# File Transfer Using Proxy Server


## Abstract

This project work is based on file transmission using MP QUIC for step 1 and implementation of the paper M. Kheirkhah and M. Lee, "AMP: An Adaptive Multipath TCP for Data Center Networks," 2019 (IFIP Networking), Warsaw, Poland, 2019, pp. 1-9, doi: 10.23919/IFIPNetworking.2019.8816848.

In Step 1 we have successfully transfered file from server to client using Multi Path QUIC transfer Protocol. We have used the decocninck mpquic repository.First we have downloaded all the required dependencies like go,quicgo minitopo and minitopo extensions. We have made a few changes in the code of client. In particular we have initialised the variable to store the response from the server.Then a new file is created which has same name as that of server and contents of the variable in it are placed in it.The file name variable is initialised as the last path of the url given by the client.Then using wireshark we test the transfer of file in single path as well as multipath.In Step 2 We have implemented the above mentioned paper by transforming multipath flow into single path flow.We have made few changes in the code in code of MPQUIC file transfer.We are still reading more papers and trying to work on the innovation.


## Step 1: Implementation on Multipath Topology

### Introduction

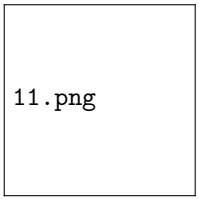
QUIC is a protocol which combines the functions of HTTP/2, TLS, and TCP directly over UDP to reduce the latency of client-server communication. MPQUIC, a major extension for QUIC that enables a QUIC connection across multiple subflows, keeping into consideration compatibility goals in order to achieve a seamless deployment of the new protocol on systems and infrastructures.

### Tools Required

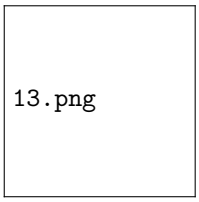
1.A Mininet VM.
2.Minitopo: Tool to build multipath networks over Mininet.
3.Minitopo experiences:Scripts running Minitopo experiences.
4.Wireshark: For capturing and Analysing packets.
5.Xming

## Installation

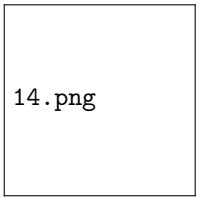
### Go



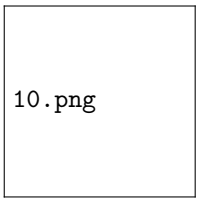

### QUIC go




### Minitopo Minitopo Experiences




## Code Modification




## Demonstration Steps

After setting up the environment, we compiled the server file and client file by using "gobuild" command. After successful compilation an executable file named main is created both in the server side and the client side.
Now we run the server using "go run example/main.go -www /var/www/". Upon starting the server gives us the port number it is currently listening

to.After getting the port number we can transfer any file from the server side using the same using "./main -m https://localhost:portNo/filename ". For single path transfer we can remove the -m in the command.

### Observations

When we use Single path for transfer we see that all the UDP packets are generated from the same source.

When we use multipath for tranferring upon analysing the packets we see that the packets are generated from multiple interfaces available on VM.

### Results

Upon Capturing the packets using wireshark we see that:

For single path Transmission: The source ip address of all the packets is same which is 127.0.0.1 with port number 43012.

Upon using the command ifconfig we get two interfaces eth0,eth1 and one loopback.

For multipath Transmission:The ip address of both eth0 and eth1 are communicating with the server.

### References

1.https://github.com/qdeconinck
2.https://github.com/lucas-clemente
3. B. Arzani, A. Gurney, S. Cheng, R. Guerin, and B. T. Loo. 2014. Deconstructing mptcp performance. In Network Protocols (ICNP), 2014 IEEE 22nd International Conference on. IEEE, 269–274.
4.R. Beverly, W. Brinkmeyer, M. Luckie, and J. P. Rohrer. 2013. IPv6 alias resolution via induced fragmentation. In PAM'13. Springer, 155–165.
5. A. Bittau, M. Hamburg, M. Handley, D. Mazieres, and D. Boneh. 2010. The Case for Ubiquitous Transport-Level Encryption.. In USENIX Security Symp osium. 403–418.
6. O. Bonaventure and S. Seo. 2016. Multipath TCP Deployments. IETF Journal (November 2016).

