

BAB 4

HASIL PENELITIAN

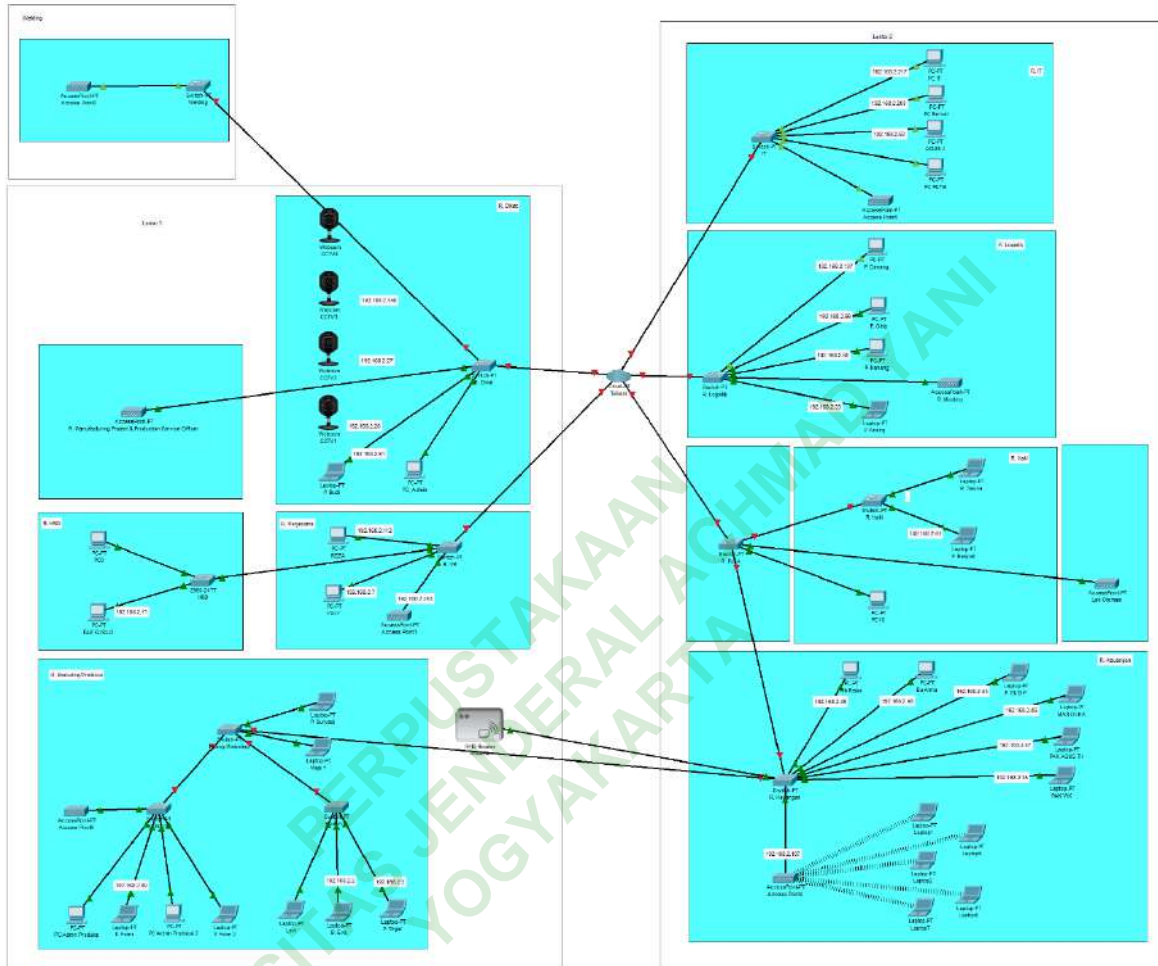
4.1 RINGKASAN HASIL PENELITIAN

Penelitian ini bertujuan untuk meningkatkan kualitas jaringan di Solo Technopark yang sering mengalami masalah *lagging* dan *disconnect* saat melakukan *streaming* YouTube dan *video conference* seperti rapat *online*. Untuk mengatasi masalah ini, diterapkan manajemen *bandwidth* menggunakan kombinasi *Simple queue* dan *Queue tree*. *Simple queue* digunakan untuk memberikan batasan *bandwidth* pada aplikasi yang tidak kritis, sementara *Queue tree* mengatur prioritas lalu lintas jaringan agar aplikasi penting, seperti *video conference* dan rapat *online*, mendapatkan prioritas lebih tinggi.

Setelah implementasi, kinerja jaringan menunjukkan perbaikan signifikan. Masalah *lagging* dan *disconnect* berkurang drastis, dan kualitas *video conference* meningkat dengan waktu respon yang lebih cepat dan stabil. Penggunaan kombinasi *Simple queue* dan *Queue tree* berhasil mengoptimalkan distribusi *bandwidth*, memastikan aplikasi penting dapat berjalan dengan lancar tanpa gangguan. Hasil ini menunjukkan bahwa pendekatan ini efektif dalam mengatasi masalah jaringan yang sebelumnya sering dialami.

4.2 DESAIN TOPOLOGI JARINGAN

Berikut adalah topologi jaringan gedung *RnD* Solo Technopark



Gambar 4.1 Topologi Jaringan Solo Technopark

4.3 PERSIAPAN PEMASANGAN APLIKASI

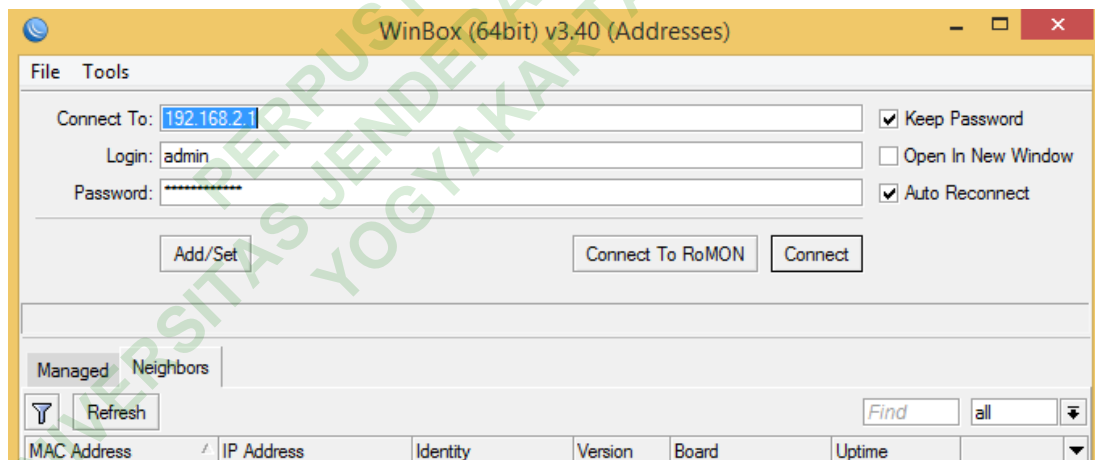
Sebelum melakukan konfigurasi pada mikrotik terlebih dahulu melakukan pemasangan sebuah aplikasi yang dapat mempermudah konfigurasi, dalam hal ini aplikasi Winbox diperlukan. Aplikasi Winbox bisa diunduh melalui situs resmi mikrotik. Setelah berhasil diunduh lalu jalankan aplikasi Winbox.

1. Anda dapat membuka aplikasi Winbox yang sudah diunduh tadi.



Gambar 4.2 Running Winbox

2. Untuk masuk, *IP address* atau *Mac Address* mikrotik dapat digunakan dengan *username* admin dan *password*, lalu dilanjutkan dengan menekan tombol *connect*, seperti terlihat pada gambar 4.3.



Gambar 4.3 Login aplikasi Winbox

4.4 KONFIGURASI ADDRESS LISTS PADA MIKROTIK

Daftar Alamat internet *protocol* yang digunakan untuk menandai seluruh alamat *IP* yang tersedia di mikrotik

1. Pada menu *IP*, kemudian diikuti dengan *Firewall* dan tab *address lists*. *Address lists* yang tersedia pada mikrotik akan muncul, seperti terlihat pada gambar 4.4 berikut.

Name	Address	Timeout	Creation Time
IP-Switch	192.168.2.0/24		Jul/11/2024 16:10:26

Gambar 4.4 Address lists

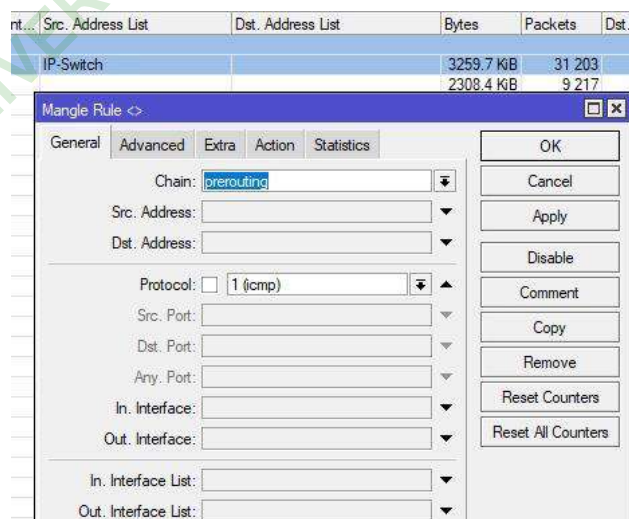
4.5 KONFIGURASI MANGLE PADA MIKROTIK

Protokol yang digunakan dapat membedakan antara *traffic upload* dan *download*. Hal ini dilakukan dengan fitur *mangle*, yang dapat digunakan untuk menandai paket dan koneksi berdasarkan *port*, *protocol*, *src* dan *dst address* serta parameter lain yang diperlukan. *Mangle* ini akan dapat membedakan antara *traffic upload* dan *download*. Berikut adalah langkah-langkah untuk menambah *mangle* pada setiap protokol dan koneksi pada mikrotik.

4.5.1 Konfigurasi Mangle ICMP (Internet Control Message Protocol)

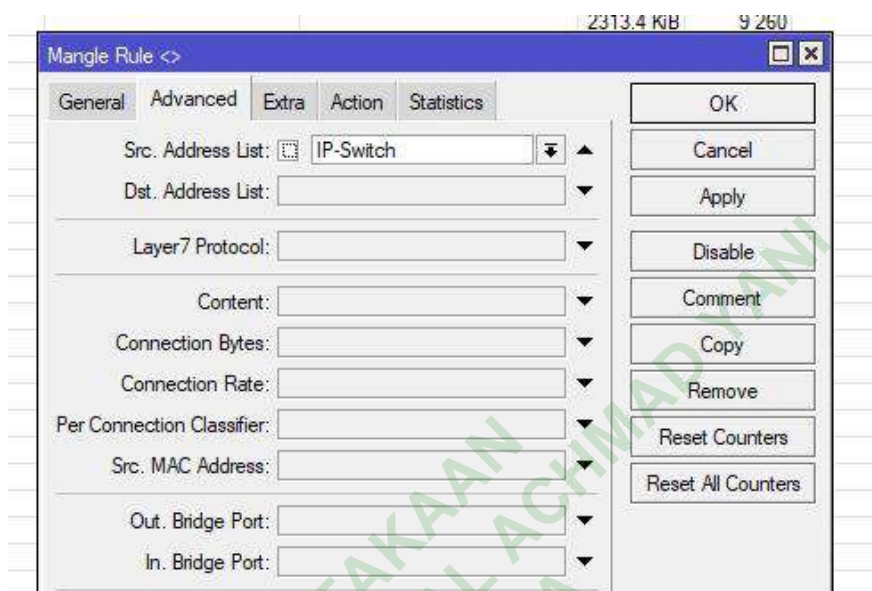
Berikut adalah langkah-langkah yang dilakukan untuk konfigurasi *mangle ICMP*

1. Pada menu *IP*, dilanjutkan dengan *Firewall* dan tab *mangle*. Tanda (+) untuk menambahkan *mangle*, seperti yang terlihat pada gambar 4.5. Selanjutnya, pada tab *general*, pada kolom *chain*, *prerouting*, dan pada kolom *protocol*, *ICMP* digunakan.



