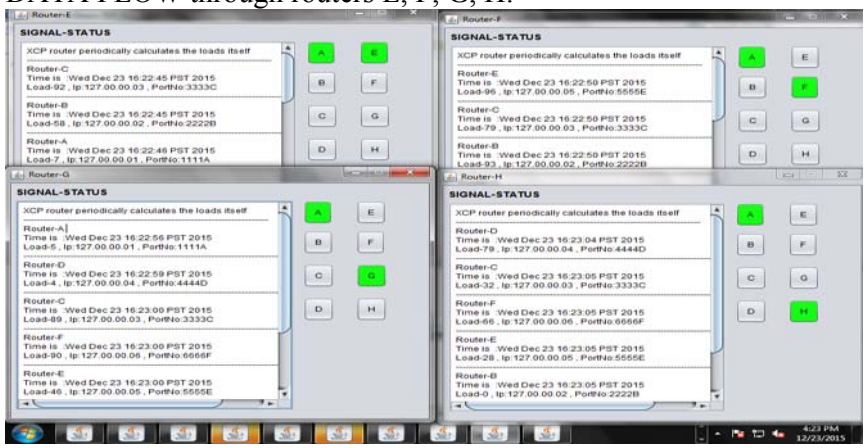

**Different Levels of Testing**

**SIMULATION:**

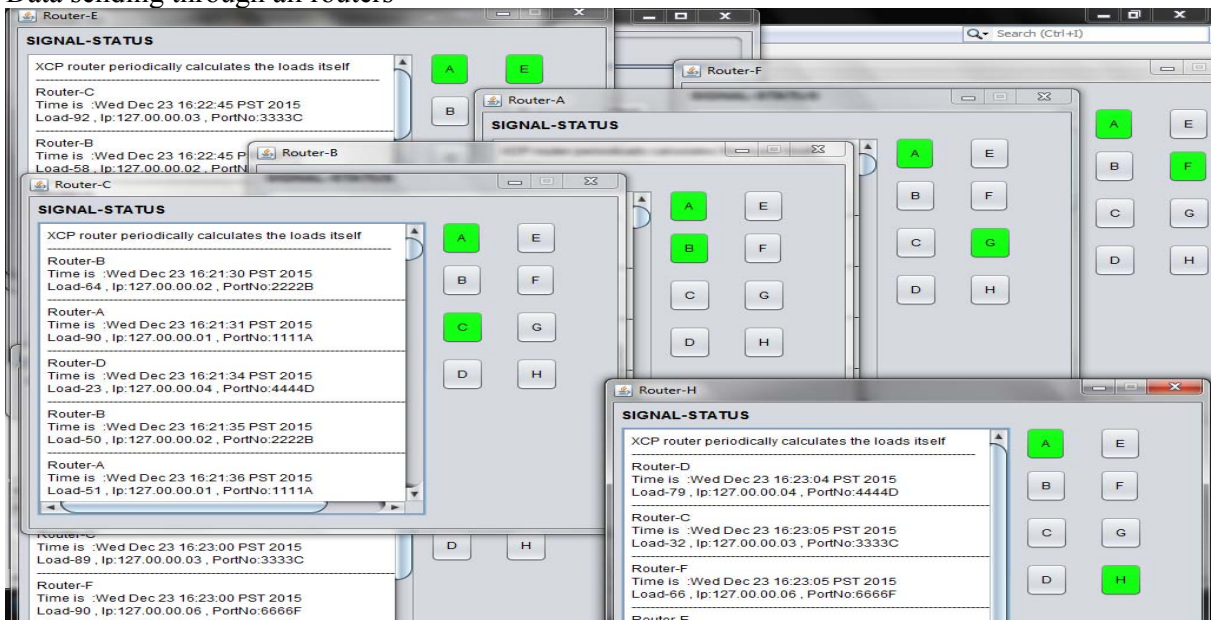
DATA FLOW through routers A, B, C, D



DATA FLOW through routers E, F, G, H.



