Simulasi Perbandingan Performa Multipath TCP dan TCP Pada Jaringan Wired

Krisna Kristiandi Hartono[1], Aji Gautama Putrada, S.T., M.T[2], Dr. Maman Abdurohman, S.T., M.T.[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]krisnakristiandi@students.telkomuniversity.ac.id,
[2]ajigps@telkomuniversity.ac.id,[3]abdurohman@telkomuniversity.ac.id

Abstrak

***TCP* yang berjalan pada *layer transport* yang umumnya di gunakan untuk pengiriman paket data, akan tetapi *TCP* dalam kondisi *Congestion* yang membebani *TCP*, *Throughput* yang di hasilkan tidak optimal. *TCP* hanya menerapkan satu jalur saat terjadi koneksi, sehingga sangat rentan saat terjadi *Congestion* yang menyebabkan *Packet Loss* , maka komunikasi transfer data akan berakhir. Untuk menanggulangi kondisi tersebut, diperlukan *Multipath TCP* dari pengembangan *TCP*. *Multipath TCP* dapat mengatasi hal tersebut dengan menggunakan metode *redundant* yaitu menggunakan antarmuka aktif dengan pengiriman secara sekaligus secara bersamaan, untuk jaringan *Wired* secara sekaligus yang harus di simulasikan terlebih dahulu sebelum di implementasikan. Uji coba menggunakan data hasil aktivitas (*Througput* dan *Packet Loss*). Dalam hasil simulasi NS-2 dan animasi Matlab dari data tersebut menerapkan metode *redundant* untuk *transfer* data secara bersamaan. Hasil uji coba jaringan *Wired* simulasi NS 2 dan animasi Matlab di dapat bahwa penggunaan *Multipath TCP* meningkatkan akurasi lebih besar dari *TCP* berdasarkan hasil percobaan beberapa *Delay RTT*, didapatkan hasil optimum pada *Delay RTT* 25 ms yaitu *Throughput* pada *Multipath TCP* sebesar 132.676 Kbps dan *TCP* 40.01 Kbps pada transfer data. Sedangkan pada *Packet Loss* pada *Multipat TCP* terendah pada *Delay RTT* 15 Ms yaitu 5.60% dan tertinggi pada *Delay RTT* 15 Ms pada jaringan *TCP* yaitu 54%.**

Kata kunci : ***Multipath TCP, TCP, Wired, Throughput, Delay RTT, Packet Loss, NS 2, Matlab.***

Abstract

TCP is running at the transport layer which is generally used for sending data packets, but TCP in Congestion conditions that burden TCP, throughput generated is not optimal. TCP only applies one path when there is a connection, so it is very vulnerable when Congestion occurs which causes Packet Loss, the data transfer communication will end. To overcome this condition, Multipath TCP is needed from the development of TCP. Multipath TCP can overcome this by using a redundant method that is using an active interface with sending simultaneously at the same time, for the Wired network at the same time which must be simulated first before being implemented. The trial uses activity result data (Througput and Packet Loss). In the NS-2 simulation results and Matlab animation of the data apply the redundant method for simultaneous data transfer. Network test results NS 2 cable simulation and Matlab animation can be used using Multipath TCP to increase verification greater than TCP based on the results of several RTT Delay trials, obtained optimal results on the 25 ms RTT Delay according to the Multipath TCP throughput of 132,676 Kbps and TCP 40.01 Kbps on data transfer. While the lowest packet loss on TCP Multipat Delay 15 RT RT Ms is 5.60% and the highest on the 15 Ms RTT Delay on the TCP network is 54%.

Keywords: Multipath TCP, TCP, Wired, Throughput, Delay RTT, Packet Loss, NS 2, Matlab.

1. Pendahuluan

Perkembangan teknologi dan informasi di Indonesia sudah sangat pesat. Hampir semua device di Indonesia saat ini sudah terkoneksi menggunakan internet, antarmuka paling umum biasanya digunakan pada pada jaringan *Wired* untuk setiap transaksi maupun transfer data antar *device.* Pada jaringan *Wired*, transmisi dilakukan menggunakan *Singlepath TCP*. Protokol *TCP* yang berjalan pada *layer transport* yang umumnya di gunakan untuk pengiriman paket data, dalam hal ini, *TCP* masih memiliki kekurangan dalam hal *throughput* yang masih rendah yang menyebabkan *traffic* mengalami *Congestion* [1].

Protokol yang berada di *layer transport* seperti *TCP* hanya melintasi *singlepath* dengan menggunakan satu antarmuka. *TCP* masih memiliki kekurangan lain yaitu *Congestion* di *singlepath TCP* yang di sebabkan antrian paket yang berlebihan akan menyebabkan *Packet Loss.* Saat *Packet Loss* terjadi, *TCP sender* akan berhenti mengirim data hingga paket yang hilang berhasil di transmisikan ulang dan *throughput* menjadi tidak optimal karena *Packet Loss* yang berlebihan [2].

Packet Loss merupakan jumlah paket yang hilang pada saat transmisi. *Packet Loss* disebabkan karena *corruption* pada media transmisi, bisa juga disebabkan karena paket di *drop* karena jaringan yang padat

(*congestion)* dan *buffer* kekurangan *space* untuk menampung. Kerugian yang di hasilkan dari *Packet Loss* dapat menyebabkan *Throughput* yang di hasilkan tidak optimal [2]. Pada jaringan Wired, TCP memiliki masalah ketika terjadi *Congestion* dan disebabkan antrian paket berlebihan yang akan menyebabkan *Packet Loss.*

Multipath TCP adalah solusi untuk masalah diatas. *Multipath TCP* memungkinkan koneksi TCP untuk beroperasi secara bersamaan dengan beberapa antarmuka yang ada untuk meningkatkan pemanfaatan sumber daya dan ketangguhan koneksi [3]. *Multipath TCP* meningkatkan *bandwidth* dan menciptakan *subflows* secara dinamis dan di hapus saat berjalan dengan opsi *TCP, subflows* membentuk koneksi *Multipath TCP* ketika ada *handshake* dan Multipath TCP mampu di kedua sisi TCP dengan perukaran paket SYN, maka akan ada tambahan *Subflows* dari antarmuka jaringan yang tersedia, *subflows* menunjukkan bahwa *Multipath TCP* dapat mencapai tiga kali *throughput* dari *TCP. TCP* data dibagi ke *TCP subflow* di *Multipath TCP* untuk mentransfer dalam jaringan pada saat yang sama [2] [6].

Untuk mengatasi masalah diatas, sudah ada penelitian yang membahas tentang *Packet Loss* pada *Multipath* TCP [2]. Dari penelitian [2], merancang *Congestion scenario* yaitu membandingkan performa *Multipath TCP* dan *Singlepath TCP* agar meminimalisir *Packet Loss* agar meningkatkan *Throughput.* Namun penelitian tersebut masih dilakukan dengan jaringan biasa.