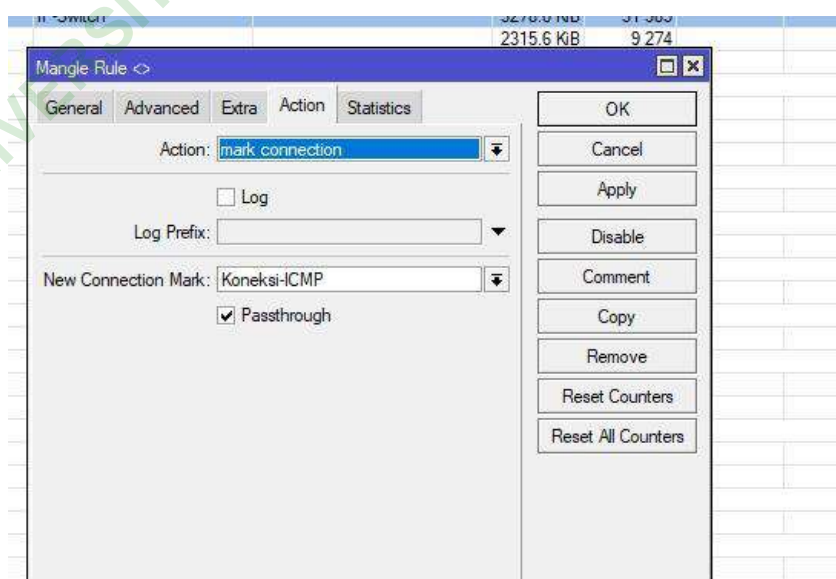
Gambar 4.5 Mark connections mangle protocol general- ICMP

2. Pada tab *advanced*, *src address lists* dan nama dari *address lists* yang telah dibuat sebelumnya (*IP-SWITCH*) tertera, seperti yang terlihat pada gambar 4.6 berikut.



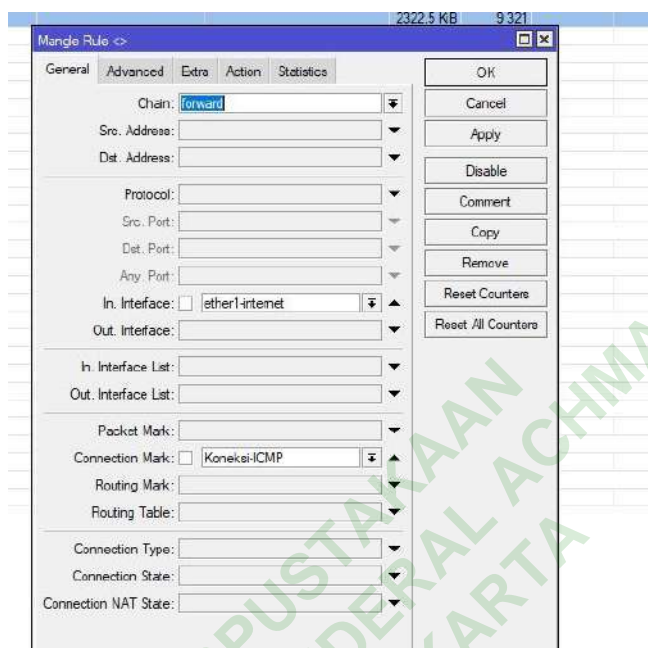
Gambar 4.6 Mark connections mangle protocol advanced-ICMP

3. Pada tab *action*, *Action-Mark connections* digunakan, dan *new connections Mark* menggunakan nama *Koneksi-ICMP*. Selanjutnya, pada *checklist* bagian *passthrough*, *apply* dan kemudian ok, seperti yang terlihat pada gambar 4.7 berikut.



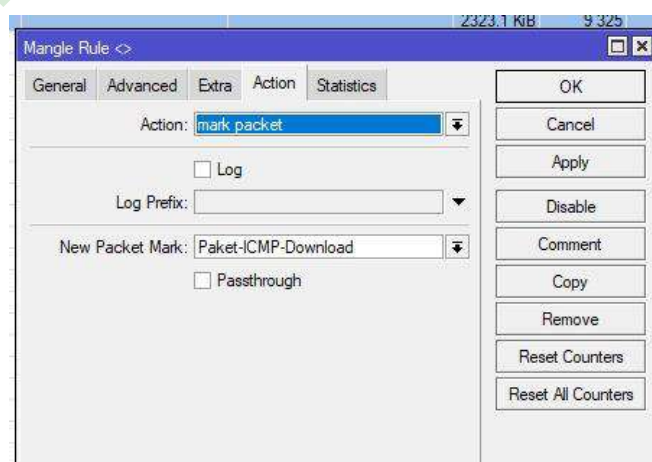
Gambar 4.7 Mark connections mangle protocol Action-ICMP

4. Pada *mangle*, tab *general* digunakan untuk memilih *chain forward*, *in.interface* diatur ke *ether1-internet* (sumber koneksi utama), dan pada bagian *Connections Mark*, Koneksi-ICMP yang telah dibuat sebelumnya ditetapkan, seperti yang terlihat pada gambar 4.8 berikut.



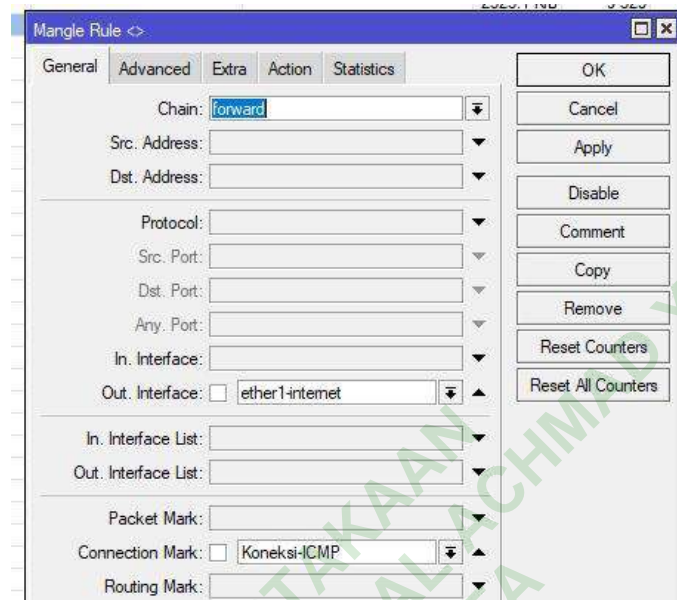
Gambar 4.8 Mark packet mangle general-ICMP download

5. Pada tab *action*, *Action-Mark packet* digunakan, dan *new packet Mark* diberi nama *Paket-ICMP-Download*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.9 berikut.



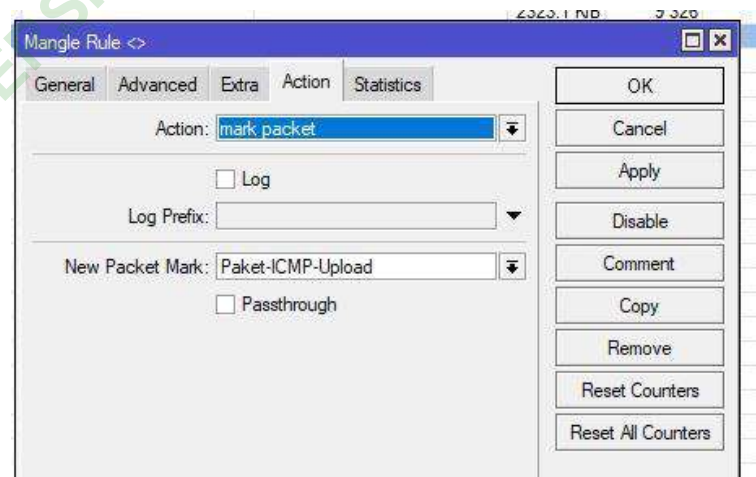
Gambar 4.9 Mark packet mangle Action-ICMP download

6. Pada tab *mangle-general*, *chain* adalah *forward*, *out interface* adalah *ether1-internet*, dan *Connections Mark* adalah *Koneksi-ICMP*, seperti yang terlihat pada gambar 4.10 berikut.



Gambar 4.10 Mark packet mangle General-ICMP upload

7. Pada tab *action*, *Action-mark-packet* digunakan, dan *New Packet Mark* bernama *Paket-ICMP-Upload*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.11 berikut.

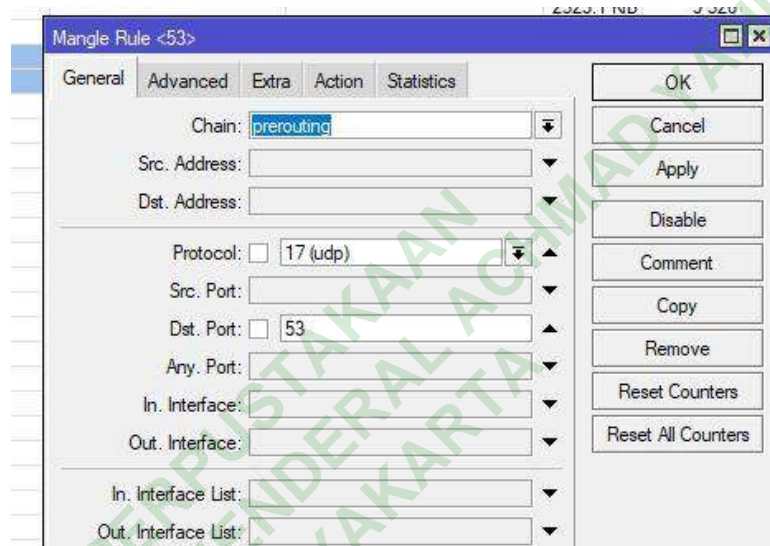


Gambar 4.11 Mark packet mangle Action-ICMP upload

4.5.2 Konfigurasi Mangle DNS (Domain Network Server)

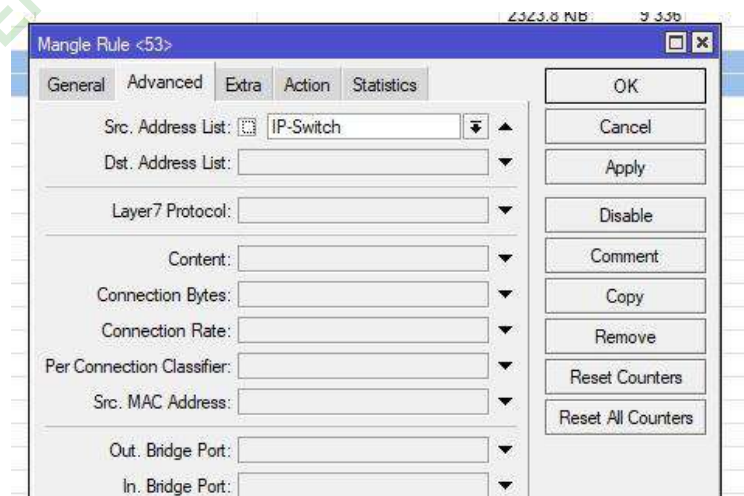
Berikut adalah langkah-langkah yang dilakukan untuk konfigurasi *mangle* DNS

1. Pada menu *IP, Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*, seperti yang terlihat pada gambar 4.12. Pada tab *general*, kolom *chain* menggunakan *prerouting* dan kolom *protocol* adalah *UDP*.