## Step 2:AMP: A Better Multipath TCP for Data Center Networks

### Abstract

In this project, we proposed two novel ECN-capable multipath congestion control algorithms for modern data center networks (DCNs). The first algorithm

is called Adaptive MultiPath (AMP) that is particularly designed to be robust against the TCP incast problem. It also coexists well with single-path flows like DCTCP, preventing the Last Hop Unfairness (LHU) problem that we've reported in this paper for the first time. In addition, the design of AMP is simple with low overhead. AMP moves traffic quickly from congested paths to less congested ones without sophisticated mechanisms such as RTT-dependent congestion window increase, as in standard MPTCP, or dynamic congestion window decrease, as in DCTCP. The second algorithm is called Data Center MultiPath (DCM), which integrates DCTCP with the standard MPTCP congestion control algorithm – Linked Increases Algorithm (LIA). To the best of our knowledge, we are the first to do such an integration.

## Introduction

Multipath data transport mechanisms such as MPTCP and XMP exist to effectively exploit the path diversity of data center networks (DCNs). However, these multipath schemes have not been widely deployed in DCNs. Two key factors among others impeded their adoption: TCP incast and minimum window syndrome.

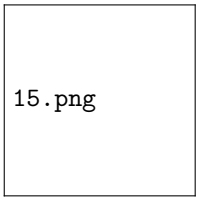
First, these mechanisms are ill-suited for workloads with a many-to-one communication pattern, commonly found in DCNs, causing frequent TCP incast collapses. Second, the syndrome we discover for the first time, results in 2-5 times lower throughput for single-path flows than multipath flows, thus severely violating network fairness. To effectively tackle these problems, we propose AMP: an adaptive multipath congestion control mechanism that quickly detects the onset of these problems and transforms its multipath flow into a single-path flow.

Once these problems disappear, AMP safely reverses this transformation and continues its data transmission via multiple paths. Our evaluation results under a diverse set of scenarios in a fat-tree topology with realistic workloads demonstrate that AMP is robust to the TCP incast problem and improves network fairness between multipath and single-path flows significantly with little performance loss.

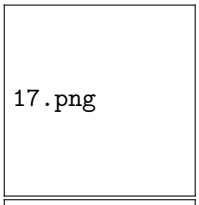
We now discuss the exact algorithm of AMP designed with the above four observations. AMP mainly consists of three components:

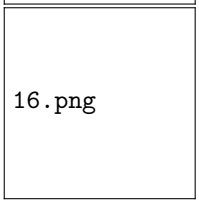
(i) subflow suppression/release (ii) constant factor decrease of congestion window (iii) RTT-agnostic congestion window increase The subflow suppression/release is a key mechanism that ensures graceful coexistence between multipath and single-path flows. The second component enables fast traffic shifting. The final part, as its name suggests, excludes RTT measurements, without any performance penalty, from the part of increasing cwnd, which over- all makes our algorithm simple.

## Algorithm




## Installation




## Implementation

As part of this project, we have implemented several networking protocols within ns-3.19.
A few of our implementations are listed below:
AMP : Adaptive MultiPath TCP
DCM : Data Center MultiPath TCP
XMP : eXplicit MultiPath
MPTCP : MultiPath TCP
MMPTCP : Maximum MultiPath TCP
DCTCP : Data Center TCP
ECN : Explicit Congestion Notification
ECMP : Equal-Cost Multi-Path

## Conclusion

In this paper we presented that existing multipath congestion control mechanisms fail to handle (1) the TCP incast problem that causes temporal switch buffer overflow due to synchronized traffic arrival; and (2) the minimum window syndrome that causes persistent buffer inflation and serious unfairness. To overcome the limitation of the existing solutions, we proposed AMP that

adaptively switches its operation between a multiple-subflow mode and single-subflow mode. Our extensive evaluation results showed that AMP is sim- ple yet effective to those problems and in general works well, which makes deploying AMP in data center better.

### References

1.M. Kheirkhah. MMPTCP: A Novel Transport Protocol for Data Centre Networks. PhD thesis, University of Sussex, 2016.
2.M. Kheirkhah, I. Wakeman, and G. Parisis. Multipath-TCP in ns-3. In Workshop on NS3, 2014.
3.Network Simulator https://www.nsnam.org/.
4.https://github.com/mkheirkhah
5.M. Kheirkhah. MMPTCP: A Novel Transport Protocol for Data Centre Networks. PhD thesis, University of Sussex, 2016.
6.M. Kheirkhah, I. Wakeman, and G. Parisis. MMPTCP: A Multipath Transport Protocol for Data Centers. In IEEE INFOCOM, 2016.
7.Y. Cao, M. Xu, X. Fu, and E. Dong. Explicit multipath congestion control for data center networks. In ACM CoNEXT, 2013.

## Submitted By

Group Number:ACN04

Vivek Joshi 2020H1030134H
Pratibha Ranade 2020H1030128H
Soumya Kothari 2020H1030124H
Omm Shree Atmaprakash 2020H1030107H