Pada Tugas Akhir ini, menggunakan sebuahe metode *redundant* untuk mengetahui perbandingan performa *Multipath TCP* dan *TCP* pada jaringan *Wired*. Dengan memaksimalkan antarmuka jaringan yang ada sehingga mengetahui *Througput* dan *Packet Loss* yang paling optimal.

Beberapa batasan yang terdapat pada tugas akhir ini adalah :

- Penelitian bersifat simulasi.
- Pengukuran hasil hanya pada *Throughput* dan *Packet Loss*
- Antar muka yang digunakan pada jaringan *Wired*.
- *Multipath TCP* maksimal menggunakan 2 server pada simulasi *Network Simulator* 2
- Jarak pada simulasi di abaikan.

Fokus tujuan dari pada tugas akhir ini adalah merancang sebuah system yang mampu menunjukan performa *Multipath TCP* terhadap *TCP* pada jaringan *Wired* kondisi pembebanan jalur yang seimbang dan menunjukkan apakah *Multipah TCP* layak di gunakan dari pada *TCP* dengan tolak ukur *throughput* dan *packet loss.*

Penulisan bab pertama membahas tentang latar belakang, rumusan masalah, serta tujuan dari tugas akhir ini. Selanjutnya pada bab kedua dilanjutkan dengan membahas mengenai studi terkait yang berisi pengamatan terhadap paper-paper penelitian terdahulu baik *paper* yang diterbitkan maupun lewat buku dengan topik yang terkait dengan permasalahan yang ingin diangkat kedalam tugas akhir ini. Pada bab 3 menunjukkan sistem yang di ajukan lalu pada bagian bab 4 akan di diskusikan mengenai hasil pengujian dan evaluasi sistem. Terakhir pada bab 5 membahas kesimpulan dari tugas akhir ini juga saran yang diberikan terkait tugas akhir.

2. Studi Terkait

Delay adalah waktu tunda suatu proses dan kondisi pengiriman paket atau data dari satu titik sumber ke titik tujuan yang melebihi jangka waktu normal. Kondisi *Delay* dapat diperoleh dari paket data diterima dikurangi dengan waktu paket data dikirimkan yang akan menghasilkan angka yang disebut waktu *Delay. Delay RTT* pada tugas akhir ini digunakan untuk mengukur waktu yang di butuhkan oleh pengirim ke penerima suatu data dan menyeimbangkan *path* dengan *troughput* sebesar – besarnya [8].

Pada penelitian [6] Menggunakan *network simulator* versi 2 (NS-2), efek dari parameter jaringan *Wired* dan kondisi pengirim dan penerima terhadap *Multipath TCP* dan *TCP* disimulasikan. Kemudian *Delay RTT, Throughput, Packet Loss* pada penerima *Wired* digunakan untuk membandingkan kinerja kedua model jaringan. Pada penelitian yang pernah ada yaitu eksperimen internet untuk evaluasi kinerja *Multipath TCP* pada jaringan *Wired* di fokuskan pada pemanfaatan *bandwith* dan *delay transfer file*.

Packet Loss merupakan jumlah paket yang hilang pada saat transmisi. *Packet Loss* disebabkan karena *corruption* pada media transmisi, bisa juga disebabkan karena paket di *drop* karena jaringan yang padat (*congestion)* dan *buffer* kekurangan *space* untuk menampung. Tingkat *Congestion* dari setiap jalur secara tidak langsung dianggap mengakibatkan hilangnya paket yang [11]. Multipath TCP dengan *coding* jaringan untuk menggunakan beberapa jaringan secara paralel untuk meningkatkan efisiensi, kualitas layanan dan mengatasi *Packet Loss* berlebih [6].

Multipath TCP didasarkan pada *TCP* dan *Multipath TCP* mentansmisikan data melalui beberapa jalur yaitu dengan metode *default*. Telah diketahui bahwa *Multipath TCP* menunjukan *Throughput* lebih baik daripada *singlepath TCP*. Dalam *paper* ini, penulis menyelidiki pengaruh ukuran *buffer router* pada *troughput Multipath TCP*. Penulis mengamati bahwa *Multipath TCP*, yang menggunakan algoritma *Lowest-RTT-First* dapat menunjukkan *throughput* yang lebih buruk dari pada *singlepath TCP* ketika *buffer router* pada jalur transmisi *RTT* terendah kecil. Penulis menyelidiki melalui simulasi dan memiliki dua skenario untuk mengetahui

throughput yang di hasilkan. Skenario pertama berdasarkan *delay* yang sudah di tentukan pada *Multipath TCP* dan hasil menunjukkan bahwa *throughput* akan besar ketika memiliki *delay* rendah di kedua jalur dibandingkan dengan *single path TCP*. Sedangkan skenario kedua yaitu berdasarkan ukuran *buffer* yang sudah di tentukan pada *Multipath TCP* hasil menunjukkan jika ukuran *buffer* sudah ditentukan, *singlepath TCP* lebih baik ketika *buffer* pada *Multipath TCP* rendah dan *RTT* rendah [9].

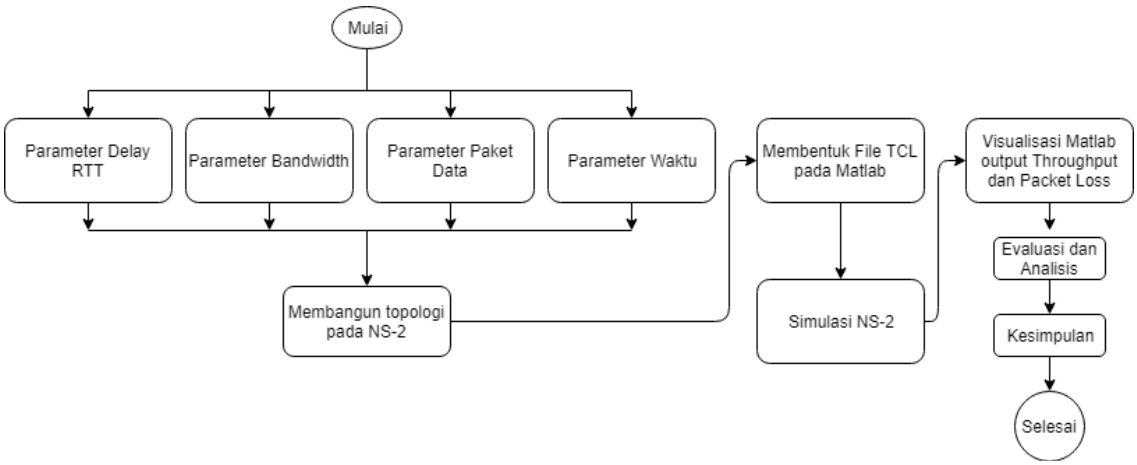
Pada penelitian [10], membahas tentang *Multipath TCP* memungkinkan koneksi *TCP* untuk menggunakan beberapa jalur untuk pengiriman data. Hal ini diharapkan dapat meningkatkan kemampuan *multihoming* dari *nodes* yang saling berkomunikasi. Pada *paper* ini memperkenalan studi eksperimental koneksi *Multipath TCP* dari node *Wireless multihomed. Tranceivers nodes* menggunakan dua antarmuka *Wireless.* Satu terhubung ke jaringan *WiFi*, sedangkan yang lain mengemulasi *link* khusus seperti 3G atau 4G. Kinerja *Multipath TCP* diselidiki dalam pengaturan praktis, dimana jaringan *WiFi* mendapat gangguan jaringan *WiFi* terdekat lainnya. Efek dari parameter koneksi seperti ukuran segmen maksimum dan ukuran buffer penerima yang di periksa. Untuk koneksi *multihomed* seperti itu, penulis menunjukkan bahwa *Coupled Congestion Control Algorithm* digabungkan dengan *Multipath TCP* dapat mentransfer lebih banyak *traffic* melalui *link Wifi* daripada *link* khusus hanya jika laju saluran *WiFi* jauh melebihi kecepatan *link* khusus.