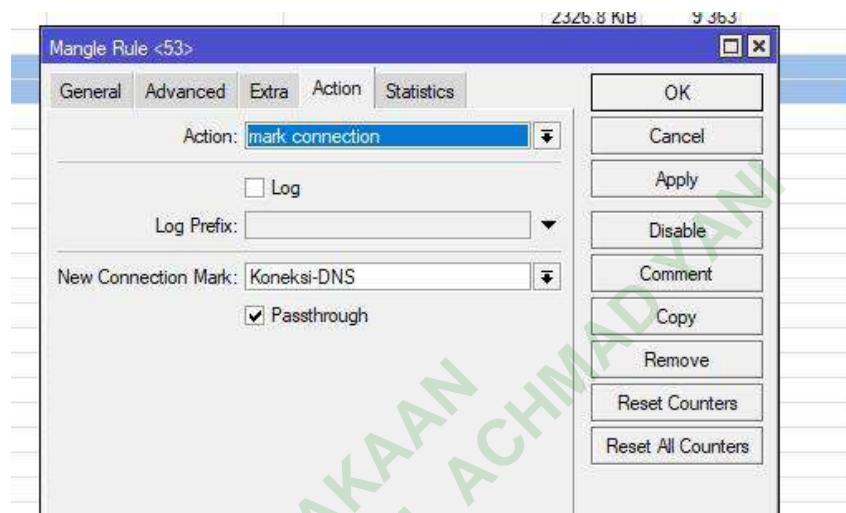
Gambar 4.12 Mark connections mangle protocol general- UDP

2. Pada tab *advanced* kolom *src.address list* menggunakan *IP-SWITCH*, seperti pada gambar 4.13 berikut.



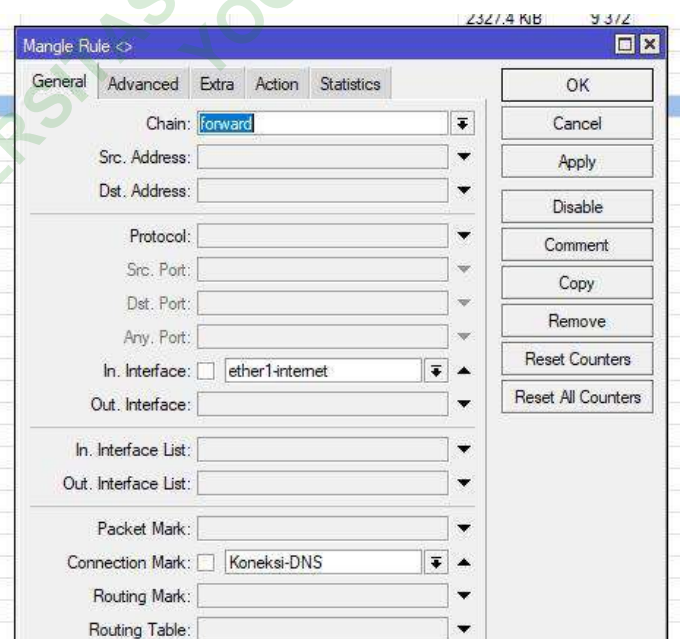
Gambar 4.13 Mark connections mangle protocol Advanced- UDP

3. Pada tab *action* menggunakan *Mark connections* dan *New Connection Mark* menggunakan nama *Koneksi-DNS*. *Checklist* pada bagian *passthrough* diaktifkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.14 berikut.



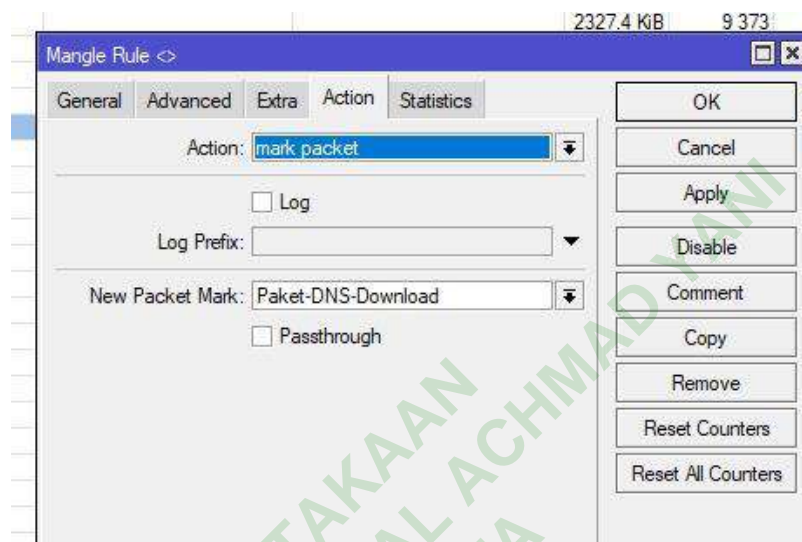
Gambar 4.14 *Mark connections mangle protocol Action- UDP*

4. Pada tab *mangle-general*, *chain* adalah *forward*, *in.interface* ke *ether1-internet*, dan *Connection Mark* menggunakan *Koneksi-DNS*, seperti yang terlihat pada gambar 4.15 berikut.



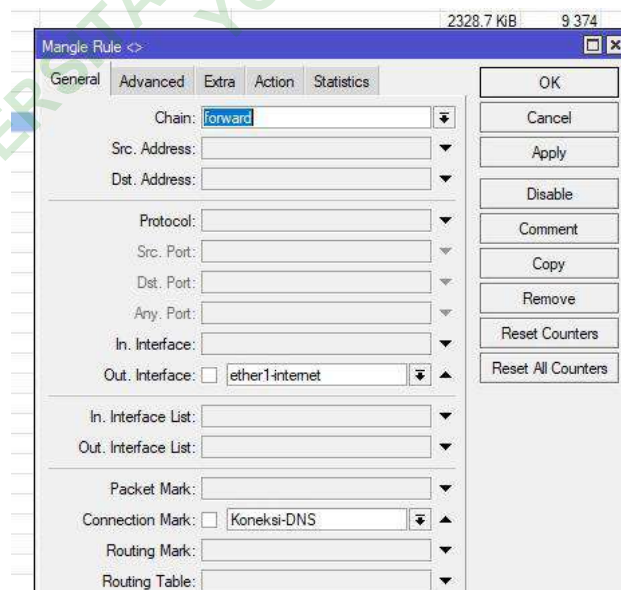
Gambar 4.15 *Mark packet mangle general-DNS download*

5. Pada tab *action*, kolom *action* menggunakan *Mark packet*, dan *New Packet Mark* menggunakan nama *Paket-DNS-Download*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.16 berikut.



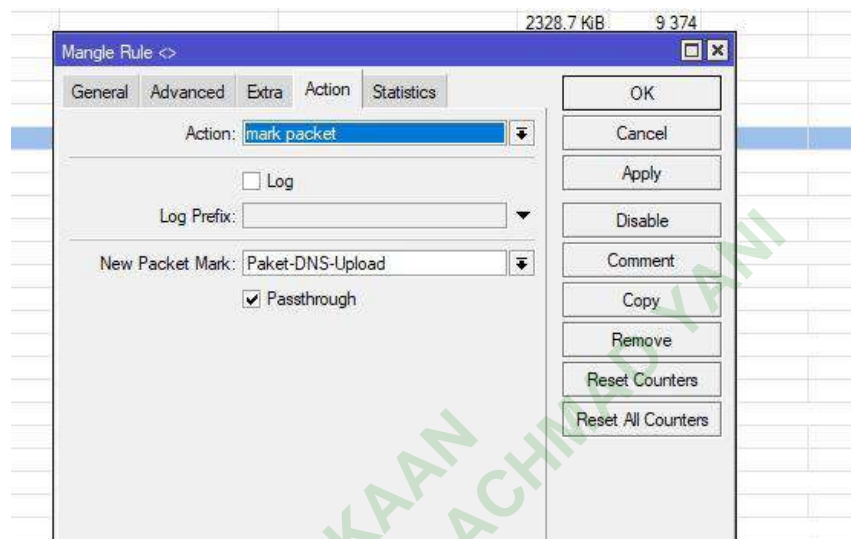
Gambar 4.16 *Mark packet mangle Action-DNS download*

6. Pada tab *mangle-general*, *chain* adalah *forward*, *out interface* adalah *ether1-internet*, dan pada bagian *Connection Mark* adalah *Koneksi-DNS*, seperti yang terlihat pada gambar 4.17 berikut.



Gambar 4.17 *Mark packet mangle general-DNS Upload*

7. Pada tab *action*, kolom *action* adalah *Mark packet*, dan *New Packet Mark* bernama *Paket-DNS-Upload*, seperti yang terlihat pada gambar 4.18 berikut.

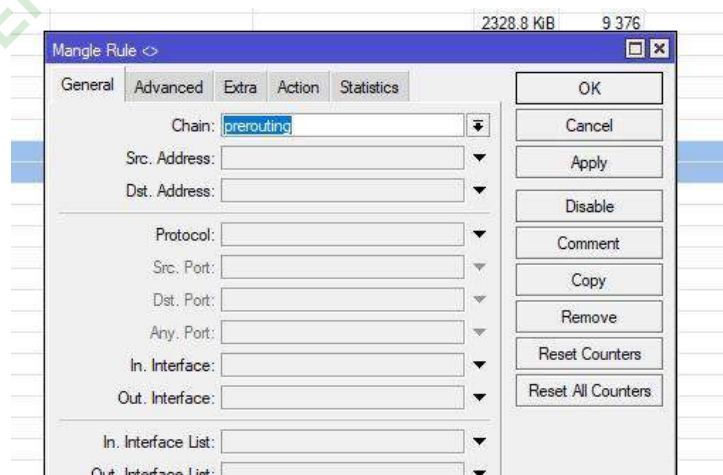


Gambar 4.18 *Mark packet mangle Action-DNS Upload*

4.5.3 Konfigurasi *Mangle Youtube*

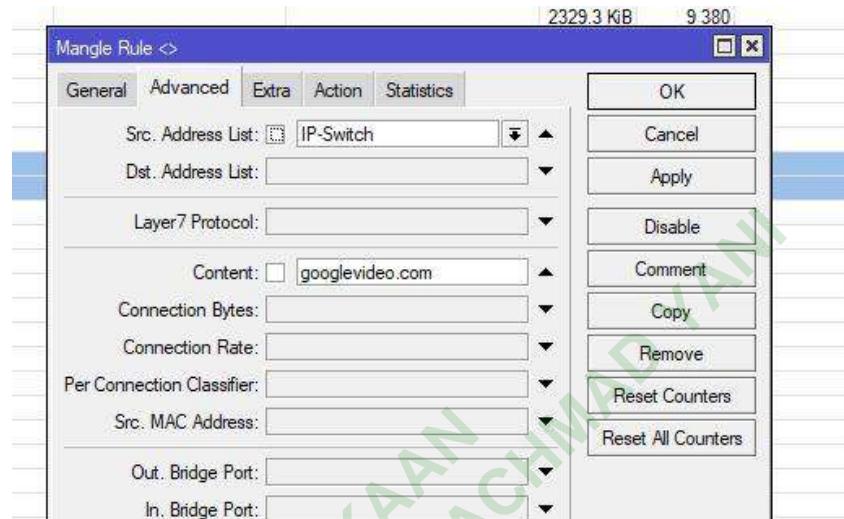
Berikut adalah langkah-langkah yang dilakukan untuk konfigurasi *mangle Youtube*

1. Pada menu *IP*, *Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*, seperti yang terlihat pada gambar 4.19. Pada tab *general*, kolom *chain* menggunakan *prerouting*.