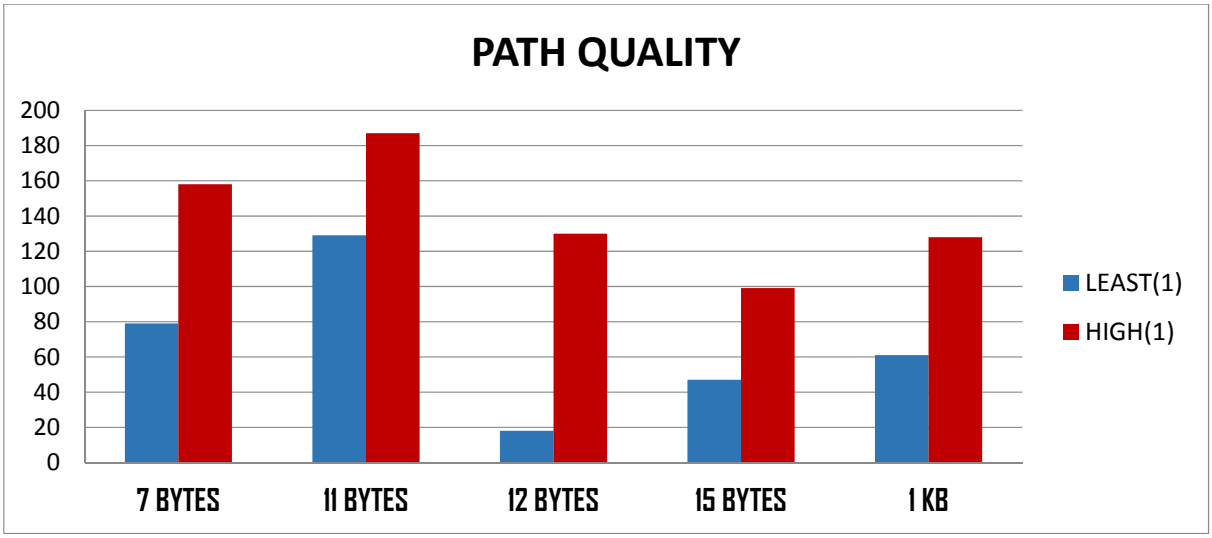
Data sending through all routers




## PATH QUALITY

| | 7 BYTES | 11 BYTES | 12 BYTES | 15 BYTES | 1 KB |
|---|---|---|---|---|---|
| LEAST(1) | 79 | 129 | 18 | 47 | 61 |
| HIGH(1) | 158 | 187 | 130 | 99 | 128 |

| PATH | PATH | PATH | PATH | PATH |
|---|---|---|---|---|
| R1R3 | R3R4 | R1R2 | R2R3 | R2R3 |
| R2R4 | R1R2 | R3R4 | R1R4 | R1R3 |



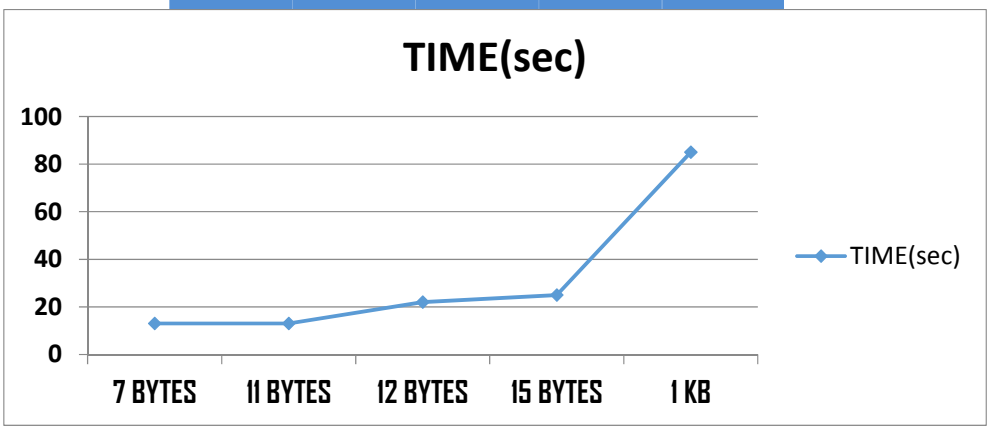

- The above graph indicates the path quality of a given byte of data.
- For knowing the path quality we have taken 7bytes, 11 bytes, 12bytes, 15bytes, 1kb of data .
- The x-axis indicates the size of the data i.e., bytes and the y-axis indicates the quality from 0-200.

Here we have showed least quality path and high quality path for each data byte.

- The path quality does not go in ascending or descending order as it checks the best path for every packet and data flows from the best packet.
- So if it gets the best quality path as fast as possible we get least quality path.

***TIME***

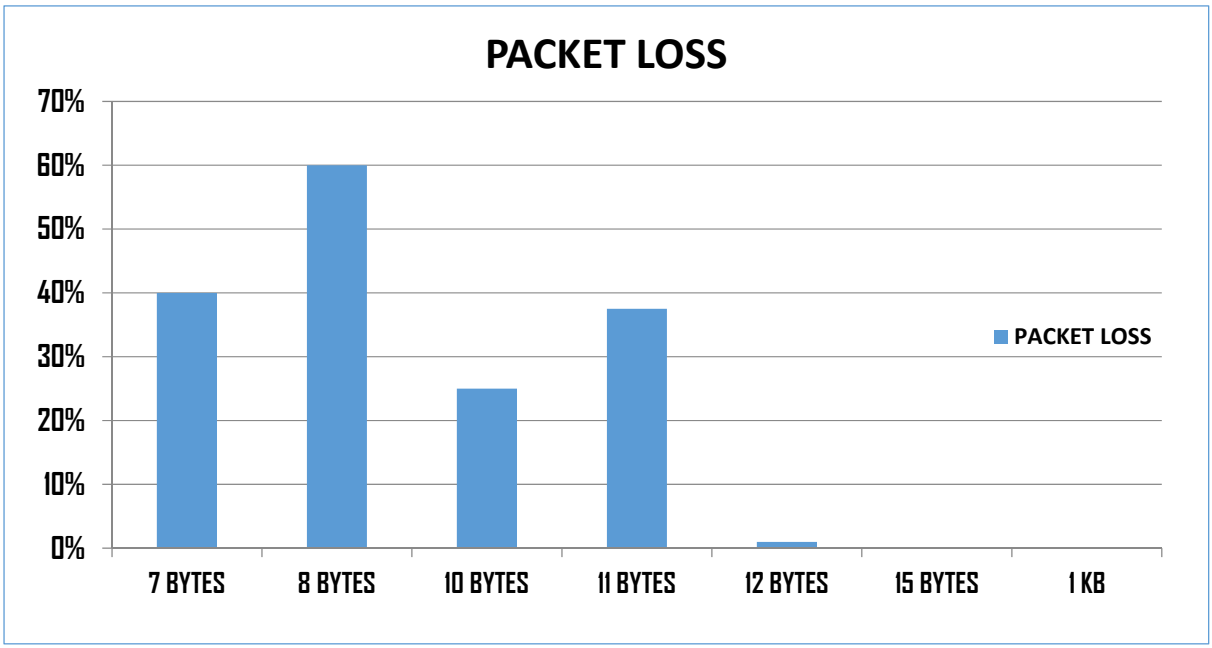
| TIME(sec) | 13 | 13 | 22 | 25 | 85 |
|---|---|---|---|---|---|