3. Sistem yang Dibangun

Dalam perancangan *Multipath TCP* pada jaringan *Wired* pada dasarnya adalah protokol *Multipath TCP* merupakan pengembangan dari protokol *TCP* yang bekerja pada *transport layer* dengan menggunakan beberapa antarmuka jaringan secara bersamaan dan juga untuk menghitung nilai *throughput* di kerjakan dengan simulasi dan beberapa kombinasi *Delay RTT*, dengan begitu *Multipath TCP* dan *TCP* mendapatkan nilai *throughput* tertinggi pada keadaan *transfer data*, secara *TCP* di kedua *path* dalam keadaan *balance*, sehingga tidak ada jalur yang dirugikan. *Multipath TCP* memiliki tiga metode dengan cara dan kebutuhan yang berbeda yakni :

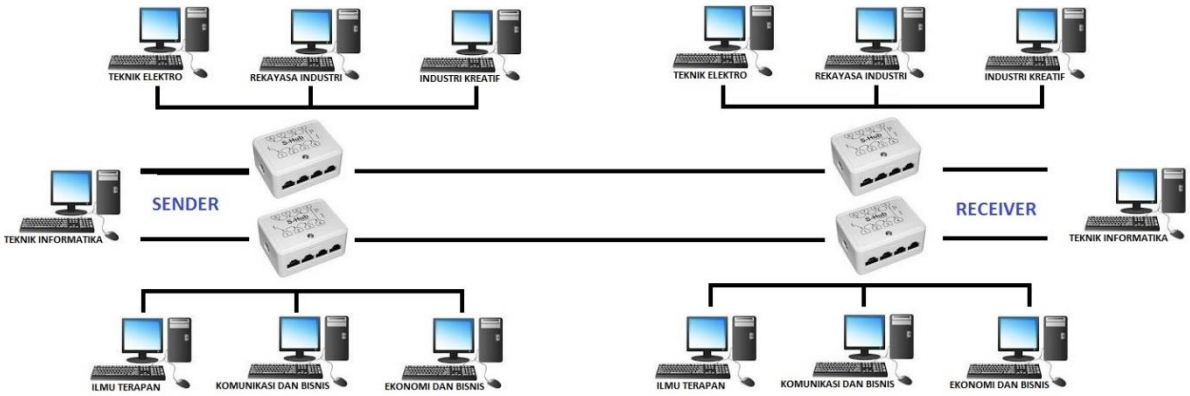
- Metode *default* : dua antarmuka aktif tetapi satu antarmuka sebagai cadangan ketika antarmuka *default* terjadi *loss* atau *congestion.* Mengirim data pada *subflows* dengan *RTT* terendah hingga *congestion-window full*, metode akan mulai metransmisikan pada subflows dengan *RTT* berikutnya yang lebih tinggi. Metode default digunakan untuk penelitian seperti konsumsi energi dari *Multipath TCP*.
- Metode *round robin* : menggunakan dua antarmuka aktif dengan pengiriman secara bergantian pada setiap paket. Metode *round robin* memiliki kinerja yang kurang baik termasuk transfer data dan perpindahan jalur. Kinerja metode ini biasanya hanya digunakan untuk tujuan akademik atau pengujian.
- Metode *redundant* : menggunakan dua antarmuka aktif dengan pengiriman sekaligus secara bersamaan sehingga memaksimalkan antarmuka yang ada. Mencoba metransmisikan *traffic* yang sibuk pada semua antarmuka yang tersedia.

Dari ketiga metode diatas, dalam tugas akhir ini menggunakan metode *redundant* agar memaksimalkan jalur dan antarmuka yang aktif dengan pengiriman secara sekaligus dan bersamaan dirasa cocok untuk membuat simulasi *Multipath TCP* dan *TCP* pada jaringan *Wired* termasuk mengatasi paket *drop*. Gambaran umum system terdapat pada Gambar 1.



Gambar 1. Gambaran Umum Sistem dan Alurnya

Pada tugas akhir ini bersifat simulasi menggunakan Network Simulator 2 (NS-2) dan visualisasi menggunakan Matlab 2010b [4]. Dengan menggunakan beberapa parameter yaitu *Delay RTT, Bandwith*, Paket Data dan *Waktu*. Kemudian menggunakan metode *redundant* dengan melibatkan semua jalur yang ada pada NS-2. Pertama-tama membuat file tcl dari Matlab pada jaringan *wired* dan menentukan *delay RTT* yang ingin di simulasikan, lalu file tcl dari Matlab di Simulasikan pada NS-2. Hasil Simulasi NS-2 yang menghasilkan file trace di animasikan pada Matlab yaitu membandingkan performa *Multipath TCP* dan *TCP* pada jaringan *Wired* apakah *Multipath TCP* lebih baik dan layak digunakan dibandingkan *TCP*. Setelah di analisis, Evaluasi dilakukan dari hasil perbandingan pada tahap sebelumnya.