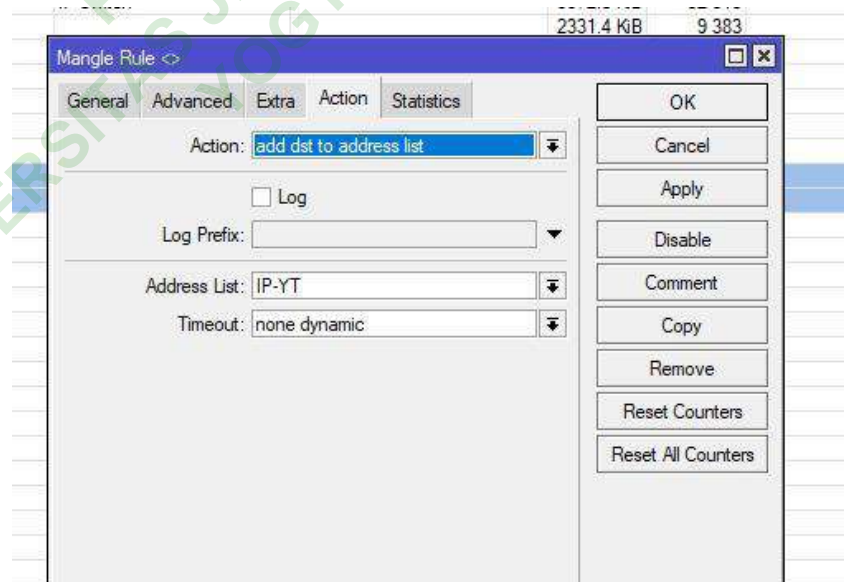
Gambar 4.19 *Mark address mangle General-youtube*

2. Pada tab *advanced*, kolom *src.address list* adalah *IP-SWITCH* dan kolom *content* berisi *URL youtube*, yaitu *googlevideo.com*, seperti yang terlihat pada gambar 4.20 berikut.



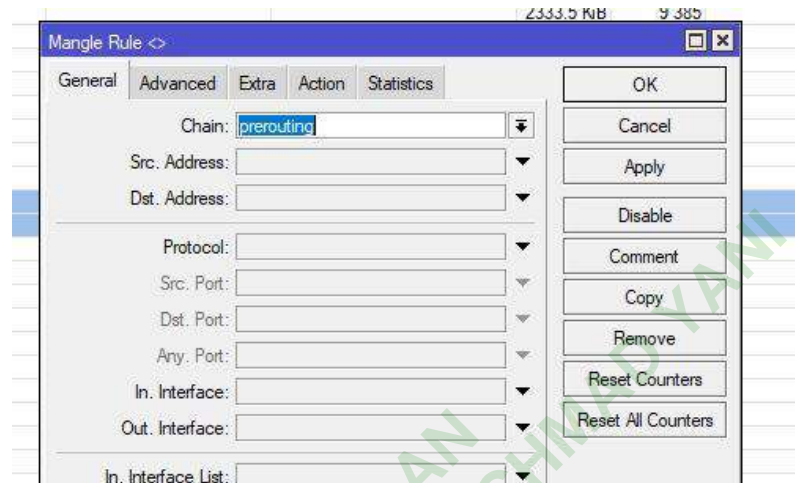
Gambar 4.20 Mark address mangle Advanced-youtube

3. Pada tab *action*, kolom *action* adalah *add dst to address list*, dan kolom *address list* berisi *IP-YT*. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.21 berikut.



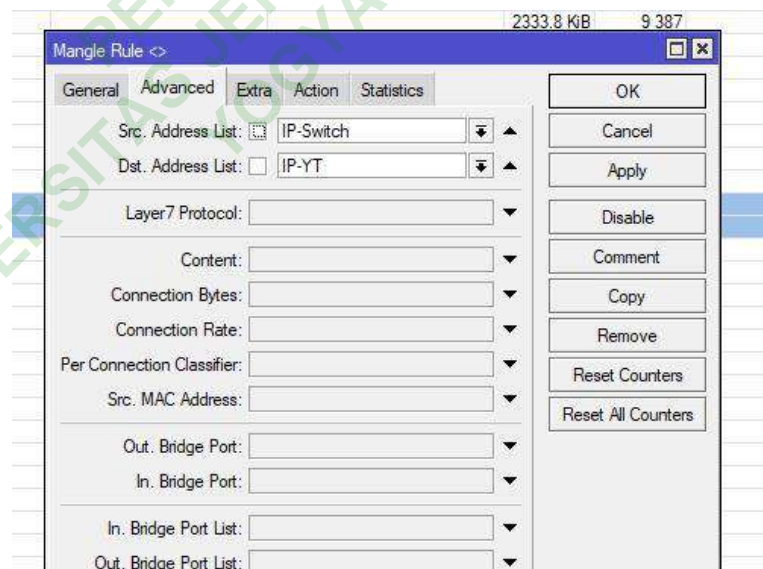
Gambar 4.21 Mark address mangle Action-youtube

4. Pada menu *IP, Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*, seperti yang terlihat pada gambar 4.22. Pada tab *general*, kolom *chain* adalah *prerouting*.



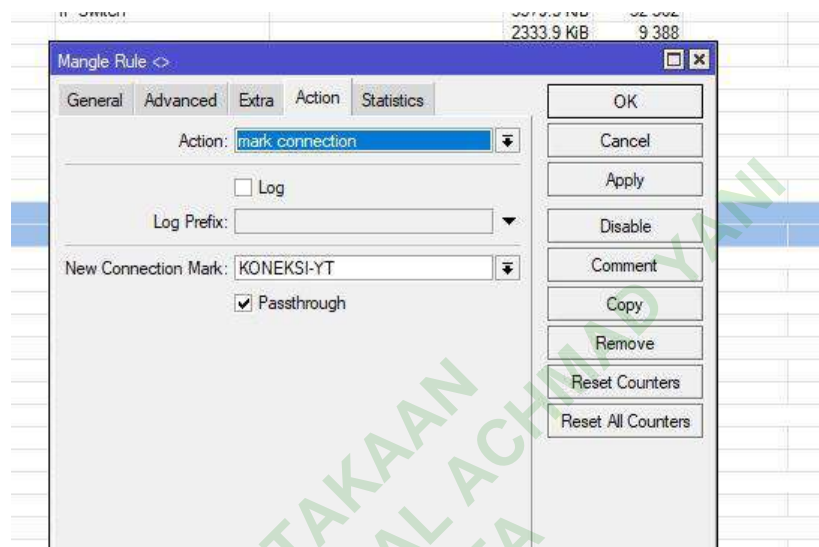
Gambar 4.22 Mark connections mangle General-youtube

5. Pada tab *advanced*, kolom *src.address list* adalah *IP-SWITCH* dan kolom *dst.address list* adalah *IP-YT*, seperti yang terlihat pada gambar 4.23 berikut.



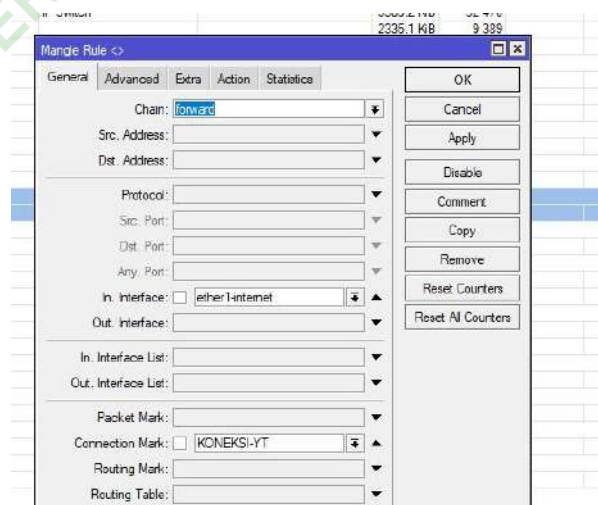
Gambar 4.23 Mark connections mangle Advanced-youtube

6. Pada tab *action*, kolom *action* adalah *Mark connection* dan *New Connection Mark* bernama KONEKSI-YT. *Checklist* pada bagian *passthrough* diaktifkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.24 berikut.



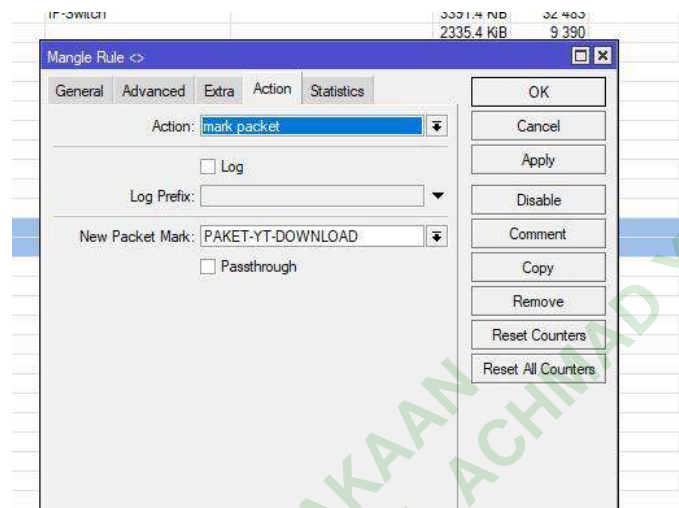
Gambar 4.24 *Mark connections mangle Action-youtube*

7. Pada tab *mangle*, (+) digunakan untuk menambahkan pengaturan *mangle* yang menandai paket youtube *download*. Pada bagian tab *general*, kolom *chain* adalah *forward*, *in.interface* adalah *ether1-internet*, dan kolom *Connection Mark* adalah KONEKSI-YT, seperti yang terlihat pada gambar 4.25 berikut.



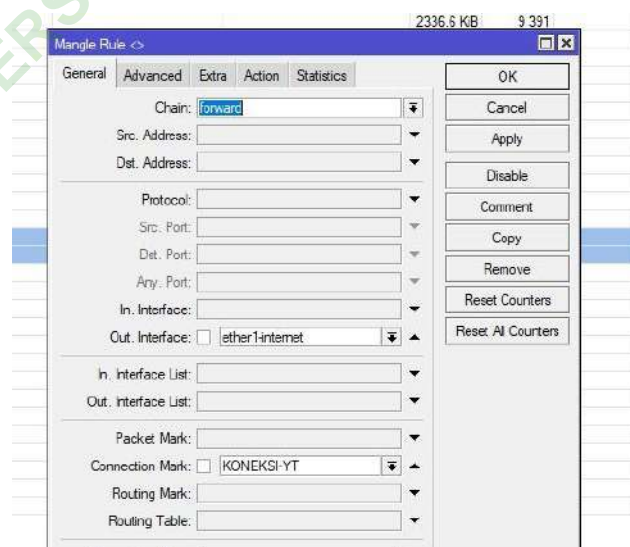
Gambar 4.25 *Mangle Mark packet general-youtube-download*

8. Pada tab *action*, kolom *action* adalah *Mark packet* dan kolom *New Packet Mark* berisi *PAKET-YT-DOWNLOAD*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.26 berikut.