- The above graph indicates the time in which the data travels of a give byte of data.
- For know the path quality we have taken 7bytes, 11 bytes, 12bytes, 15bytes, 1kb of data .
- The x-axis indicates the size of the data taken i.e., bytes and the y-axis indicates the time in seconds.
- Here we can clearly see that as the size of the data increases the time taken also increases

## 4 PACKET LOSS

| | 7 BYTES | 8 BYTES | 10 BYTES | 11 BYTES | 12 BYTES | 15 BYTES | 1 KB |
|---|---|---|---|---|---|---|---|
| PACKET LOSS | 40% | 60% | 25% | 38% | 1% | 0% | 0% |



- The above graph indicates the packet loss of a give byte of data.
- The x-axis indicates the size of the data taken i.e., bytes and the y-axis indicates the percentage from 0%-70%.
- The packet loss is given by

$$\frac{\text{No. of lossed packets}}{\text{No. of received packets}}$$

- Here if we take 7bytes it has inefficient encoding data so it has more packet loss whereas for 15byes the packet loss is 0 as it has sufficient data

## REFERENCES

[1] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, "Design, implementation and evaluation of congestion control formulti path TCP," in Proc. 8th USENIX NSDI, 2011, p. 8.

[2] S.Barré,C.Paasch, and O.Bonaventure ,"MultipathTCP: fromtheory to practice," in Proc. Netw.,2011,pp.444–457.

[3] A.Ford, C.Raiciu, M.Handley, and O.Bonaventure, "TCPextensions formultipathoperationwithmultipleaddresses,"RFC6824,2013[Online]. Available: http://tools.ietf.org/html/rfc6824

[4] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M.Handley, "How hard can it be? Designing and implementing a deployable multipathTCP," in Proc.9thUSENIXNSDI, 2012, vol. 12, p. 29.

[5] H.HsiehandR.Sivakumar, "A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts," Wireless Netw. vol. 11, no. 1-2, pp. 99–114, 2005.

[6] K.Kim,Y.Zhu,R.Sivakumar, and H.Hsieh, "Areceiver-centric transport protocol for mobilehosts with heterogeneous wireless interfaces," Wireless Netw.,vol.11,no.4,pp.363–382,2005.

[7] D. MacKay, "Fountain codes," IEE Proc., Commun., vol. 152, no. 6, pp. 1062–1068, 2005.

[8] M.Luby, A.Shokrollahi, M.Watson, T.Stockhammer, and L.Minder, "Raptorq forward error correction scheme for object delivery," RFC6330,2011[Online].Available:http://tools.ietf.org/html/rfc6330

[9] J.Iyengar, P.Amer, and R. Stewart, "Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths,"IEEE/ACMTrans.Netw.,vol.14,no.5,pp.951–964,Oct.2006.

[10] T. Dreibholz, M. Becke, H. Adhari, and E. Rathgeb, "On the impact of congestioncontrol for concurrent multipath transfer on the transport layer," inProc.11thIEEEConTEL,2011,pp.397–404.

[11] Y.Cao,M.Xu,andX.Fu, "Delay-based congestion control for multipath TCP,"in Proc.20thIEEE ICNP,2012,pp.1–10.

[12] Y. Cao, M. Xu, X. Fu, and E. Dong, "Explicit multipath congestion control for data center networks," in Proc. 9th ACM CoNEXT, 2013, pp. 73–84.

[13] M.Becke, T.Dreibholz, H.Adhari, and E.Rathgeb, "Onthefairness of transport protocols in a multi-path environment," in Proc. IEEE ICC, Ottawa, ON, Canada, 2012, pp. 2666–2672.

[14] Y. Cui, X. Ma, H. Wang, I. Stojmenovic, and J. Liu, "A survey of energy efficient wireless transmission and modeling in mobile cloud computing,"MobileNetw.Appl.,vol.18,no.1,pp.148–155,2013.