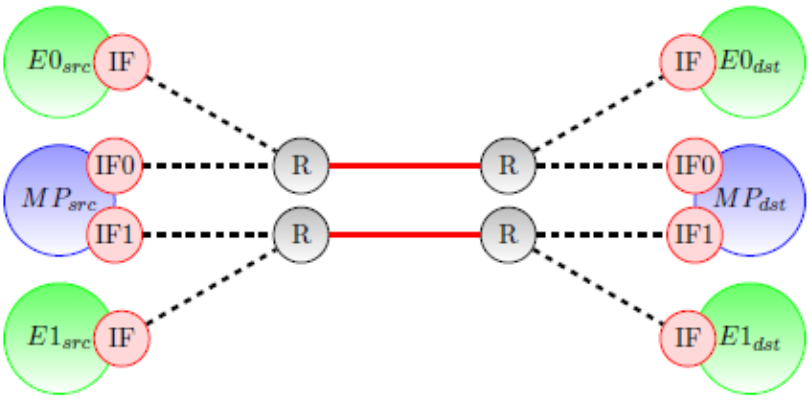


Gambar 2. Topologi Wired

Pada topologi di atas adalah topologi Universitas Telkom (Gambar 2), *Multipath TCP* dan *TCP* menggunakan antar muka *Wired* yang menghubungkan antar terjadi transfer data Server dan Client antara gedung fakultas pada Universitas Telkom. Pada simulasi ini, jenis transport agent yang dipakai adalah *Multipath TCP* dan *TCP*. Simulasi koneksi *TCP* pada simulasi ini menggunakan dua agent yang berpasangan yaitu *TCP sender* dan *TCP sink*. Untuk *Multipath TCP* atau *TCP sender*, pada simulasi ini menggunakan *Multipath TCP* atau *TCP sender base*. *TCP sink* bertugas mengirim *ACK per packet* yang diterima pada *TCP sender* pasangannya.

Skenario yang di bentuk pada tugas akhir ini yaitu : mengetahui hasil *throughput* dan *Packet Loss* dengan ukuran *Delay RTT* pada antarmuka *Wired*. Dimulai dengan menentukan parameter pengujian yang ingin dilakukan kemudian mengatur ukuran *Delay RTT* sesuai parameter yang ditentukan pad jaringan *Wired* yaitu 10 Ms, 15 Ms, 25 Ms, 50 Ms, 100 Ms. *File* TCL dibentuk melalui Matlab lalu *file* tcl di simulasikan pada NS-2 memakan waktu 5 menit lalu menghasilkan *file trace* yang akan di jalankan kembali melalui visualisasi Matlab yang memakan waktu rata-rata 6–8 jam. Kemudian hasil dapat dilihat *throughput* dan *Packet Loss*.

3.1 *Simulasi NS-2*



Gambar 3. Topologi saat Simulasi

Pada (Gambar 3) menunjukkan topologi yang di gunakan saat simulasi. Ini adalah komunikasi satu arah dari MP_{src} ke MP_{dst}, dari E0src ke $E0_{dst}$ dan dari $E1_{src}$ ke $E1_{dst}$. Konfigurasi ini adalah kasus Multipath dan sangat mungkin digunakan dalam penggunaan *Multipath TCP*.

3.1.1 Konfigurasi topologi

Setelah menentukan topologi untuk simulasi, yaitu melakukan konfigurasi pada setiap fakultas yang di tentukan Topologi dalam model simulasi dengan menggunakan NS2 dibangun oleh *Node* dan *link*. :

```
#
# mptcp sender FAKULTAS TEKNIK INFORMATIKA
#
set n0 [$ns node]
set n0_0 [$ns node]
set n0_1 [$ns node]
$n0 color red
$n0_0 color red
$n0_1 color red
$ns multihome-add-interface $n0 $n0_0
$ns multihome-add-interface $n0 $n0_1

#
# mptcp receiver FAKULTAS TEKNIK INFORMATIKA
#
set n1 [$ns node]
set n1_0 [$ns node]
set n1_1 [$ns node]
$n1 color blue
$n1_0 color blue
$n1_1 color blue
$ns multihome-add-interface $n1 $n1_0
$ns multihome-add-interface $n1 $n1_1

#
# router tcp 1
#
set n2 [$ns node]
set n3 [$ns node]
$n2 color yellow
$n3 color yellow

#
# router tcp 2
#
set n4 [$ns node]
set n5 [$ns node]
$n4 color green
$n5 color green
#
# FAKULTAS TEKNIK ELEKTRO
# sender
set n6 [$ns node]
# receiver
set n7 [$ns node]
$n6 color cyan
$n7 color cyan

# FAKULTAS REKAYASA INDUSTRI
# sender
set n8 [$ns node]
```

```
# receiver
set n9 [$ns node]
$n8 color cyan
$n9 color cyan

# FAKULTAS INDUSTRI KREATIF
# sender
set n10 [$ns node]
# receiver
set n11 [$ns node]
$n10 color cyan
$n11 color cyan

# FAKULTAS EKONOMI DAN BISNIS
# sender
set n12 [$ns node]
# receiver
set n13 [$ns node]
$n12 color cyan
$n13 color cyan

# FAKULTAS ILMU TERAPAN
# sender
set n14 [$ns node]
# receiver
set n15 [$ns node]
$n14 color cyan
$n15 color cyan

# FAKULTAS KOMUNIKASI DAN BISNIS
# sender
set n16 [$ns node]
# receiver
set n17 [$ns node]
$n16 color cyan
$n17 color cyan
```

Pada simulasi ini, *Node* yang dibangun 2 *Node* yang menggunakan mode MPTCP dan 2 node yang memakai mode TCP. Tiap *Node* didefinisikan dengan menggunakan *command* $ns *Node*. Artinya, tiap *Node* di*set* menjadi sebuah objek *Node*. Kemudian tiap *Node* diberi warna untuk membedakan node 1 dengan yang lainnya sesuai dengan system transfer data yang sesuai dengan gambar topologi jaringan skenario simulasi.

3.1.2 Pembangunan link dan Bandwidth

```
$ns duplex-link $n2 $r1 10Mb 5ms DropTail
$ns duplex-link $r3 $n3 10Mb 5ms DropTail
$ns duplex-link $n4 $r2 10Mb 5ms DropTail
$ns duplex-link $r4 $n5 10Mb 5ms DropTail

$ns duplex-link $n0_0 $r1 10Mb 5ms DropTail
$ns duplex-link $r1 $r3   1Mb 5ms DropTail
$ns queue-limit $r1 $r3 30
$ns duplex-link $n1_0 $r3 10Mb 5ms DropTail

$ns duplex-link $n0_1 $r2 10Mb 5ms DropTail
$ns duplex-link $r2 $r4   1Mb 5ms DropTail
$ns queue-limit $r2 $r4 30
$ns duplex-link $n1_1 $r4 10Mb 5ms DropTail
```

Link dibuat antara dua buah *Node* dan digunakan untuk menghubungkan kedua *Node* tersebut. Pada simulasi ini, jenis *link* yang digunakan adalah duplex *link*. Hal ini menandakan bahwa antara kedua *Node* terdapat *link* dua arah dan terjadi komunikasi data dua arah antar kedua *Node*. Tiap *link* memiliki parameter *bandwidth* sebesar 10Mb, *delay* sebesar 5 ms, dan jenis antrian yang digunakan adalah DropTail.