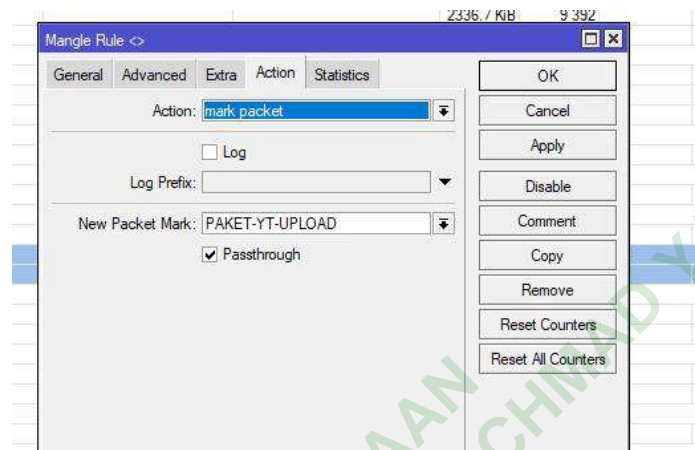
Gambar 4.26 Mangle Mark packet action-youtube-download

9. Pada tab *mangle*, (+) digunakan untuk menambahkan pengaturan *mangle* yang menandai paket youtube *upload*. Kolom *chain* adalah *forward*, kolom *out.interface* adalah *ether1-internet*, dan kolom *Connection Mark* adalah *KONEKSI-YT*, seperti yang terlihat pada gambar 4.27 berikut.



Gambar 4.27 Mangle Mark packet general-youtube-upload

10. Pada tab *action*, kolom *action* adalah *Mark packet* dan *New Packet Mark* bernama *PAKET-YT-UPLOAD*. *Checklist* pada bagian *passthrough* diaktifkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.28 berikut.

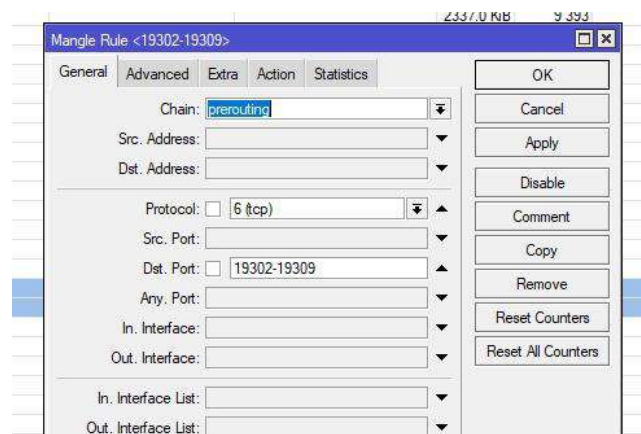


Gambar 4.28 Mangle Mark packet action-youtube-upload

4.5.4 Konfigurasi Mangle Google Meet

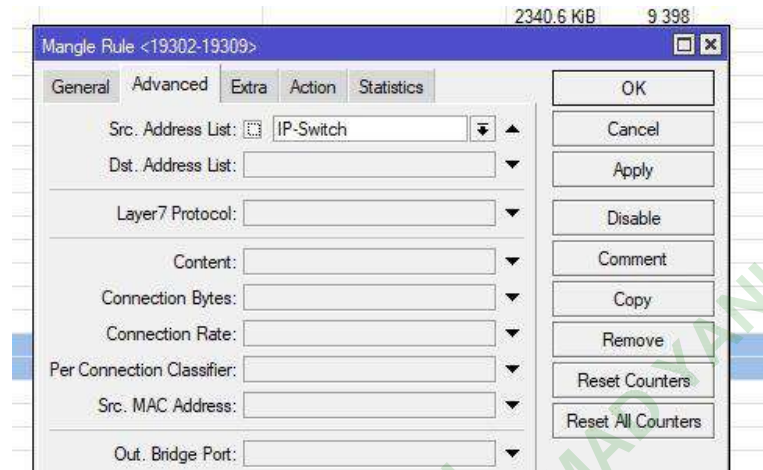
Berikut adalah langkah-langkah yang dilakukan untuk konfigurasi *mangle* Google Meet

1. Pada menu *IP, Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *prerouting*, kolom *protocol* berisi *TCP*, dan kolom *dst.port* adalah *port* dari Google Meet, yaitu 19302-19309, seperti yang terlihat pada gambar 4.29 berikut.



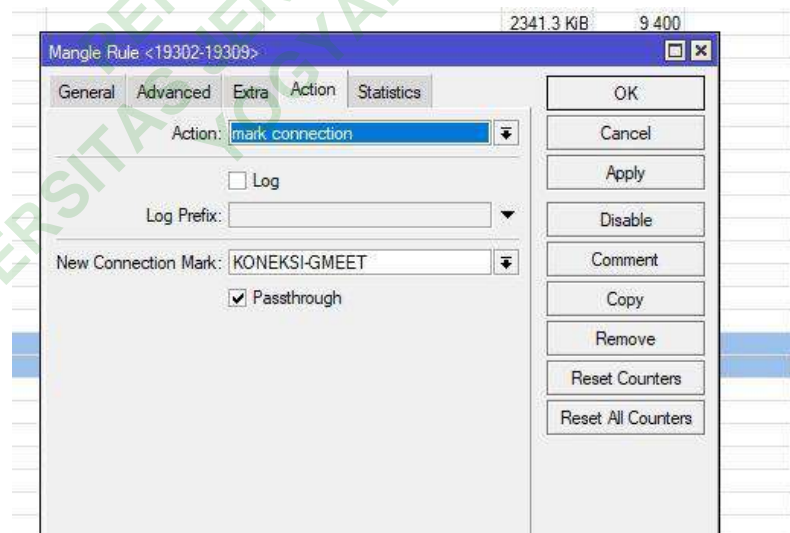
Gambar 4.29 Mangle Mark connection general-google meet (TCP)

2. Pada tab *advanced*, kolom *src.address list* menggunakan *IP-SWITCH*, seperti pada gambar 4.30 berikut



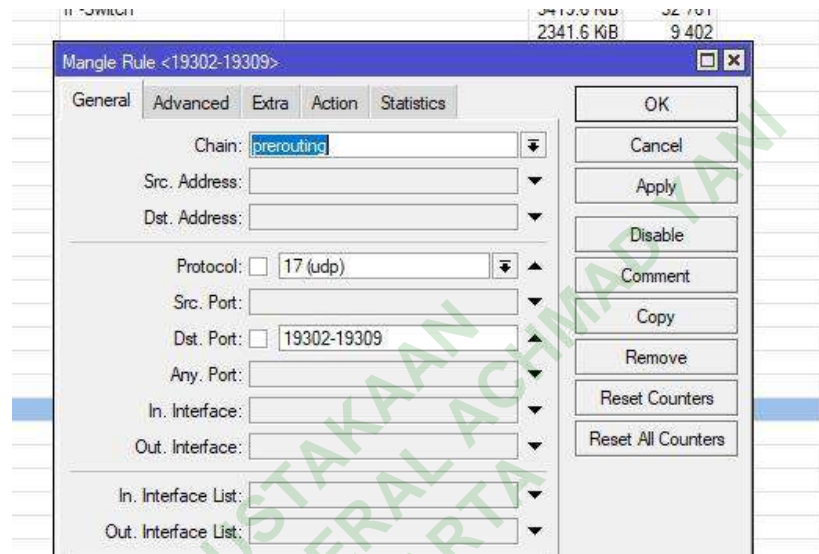
Gambar 4.30 Mangle Mark connection advanced-google meet (TCP)

3. Pada tab *action*, kolom *action* adalah *Mark connection* dan kolom *New Connection Mark* berisi *KONEKSI-GMEET*. *Checklist* pada bagian *passthrough* diaktifkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.31 berikut.



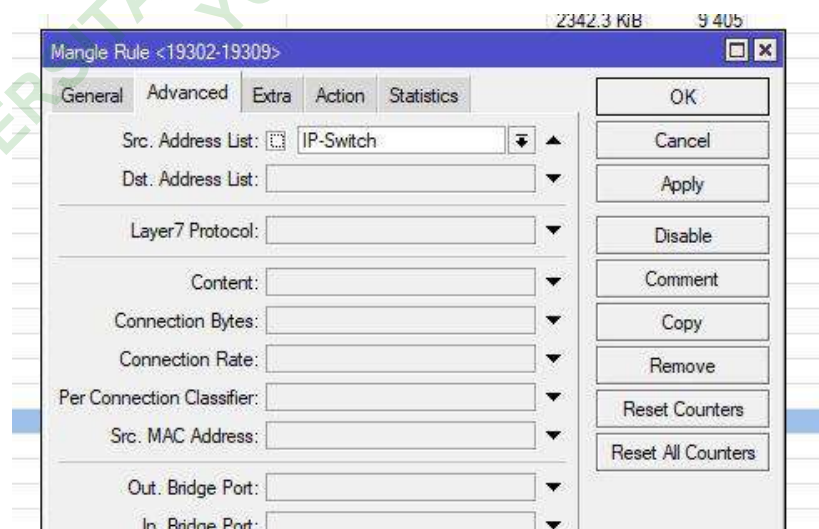
Gambar 4.31 Mangle Mark connection action-google meet (TCP)

4. Pada menu *IP, Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *prerouting*, kolom *protocol* berisi *UDP*, dan kolom *dst.port* adalah *port* Google Meet, yaitu 19302-19309, seperti yang terlihat pada gambar 4.32 berikut.



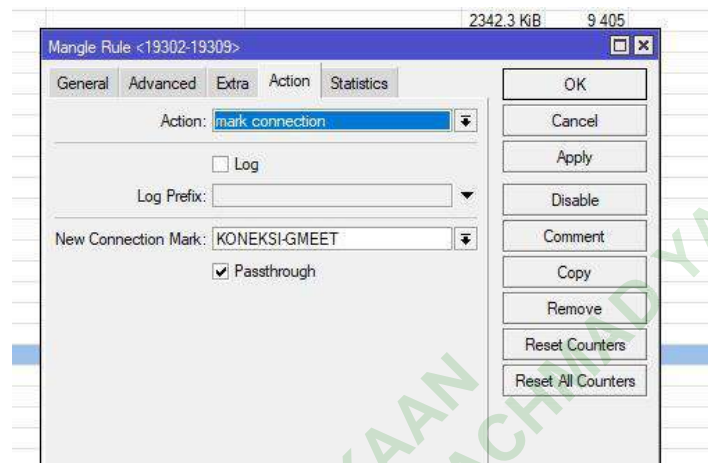
Gambar 4.32 Mangle Mark connection General-google meet (UDP)

5. Pada tab *advanced*, kolom *src.address list* menggunakan *IP-SWITCH*, seperti pada gambar 4.33 berikut.



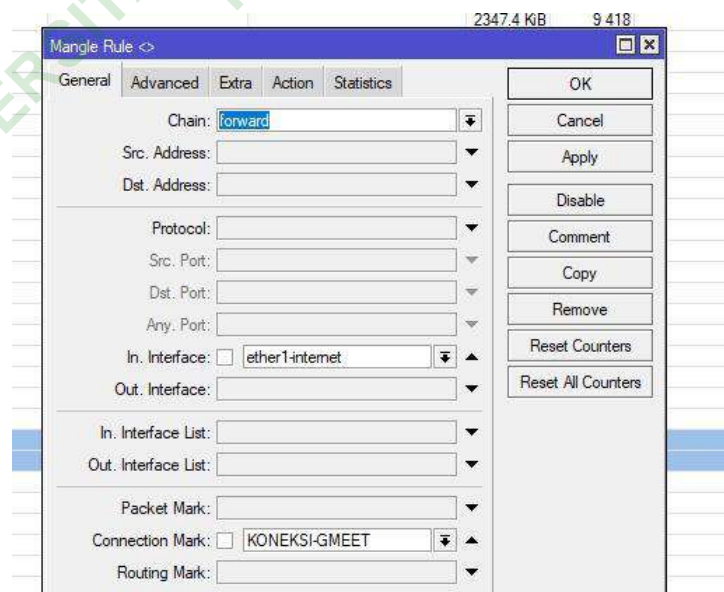
Gambar 4.33 Mangle Mark connection Advaced-google meet (UDP)

6. Pada tab *action*, kolom *action* adalah *Mark connection* dan *New Connection Mark* bernama KONEKSI-GMEET. *Checklist* pada bagian *passthrough* diaktifkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.34 berikut.



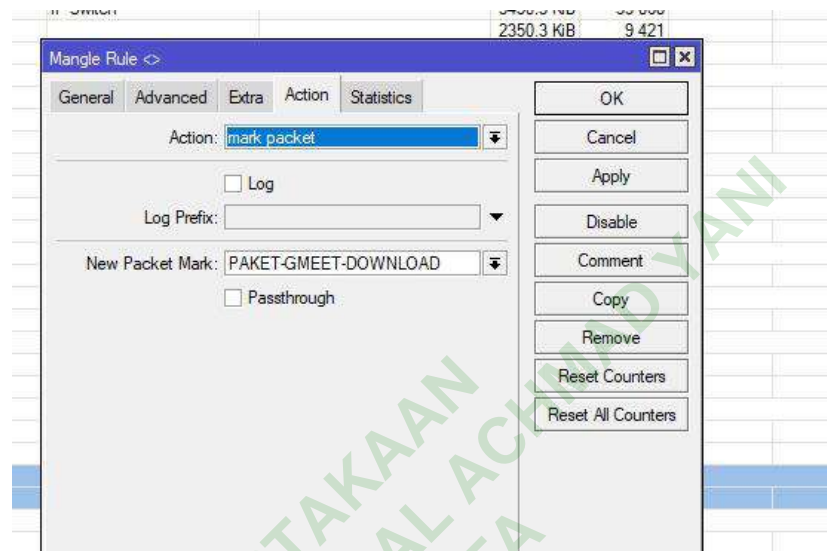
Gambar 4.34 Mangle Mark connection Action-google meet (UDP)

7. Pada menu *IP*, *Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *forward*, kolom *in.interface* adalah *ether1-internet*, dan *Connection Mark* adalah KONEKSI-GMEET, seperti yang terlihat pada gambar 4.35 berikut.



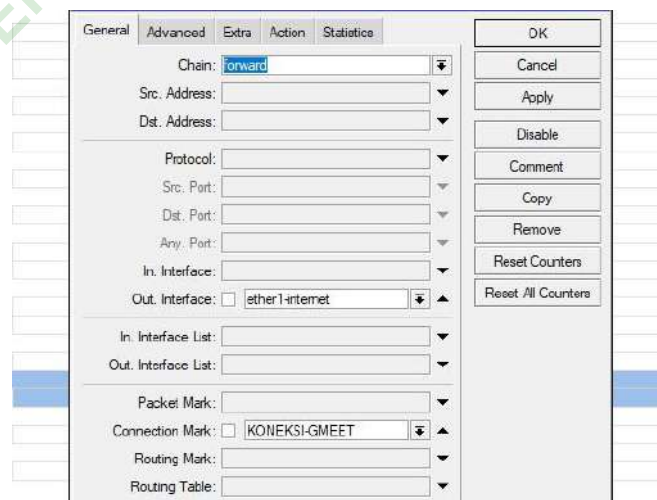
Gambar 4.35 Mangle Mark packet General-google meet download

8. Pada tab *action*, kolom *action* adalah *Mark packet* dan kolom *New Packet Mark* bernama *PAKET-GMEET-DOWNLOAD*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.36 berikut.



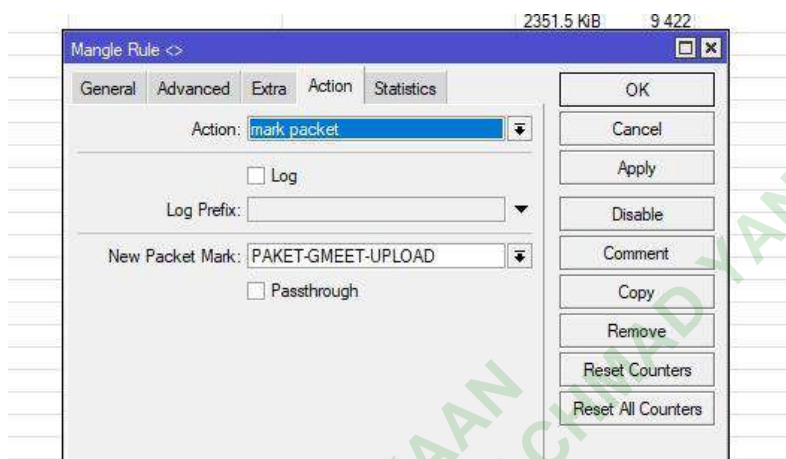
Gambar 4.36 Mangle Mark packet Action-google meet download

9. Pada menu *IP, Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *forward*, kolom *out.interface* adalah *ether1-internet*, dan *Connection Mark* adalah *KONEKSI-GMEET*, seperti yang terlihat pada gambar 4.37 berikut.



Gambar 4.37 Mangle Mark packet General-google meet upload

10. Pada tab *action*, kolom *action* adalah *Mark packet* dan kolom *New Packet Mark* berisi *PAKET-GMEET-UPLOAD*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.38 berikut.

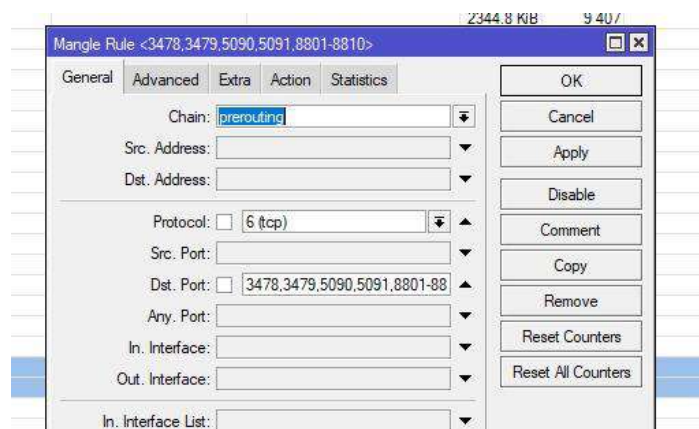


Gambar 4.38 Mangle Mark packet Action-google meet upload

4.5.5 Konfigurasi Mangle Zoom Meet

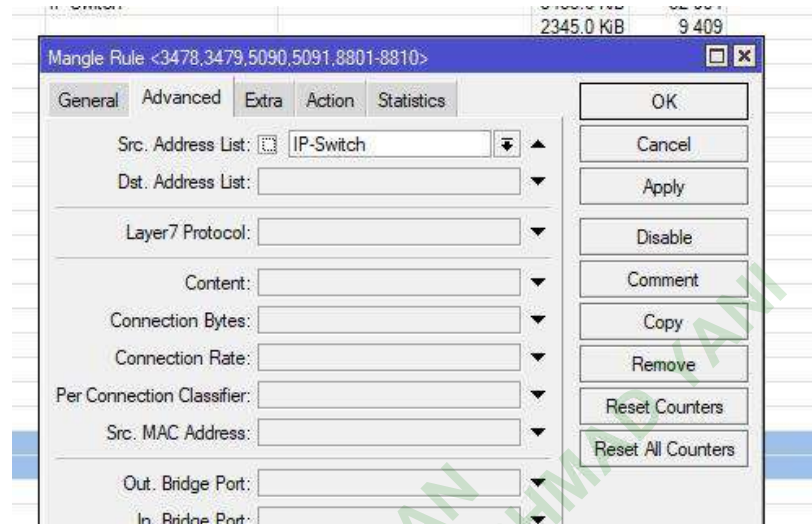
Berikut adalah langkah-langkah yang dilakukan untuk konfigurasi *mangle* Zoom Meet

1. Pada menu *IP*, *Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *prerouting*, kolom *protocol* adalah *TCP*, dan kolom *dst.port* berisi *port TCP* dari Zoom, yaitu 3478, 3479, 5090, 5091, 8801-8810, seperti yang terlihat pada gambar 4.39 berikut.



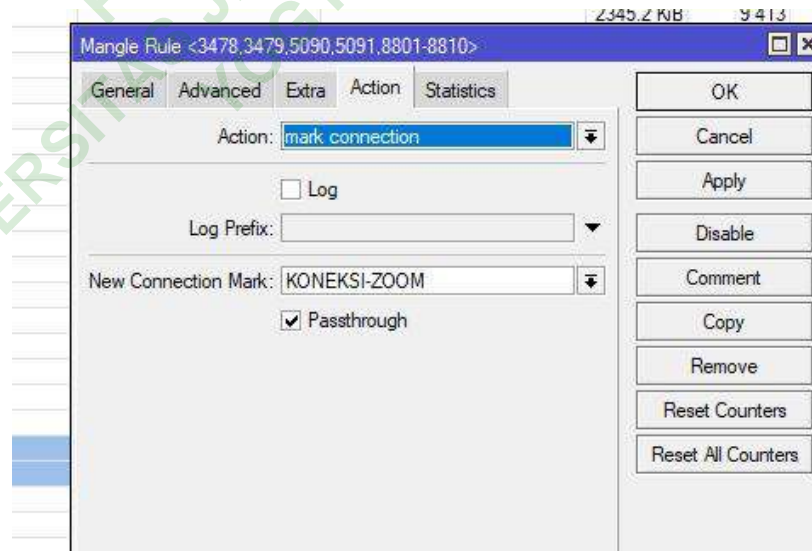
Gambar 4.39 Mangle Mark connection general-zoom meet (TCP)

2. Pada tab *advanced*, kolom *src.address list* menggunakan *IP-SWITCH*, seperti pada gambar 4.40 berikut.



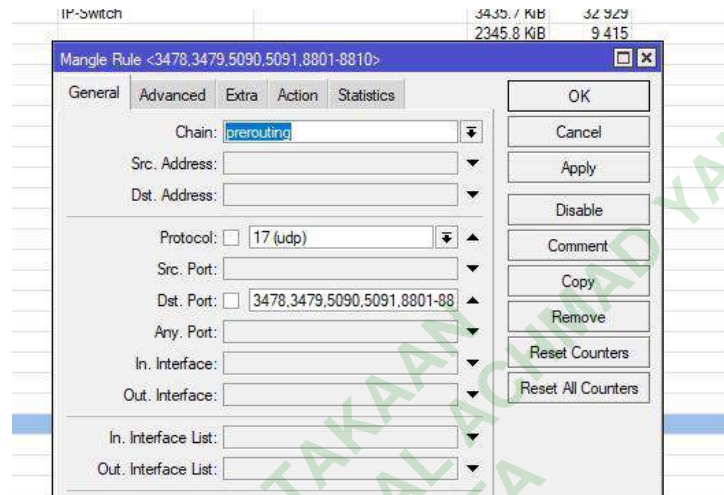
Gambar 4.40 Mangle Mark connection Advanced-zoom meet (TCP)

3. Pada tab *action*, kolom *action* adalah *Mark connection* dan kolom *New Connection Mark* berisi *KONEKSI-ZOOM*. *Checklist* pada bagian *passthrough* diaktifkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.41 berikut.