3.1.3 Pembuatan Transport Agen dan Pasangannya

Pada jaringan dikenal istilah *layer* istilah *layer* komunikasi. Terdapat empat *layer* komunikasi yaitu: *layer* aplikasi, transport, IP, dan network access. Lapisan transport merupakan *layer* komunikasi yang mengatur komunikasi data yang akan digunakan oleh lapisan aplikasi di atasnya. Pada simulasi menggunakan NS2 ini, lapisan transport disimulasikan dengan objek simulasi yang bernama *agent*.

```
#
# buat pengirim mptcp
#
set tcp0 [new Agent/TCP/FullTcp/Sack/Multipath]
$tcp0 set window_ 100
$ns attach-agent $n0_0 $tcp0
set tcp1 [new Agent/TCP/FullTcp/Sack/Multipath]
$tcp1 set window_ 100
$ns attach-agent $n0_1 $tcp1
setmptcp [new Agent/MPTCP]
$mptcpattach-tcp $tcp0
$mptcp attach-tcp $tcp1
$ns multihome-attach-agent $n0 $mptcp
set ftp [new Application/FTP]
$ftp attach-agent $mptcp
#
# buat penerima mptcp
#
setmptcpsink [new Agent/MPTCP]
set sink0 [new Agent/TCP/FullTcp/Sack/Multipath]
$ns attach-agent $n1_0 $sink0
set sink1 [new Agent/TCP/FullTcp/Sack/Multipath]
$ns attach-agent $n1_1 $sink1
$mptcpsink attach-tcp $sink0
$mptcpsink attach-tcp $sink1
$ns multihome-attach-agent $n1 $mptcpsink
$ns multihome-connect $mptcp $mptcpsink
$mptcpsink listen
#
# buat koneksi TCP 1
#
set reno0 [new Agent/TCP/FullTcp/Sack]
$reno0 set window_ 100
$ns attach-agent $n2 $reno0
set renosink0 [new Agent/TCP/FullTcp/Sack]
$ns attach-agent $n3 $renosink0
$ns connect $reno0 $renosink0
$renosink0 listen
setftp0 [new Application/FTP]
$ftp0 attach-agent $reno0
#
# buat koneksi TCP 2
#
set reno1 [new Agent/TCP/FullTcp/Sack]
$reno1 set window_ 100
$ns attach-agent $n4 $reno1
set renosink1 [new Agent/TCP/FullTcp/Sack]
$ns attach-agent $n5 $renosink1
$ns connect $reno1 $renosink1
$renosink1 listen
```

```
set ftp1 [new Application/FTP]
$ftp1 attach-agent $reno1
```

Pada simulasi ini, jenis *transport agent* yang dipakai adalah Multipath *Transport Control Protocol* (MPTCP) dan *Transport Control Protocol* (TCP).. Simulasi koneksi TCP pada simulasi ini menggunakan dua *agent* yang berpasangan, yaitu TCP Sender dan TCP Sink. Untuk MPTCP / TCP sender, pada simulasi ini menggunakan MPTCP / TCP sender base. Sedangkan untuk MPTCP / TCP Sink menggunakan Base TCP Sink. TCP Sink bertugas mengirimkan ACK per *packet* yang diterima pada TCP Sender pasangannya.

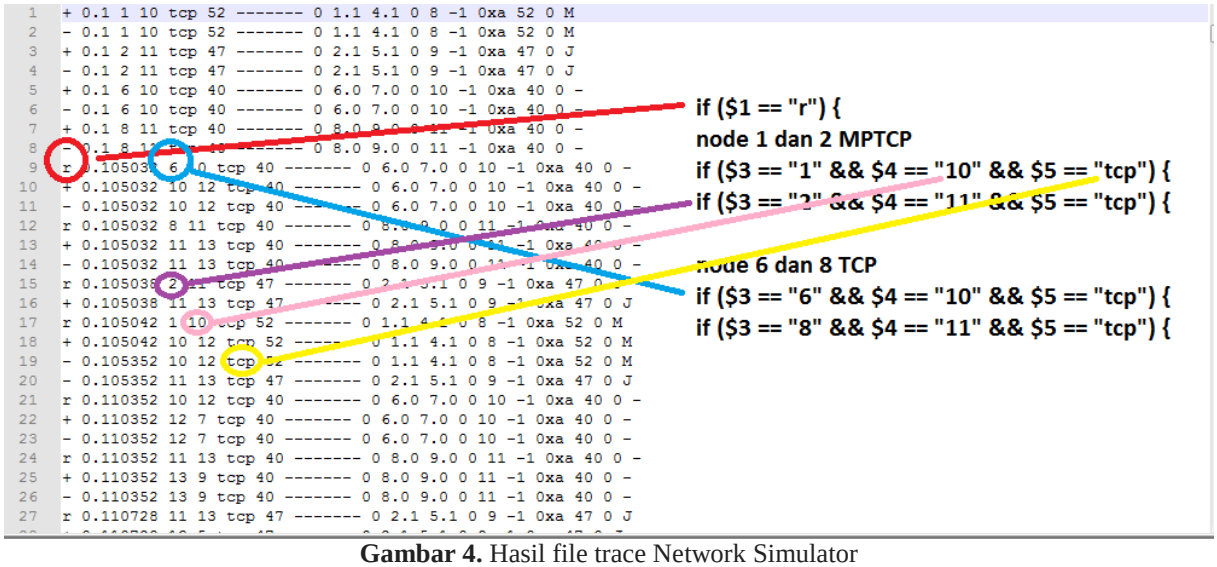
3.1.4 Pembuatan Layer Aplikasi

Langkah peratma membuat procedure finish yaitu bila simulasi berakhir akan mengirim global variable yang akan digunakan program serta menutup semua file trace simulasi. Langlah berikutnya adalah mendifinisikan pekerjaan pada step waktu simulasi yaitu menjalankan semua FTP pada waktu 0.1 detik simulasi dan menghentikan simulasi pada waktu 100 detik.

```
proc finish {} {
        global ns f
        globalnf
        $ns flush-trace
        close $f
        close $nf
        exit
}
$ns at 0.1 "$ftp start"
$ns at 0.1 "$ftp0 start"
$ns at 0.1 "$ftp1 start"
$ns at 100 "finish"
$ns run
```

3.1.4 Hasil File Trace Simulasi

Untuk menghitung nilai throughput adalah dengan menggunakan script awk, langkah-langkah pada script Matlab dengan mempertimbangkan hasil file trace simulasi berikut :



Gambar 4. Hasil file trace Network Simulator

Berdasarkan hail berikut maka jika header adalah huruf "r" maka data baris tersebut digunakan. Selanjunya mempertimbangkan kolom ke 3 yaitu 1 dan 2 untuk terminal yang menggunakan *Multipath TCP*. Sedangkan jika bernilai 6 atau 8 maka baris tersebut menunjukkan bahwa data tersebut untuk terminal yang menggunakan mode *TCP*. Data valid bilamana kolom ke 4 bernilai 10 atau 11 dan kolom ke 5 adalah "tcp". Script Matlab untuk tujuan tersebut diatas adalah sebagai berikut:

1. Membaca file data trace dari hasil simulasi NS- 2

```
TRdata = textread('wire_delay_RTT_10ms.tr','%s','delimiter','\n','whitespace','');
```

2. Memenggal data teks menjadi kolom-kolom

a. membaca 1 baris data

```
txt=char(TRdata(i));
```

Hasilnya

```
txt = + 0.1 1 10 tcp 52 ------- 0 1.1 4.1 0 8 -1 0xa 52 0 M
```

b. Memenggal teks menjadi data per kolom dengan script

```
idx=0;
nilai='';
for j=1:length(txt)
   if txt(j)==' '
      idx=idx+1;
      dat(idx)={nilai};
      nilai='';
   else
      nilai=[nilai txt(j)];
   end
end
idx=idx+1;
dat(idx)={nilai};
```

Hasilnya

```
dat =    '+'   '0.1'   '1'   '10'   'tcp'   '52'   '-------'   '0'   '1.1'   '4.1'   '0'   '8'   '-1'   '0xa'   '52'   '0'   'M'
```

3. Melihat apakah data pada kolom pertama adalah data yang diterima node

```
if char(dat(1))=='r'
```

4. Melihat apakah yang diterima adalah data tcp

```
if sum(char(dat(5))=='tcp')==3
```

5. Melihat apakah node yang menerima adalah node Mptcp (node 1 yang merupakan jalur ke server MPTCP)

```
if str2num(cell2mat(dat(3)))==1
```

Apakah path 1

```
if str2num(cell2mat(dat(4)))==10
```

Jika ya data diambil

- Jumlah bit ada di kolom ke 6

```
b1=str2num(cell2mat(dat(6)));
```

- Ditambahkan dengan jumlah bit mptcp yang diterima

```
mptcp=mptcp+b1;
```

- Mengambil waktu penerimaan yang merupakan data kolom ke 2

```
w2r=str2num(cell2mat(dat(2)));
```

- Menghitung kecepatan transfer data dengan menghitung jumlah bit yang diterima dibandingkan waktu penerimaan dan membagi dengan 1000 sebagai konversi ke bit ke Kbit

```
kbps2=(mptcp/(w2r-0.1))/(1000) ;
```

6. Cara yang sama untuk node 2

```
if str2num(cell2mat(dat(3)))==2
   if str2num(cell2mat(dat(4)))==11
      rev=1;
      b1=str2num(cell2mat(dat(6)));
      mptcp=mptcp+b1;
      w2r=str2num(cell2mat(dat(2)));
      kbps2=(mptcp/(w2r-0.1))/(1000) ;
```

7. Untuk tcp dibedakan antara node 6 dan 8

```
if str2num(cell2mat(dat(3)))==6
   if str2num(cell2mat(dat(4)))==10
      b2=str2num(cell2mat(dat(6)));
      tcp1=tcp1+b2;
      w3r=str2num(cell2mat(dat(2)));
      kbps=(tcp1/(w3r-0.1))/(1000) ;
```

```
Sedang untuk node 8 dipisah nilainya
        b3=str2num(cell2mat(dat(6)));
    tcp2=tcp2+b3;
    w4r=str2num(cell2mat(dat(2)));
    kbps=(tcp2/(w4r-0.1))/(1000) ;
```

3.2 *Throughput*

Throughput adalah kecepatan transfer data efektif, yang fi ukur dalam Kbps. *Throughput* merupakan jumlah total kedatangan paket yang sukses yang diamati selamat interval waktu tertentu dan juga di bagi oleh durasi interval waktu tersebut.

$$Throughput = \frac{\text{Paket data yang diterima}}{Waktu\ Pengiriman\ Data} \qquad (2)$$

Untuk membuktikan bahwa performa *Multipath TCP* adalah solusi untuk meningkatkan *throughput* pada *TCP* perlu adanya perhitungan dan data dari hasil simulasi yang mencakup paket data yang diterima dam waktu pengiriman data oleh *Multipath TCP* dan *TCP*. Dengan membuktikan teori bahwa *Multipath TCP* menggunakan beberapa jaringan secara paralel untuk meningkatkan efisiensi, kualitas layanan dan mengatasi drop nya troughput pada TCP [2].

Tabel 1. Paket Data yang Dikirim dan Waktu pengiriman Data

Paket Data Dikirim	Paket Data Diterima TCP	Paket Data Diterima Multipath TCP	Waktu Pengiriman Data TCP	Waktu Pengiriman Data Multipath TCP
Delay RTT 10 Ms = 222.414	146.103	207.601	76,311 s	14,813 s
Delay RTT 15 Ms = 226.384	141.652	213.515	84,732 s	12,689 s
Delay RTT 25 Ms = 215.193	151.012	200.541	64, 181 s	14,652 s
Delay RTT 50 Ms = 225.232	103.401	210.133	121,831 s	15,099 s
Delay RTT 100 Ms = 226.706	150.403	217.141	76,303 s	9,565 s

Tujuan dari adanya tabel diatas memudahkan perhitungan yang akan di akumulasikan menggunakan rumus *throughput*, dari perhitungan tersebut akan membuktikan bahwa performa *Multipath TCP* akan memaksimalkan *throughput* dari pada *Singlepath TCP*.

3.3 *Packet Loss*

Pembahasan kali ini yaitu *Packet Loss* yaitu suatu kondisi yang menunjukkan banyak paket data yang hilang. *Packet loss* terjadi disebabkan karena adanya Congestion atau antrian pengiriman paket data yang berlebih pada satu jalur. Solusi yang di tawarkan adalah *Multipath TCP* dengan pengembangan dari jalur *Singlepath TCP*. Jumlah paket yang dikirimkan di dapat dari file trace hasil simulasi Network Simulator 2. Untuk menghitung berapa paket data yang hilang, *Packet Loss* mempunyai rumus yaitu :

$$Packet\ Loss = \left(\frac{\text{Data yang di kirim} - \text{Paket data yang diterima}}{\text{Paket data yang dikirim}}\right) x\ 100\% \qquad (1)$$

Untuk membuktikan bahwa *Multipath TCP* adalah solusi untuk mengurangi *Packet Loss* pada *TCP* perlu adanya perhitungan dan data dari hasil simulasi yang mencakup paket data yang dikirim, dan paket data yang diterima oleh *Multipath TCP* dan *TCP*. Dengan membuktikan teori bahwa *Multipath TCP* menggunakan beberapa jaringan secara paralel untuk meningkatkan efisiensi, kualitas layanan dan mengatasi *Packet Loss* berlebih [6].