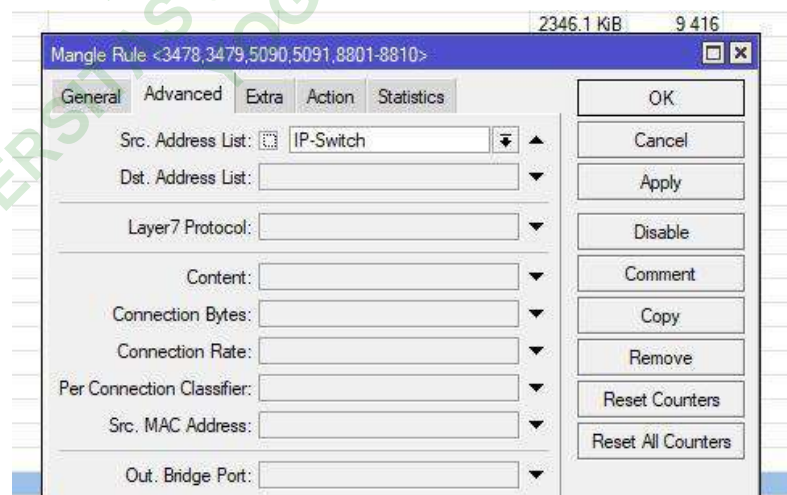
Gambar 4.41 Mangle Mark connection Action-zoom meet (TCP)

4. Pada menu *IP*, *Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *prerouting*, kolom *protocol* adalah *UDP*, dan *dst.port* berisi *port Zoom Meet*, yaitu 3478, 3479, 5090, 5091, 8801-8810. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.42 berikut.



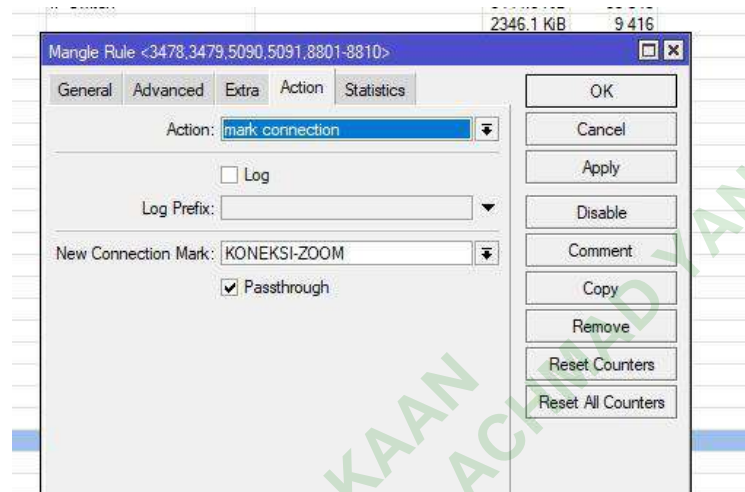
Gambar 4.42 Mangle Mark connection General-zoom meet (UDP)

5. Pada tab *advanced*, kolom *src.address list* menggunakan *IP-SWITCH*, seperti pada gambar 4.43 berikut.



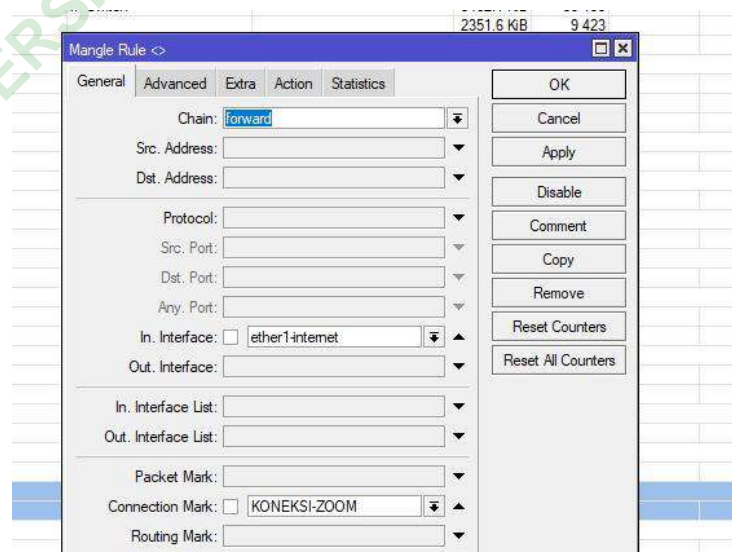
Gambar 4.43 Mangle Mark connection Advanced-zoom meet (UDP)

6. Pada tab *action*, kolom *action* adalah *Mark connection* dan kolom *New Packet Mark* berisi *KONEKSI-ZOOM*. *Checklist* pada bagian *passthrough* diaktifkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.44 berikut.



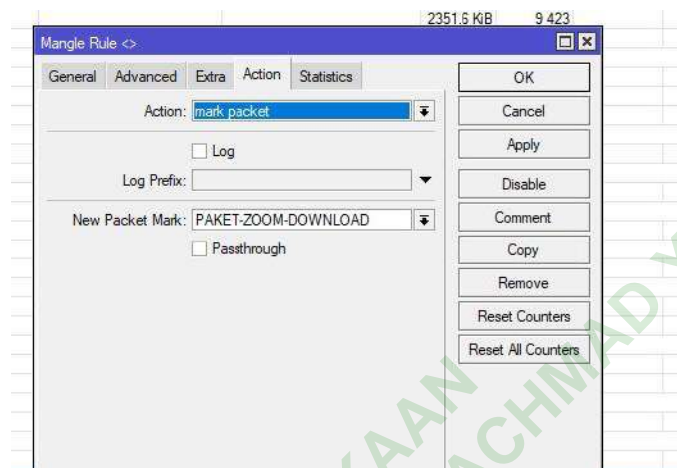
Gambar 4.44 Mangle Mark connection Action-zoom meet (UDP)

7. Pada menu *IP*, *Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *forward*, kolom *in.interface* adalah *ether1-internet*, dan kolom *Connection Mark* adalah *KONEKSI-ZOOM*, seperti yang terlihat pada gambar 4.45 berikut.



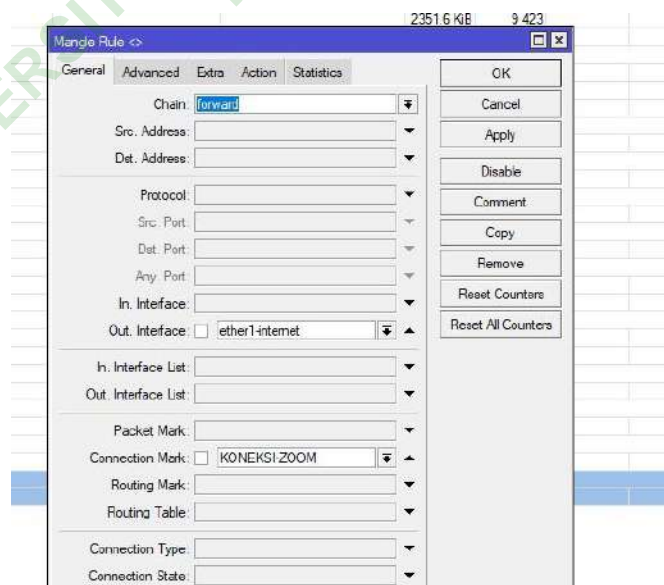
Gambar 4.45 Mangle Mark packet General-zoom meet download

8. Pada tab *action*, kolom *action* adalah *Mark packet* dan kolom *New Packet Mark* berisi *PAKET-ZOOM-DOWNLOAD*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.46 berikut.



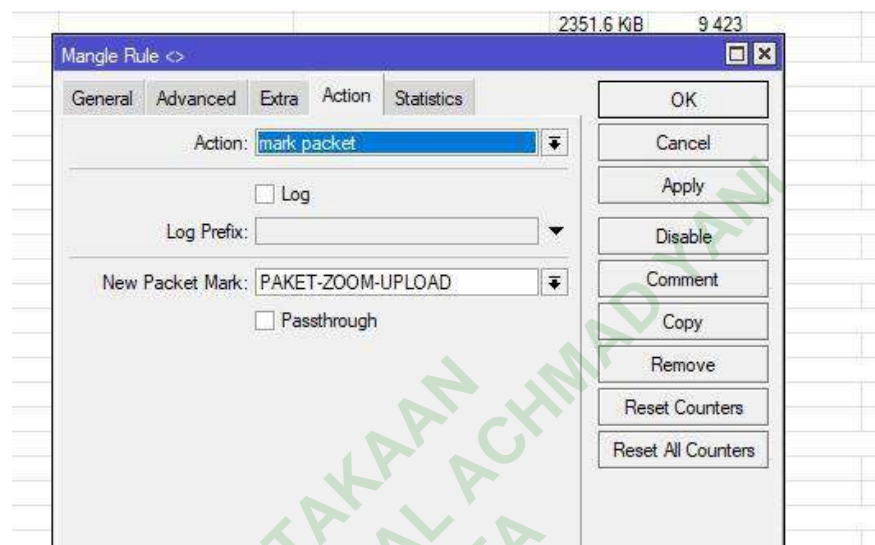
Gambar 4.46 Mangle Mark packet Action-zoom meet download

9. Pada menu *IP*, *Firewall*, dan tab *mangle* digunakan. Tanda (+) digunakan untuk menambahkan *mangle*. Pada tab *general*, kolom *chain* adalah *forward*, kolom *out.interface* adalah *ether1-internet*, dan kolom *Connection Mark* adalah *KONEKSI-ZOOM*, seperti yang terlihat pada gambar 4.47 berikut.



Gambar 4.47 Mangle Mark packet General-zoom meet upload

10. Pada tab *action*, kolom *action* adalah *Mark packet* dan kolom *New Packet Mark* berisi *PAKET-ZOOM-UPLOAD*. *Checklist* pada bagian *passthrough* dihilangkan, kemudian *apply* dan *ok*, seperti yang terlihat pada gambar 4.48 berikut.



Gambar 4.48 Mangle Mark packet Action-zoom meet upload

4.5.6 Konfigurasi Mangle Koneksi Umum

Untuk konfigurasi koneksi umum, caranya sama seperti langkah-langkah di atas, dengan penambahan *port* untuk *protocol TCP* dan *UDP*, yaitu 80, 81, 443, 8000-8081, 21, 22, 23, 81, 88, 5050, 843, 182, dan 53. Hasil konfigurasi dapat dilihat pada gambar 4.49 berikut.