Tabel 1. Paket Data yang Dikirim dan Paket Data Diterima

Paket Data Dikirim	Paket Data Diterima TCP	Paket Data Diterima Multipath TCP
Delay RTT 10 Ms = 222.414	146.103	207.601
Delay RTT 15 Ms = 226.384	141.652	213.515
Delay RTT 25 Ms = 215.193	151.012	200.541
Delay RTT 50 Ms = 225.232	103.401	210.133
Delay RTT 100 Ms = 226.706	150.403	217.141

Tujuan dari adanya tabel diatas memudahkan perhitungan yang akan di akumulasikan menggunakan rumus *Packet Loss*, dari perhitungan tersebut akan membuktikan bahwa performa *Multipath TCP* akan mengurangi *Packet Loss* yang akan terjadi pada *Singlepath TCP*.

4. Evaluasi

Pada bagian ini berisi tiga buah sub-bagian, yang berisi Parameter Pengujian, Hasil Pengujian, dan Analisis Hasil Pengujian. Pengujian dan analisis yang dilakukan selaras dengan tujuan Tugas Akhir sebagaimana dinyatakan dalam bab pendahuluan.

4.1 Parameter Pengujian

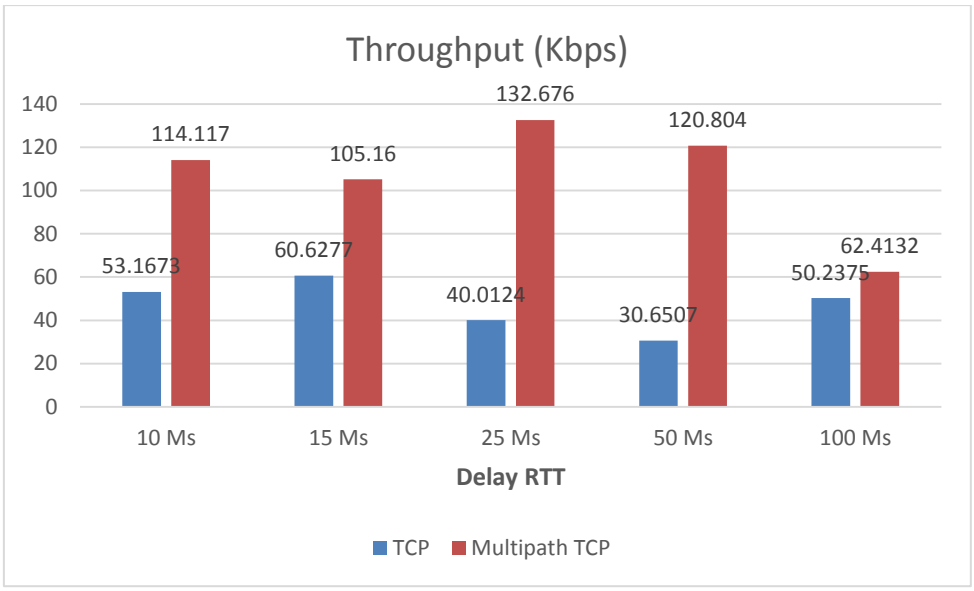
Tabel 2. Parameter Simulasi NS-2 dan Animasi Matlab di Multipath TCP dan TCP Pada Jaringan Wired.

Delay RTT	10 Ms	15 Ms	25 Ms	50 Ms	100 Ms
Bandwidth	10 Mbps	10 Mbps	10 Mbps	10 Mbps	10 Mbps
Paket Data	222.414	226.384	215.193	225.232	226.706
Waktu Animasi	6 jam	6 jam	7 jam	8 jam	8 jam

Tujuan pengujian yaitu parameter uji Delay RTT untuk menyeimbangkan path dengan throughput sebesar-besarnya. Sebagai pengukuran Parameter yang di tentukan akan di analasisi dari antarmuka yaitu *Multipath TCP* dan *TCP* pada jaringan *Wired* dengan berbagai *Delay RTT* (10 Ms, 15 Ms, 25 Ms, 50 Ms, 100 Ms). Bandwidth di sama ratakan saat agar simulasi berjalan pada tetapan yang sama. Paket data di dapat dari *file trace* hasil simulasi pada *NS-2* dan untuk menghitung *Packet Loss* yang terjadi pada *Multipath TCP* dan *TCP*. Waktu adalah pengukuran panjang proses animasi matlab dengan *Delay RTT, Bandwidth*, dan Paket Data yang digunakan. Untuk mendapatkan hasil sebagai analisa dari performansi antarmuka yang sudah di tentukan pada Tabel 1.

4.2 Hasil Pengujian

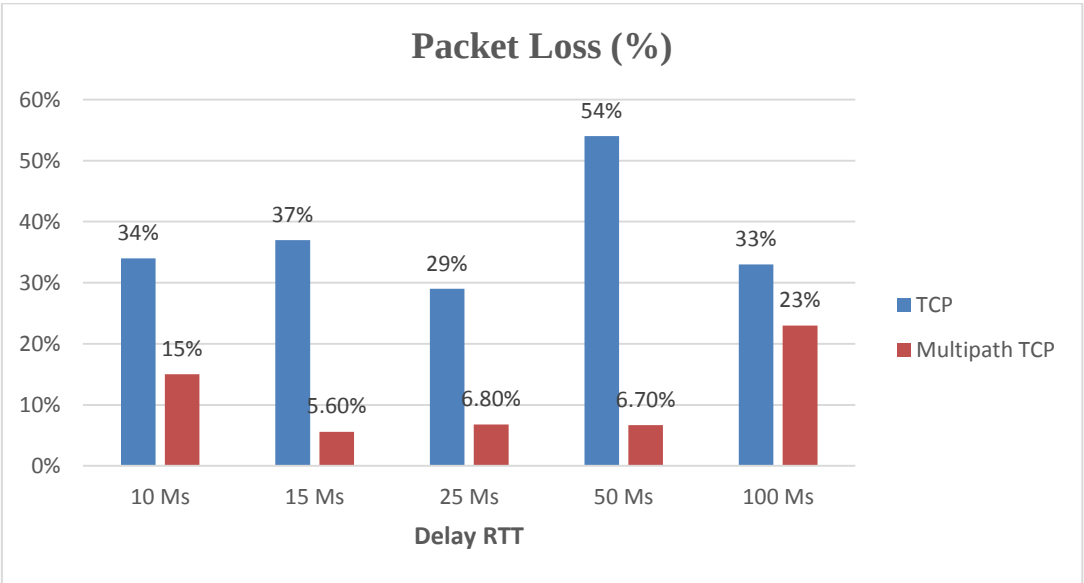
4.2.1 *Throughput*



Gambar 4. *Throughput Performance for Wired Network*

Pada (Gambar 4) hasil dari pengujian *throughput* di antarmuka yang berbeda dengan pengukuran beberapa *Delay RTT*, didapatkan hasil optimum pada *Delay RTT* 25 Ms, dengan *throughput* sebesar 144.788 Kbps dan kedua path *TCP* tetap pada posisi balance yaitu masing-masing sebesar 52.1136 Kbps. Perolehan *throughput Multipath TCP* lebih besar dari *TCP*. Jika dibandingkan dengan pengukuran beberapa *Delay RTT* lainnya hasil *throughput Multipath TCP masih di atas TCP*. Sangat sesuai dengan teori bahwa *Multipath TCP* memaksimalkan *throughput* dengan menggunakan antarmuka yang ada dan menggunakan metode redundant dengan memaksimalkan antarmuka aktif dengan pengiriman sekaligus secara bersamaan *througput* yang di hasilkan tinggi.

4.2.2 *Packet Loss*



Gambar 8. Hasil *Packet Loss Wired*

Pada (Gambar 8) hasil dari pengujian *Packet Loss* di antarmuka yang berbeda dengan pengukuran beberapa *Delay RTT*, didapatkan hasil terbaik pada *Delay RTT* 15 Ms, dengan *Packet Loss* terkecil sebesar 5.60% dan kedua path *TCP* mendapatkan posisi Packet loss terkecil pada Delay RTT 25 Ms yaitu sebesar 29%. Perolehan *Packet Loss Multipath TCP* lebih kecil dari *TCP*. Jika dibandingkan dengan pengukuran beberapa *Delay RTT*

lainnya hasil *Packet Loss* pada *Multipath TCP* lebih baik dari *TCP*. Sangat sesuai dengan teori bahwa *Multipath TCP* menggunakan beberapa jaringan secara paralel untuk meningkatkan efisiensi, kualitas layanan dan mengatasi *Packet Loss* berlebih [6].

4.3 Analisis Hasil Pengujian

Dari hasil pengujian diatas, metode *redundant* berpengaruh pada hasil *Throughput* dan *Packet Loss*. Memaksimalkan antarmuka dan jalur yang ada. Dari Masalah yang ada, metode *redundant* pada *Multipath TCP* sangat memperbaiki kinerja. Menerapkan *Delay RTT* dan *buffer* untuk membuktikan *Multipath TCP* lebih tinggi performansinya dibandingkan *TCP*.

Untuk Perbandingan *Multipath TCP* dan *TCP* pada antarmuka *Wired* menggunakan metode *redundant* yang memaksimalkan antarmuka yang ada dalam pengiriman data dan jalur yang ada. *Throughput* yang di hasilkan *Multipath TCP* lebih besar dibandingkan dengan *TCP* pada antarmuka. Lalu *Packet Loss* pada *Multipath TCP* cukup baik dibandingkan dengan *TCP* yang mengalami *Packet Loss* yang cukup besar.

5. Kesimpulan

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan sebelumnya, maka bisa ditarik kesimpulan yaitu: *Multipath TCP* dengan Metode redundant dengan penggunaan antarmuka *Wired* secara bersamaan mampu membuktikan bahwa pada jaringan *Wired* Performa *Multipath TCP* lebih baik dibandingkan dengan *TCP* dengan mengurangi Packet Loss yang terjadi akibat disebabkan karena paket di *drop* karena jaringan yang padat (*congestion)*.

Dalam tugas akhir ini, hasil pengujian dari tiga skenario dapat ditarik kesimpulan bahwa Performa *Multipath TCP* lebih baik dari *TCP* pada dua antarmuka (*Wired*), Pada jaringan *Wired Throughput* dan *Packet Loss* menunjukkan bahwa *Multipath TCP* dengan metode *redundant* lebih baik dari *TCP*.

Hasil *Throughput* yang di hasilkan *Multipath TCP* pada jaringan *Wired* paling optimal pada *Delay RTT* 25 Ms dengan hasil *throughput* yang tinggi dan *balance* yaitu 132.676 Kbps lalu pada *TCP* yaitu 40.01 Kbps. Pengujian terakhir yaitu pada *Packet Loss* pada *Multipath TCP* di jaringan *Wired* terendah sebesar pada *Delay RTT* 15 Ms yaitu 5.60% dan terbesar 23% pada *Delay RTT* 100 Ms, sedangkan *Packet Loss* pada *TCP* terendah pada *Delay RTT* 25 Ms yaitu 29% dan tertinggi pada *Delay RTT* 50 Ms yaitu 54%.

5.2 Saran

Perbandingan Performansi *Multipath TCP* dan *TCP* pada jaringan *Wired* dengan metode redundant memberikan pembuktian pada simulasi bahwa *Multipath TCP* lebih baik dan layak dari *TCP*. Di tugas akhir selanjutnya, peneliti akan melakukan penambahan parameter pengujian seperti *congestion control* dan juga menggunakan metode yang berbeda agar *Multipath TCP* semakin baik dan layak di gunakan di masa mendatang.

Daftar Pustaka

[1] MARTINEZ, Ivan; RAMOS, Victor. Choosing a TCP version over static ad hoc wireless networks: wired TCP or wireless TCP?. In: *2013 Seventh International Conference on Next Generation Mobile Apps, Services and Technologies*. IEEE, 2013. p. 170-174..

[2] HWANG, Jaehyun; WALID, Anwar; YOO, Joon. Fast coupled retransmission for multipath TCP in data center networks. *IEEE Systems Journal*, 2016, 12.1: 1056-1059.

[3] DONG, Pingping, et al. Performance enhancement of multipath TCP for wireless communications with multiple radio interfaces. *IEEE Transactions on Communications*, 2016, 64.8: 3456-3466.

[4] FROMMGEN, Alexander, et al. ReMP TCP: Low latency multipath TCP. In: *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016. p. 1-7.

[5] RAICIU, C.; HANDLEY, M.; WISCHIK, D. Coupled multipath-aware congestion control. *IETF draft draft-raiciu-mptcp-congestion-01. txt*, 2010.

[6] PHEJRSUKSAI, Khemmapath; PATTARAMALAI, Suwat. Performance comparison of multipath TCP data transferring in bottleneck and disjoint-path wired networks connected with Wi-Fi. In: *2017 International Electrical Engineering Congress (iEECON)*. IEEE, 2017. p. 1-4.

[7] ALIAS, Mohd Shafiq, et al. An experimental QoE performance evaluation of HTTP over Multipath TCP. In: *2016 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. IEEE, 2016. p. 320-325.

[8] Wang, J., Wen, J., Li, C., Xiong, Z. and Han, Y., 2015. DC-Vegas: a delay-based TCP congestion control algorithm for datacenter applications. *Journal of Network and Computer Applications*, *53*, pp.103-114.

[9] KIM, Hyungjik; CHOI, Sunwoong. The effect of routing path buffer size on throughput of multipath TCP. In: *2016 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2016. p. 1261-1263.

[10] *ABDRABOU, Atef; PRAKASH, Monika. Experimental performance study of multipath TCP over heterogeneous wireless networks. In: 2016 IEEE 41st Conference on Local Computer Networks (LCN). IEEE, 2016. p. 172-175.*

[11] OU, Shih-Hao, et al. Out-of-order transmission enabled congestion and scheduling control for multipath TCP. In: *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2016. p. 1069-1073.