::: KONEKSI-UMUM							
18	mark connection	prerouting	6 (tcp)	80,81,443,8000-8081,21,22,23,8...			IP-Switch
19	mark connection	prerouting	17 (udp)	80,81,443,8000-8081,21,22,23,8...			IP-Switch
::: PAKET UMUM DOWNLOAD							
20	mark packet	forward				ether1-internet	
::: PAKET UMUM UPLOAD							
21	mark packet	forward				ether1-internet	

Gambar 4.49 Konfigurasi mangle koneksi umum

4.5.7 Hasil Konfigurasi Mangle

Berikut adalah hasil konfigurasi *mangle* mikrotik yang telah berhasil di buat, seperti pada gambar 4.50 berikut.

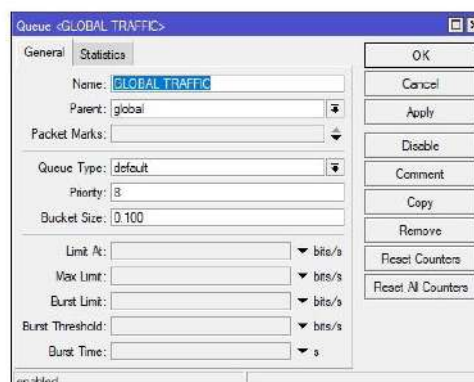
#	Action	Chain	Src. Address	Protocol	Src. Port	Dst. Port	In. Interface	Out. Interface	In. Inter.	Out. Int.	Src. Address List	Dst. Address List	Bytes	Packets	Dst. Address
0	rule														
1	rule														
2	rule														
3	rule														
4	rule														
5	rule														
6	rule														
7	rule														
8	rule														
9	rule														
10	rule														
11	rule														
12	rule														
13	rule														
14	rule														
15	rule														
16	rule														
17	rule														
18	rule														
19	rule														
20	rule														
21	rule														

Gambar 4.50 Hasil konfigurasi *mangle* mikrotik

4.6 KONFIGURASI *QUEUE TREE* PADA MIKROTIK

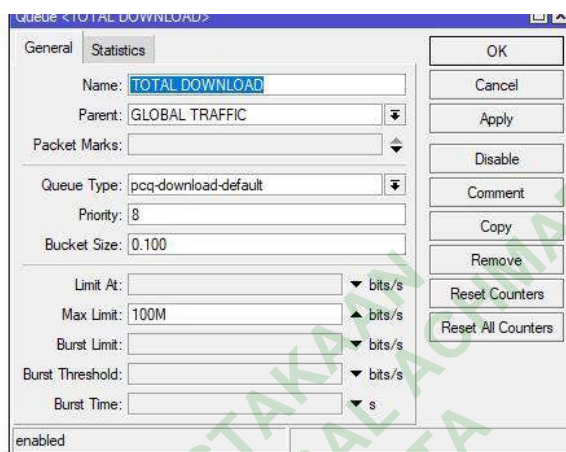
Queue tree adalah fitur manajemen jaringan yang membagi *bandwidth* ke antrian berdasarkan prioritas. Tujuannya adalah untuk mengoptimalkan penggunaan *bandwidth* dan memberikan prioritas pada jenis lalu lintas tertentu. Dengan ini, *Queue tree* mencegah kemacetan dan menjaga kinerja jaringan tetap optimal. Berikut langkah-langkah yang dilakukan :

1. Pertama, *parent global traffic download dan upload pada Queue tree*. Menu *queues*, tab *Queue tree*, dan (+) digunakan untuk menambahkan konfigurasi. Pada tab *general*, kolom *name* adalah *GLOBAL TRAFFIC*, bagian *parent* adalah *global*, dan kolom *queue type* adalah *default*. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.51 berikut.



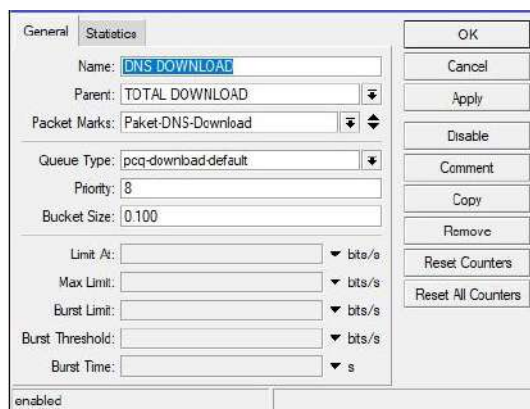
Gambar 4.51 *Global traffic*

2. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk total *download*. Pada tab *general*, kolom *name* adalah *TOTAL DOWNLOAD*, *parent* adalah *GLOBAL TRAFFIC*, *queue type* adalah *pcq-download-default*, dan *max limit* adalah 100M (d disesuaikan dengan kecepatan *download* dari *ISP*). *Apply* dan *ok*, seperti yang terlihat pada gambar 4.52 berikut.



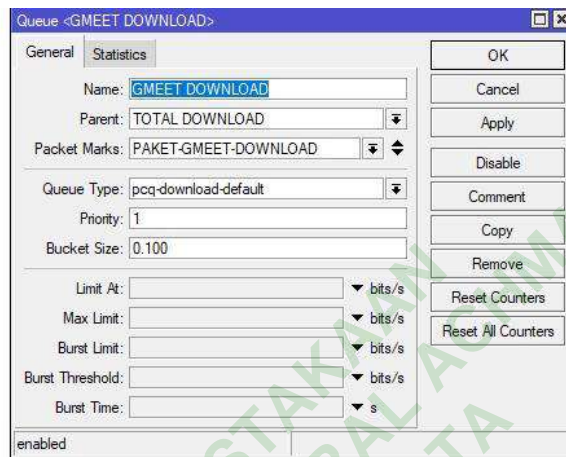
Gambar 4.52 Total download

3. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *DNS download*. Pada tab *general*, kolom *name* adalah *DNS DOWNLOAD*, kolom *parent* adalah *TOTAL DOWNLOAD*, kolom *Packet Mark* adalah *Packet-DNS-Download*, dan kolom *queue type* adalah *pcq-download-default*, seperti yang terlihat pada gambar 4.53 berikut.



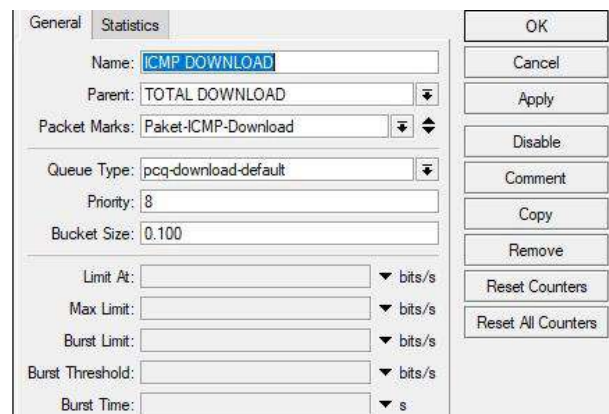
Gambar 4.53 Total download-DNS

4. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk Google Meet *download*. Pada tab *general*, kolom *name* adalah *GMEET DOWNLOAD*, kolom *parent* adalah *TOTAL DOWNLOAD*, kolom *Packet Mark* adalah *PACKET-GMEET-DOWNLOAD*, kolom *queue type* adalah *pcq-download-default*, dan *priority* adalah 1, seperti yang terlihat pada gambar 4.54 berikut.



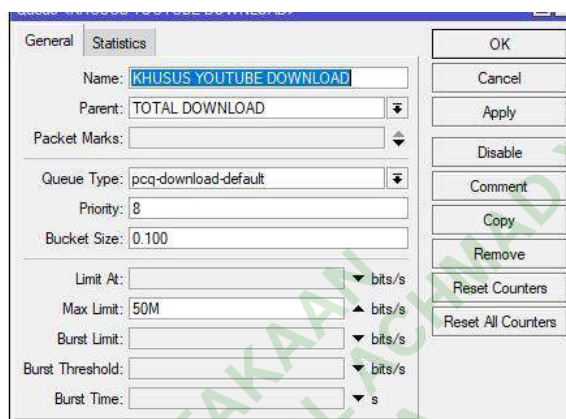
Gambar 4.54 Total *download*-google meet

5. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *ICMP download*. Pada tab *general*, kolom *name* adalah *ICMP DOWNLOAD*, kolom *parent* adalah *TOTAL DOWNLOAD*, kolom *Packet Mark* adalah *Packet-ICMP-Download*, dan kolom *queue type* adalah *pcq-download-default*, seperti yang terlihat pada gambar 4.55 berikut.



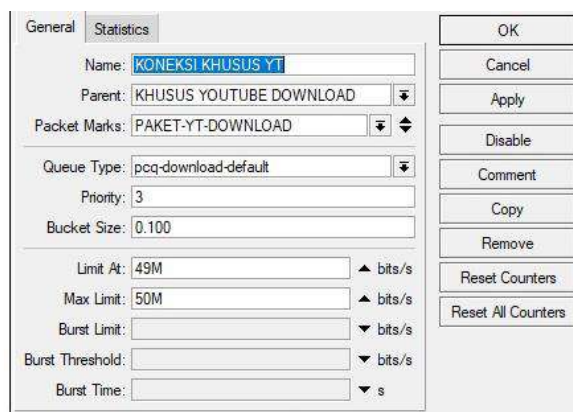
Gambar 4.55 Total *download*-ICMP

6. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *parent traffic* YouTube *download*. Pada tab *general*, kolom *name* adalah KHUSUS YOUTUBE DOWNLOAD, kolom *parent* adalah TOTAL DOWNLOAD, kolom *queue type* adalah *pcq-download-default*, dan *max limit* adalah 50M, seperti yang terlihat pada gambar 4.56 berikut.



Gambar 4.56 Parent total youtube-download

7. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *child* koneksi YouTube *download*. Pada tab *general*, kolom *name* adalah KONEKSI KHUSUS YT, kolom *parent* adalah KHUSUS YOUTUBE DOWNLOAD, kolom *Packet Mark* adalah *PACKET-YT-DOWNLOAD*, kolom *queue type* adalah *pcq-download-default*, *priority* adalah 3, kolom *limit* adalah 49M, dan *max limit* adalah 50M, seperti yang terlihat pada gambar 4.57 berikut.



Gambar 4.57 Child total youtube-download

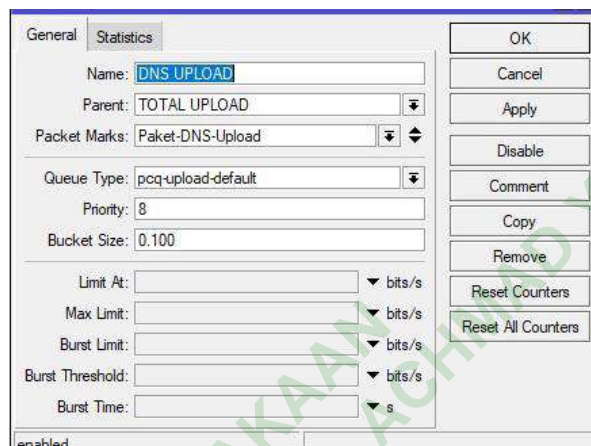
8. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *Zoom download*. Pada tab *general*, kolom *name* adalah *ZOOM DOWNLOAD*, kolom *parent* adalah *TOTAL DOWNLOAD*, kolom *Packet Mark* adalah *PACKET ZOOM DOWNLOAD*, kolom *queue type* adalah *pcq-download-default*, dan *priority* adalah 2, seperti yang terlihat pada gambar 4.58 berikut.

Gambar 4.58 Total download-zoom meet

9. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk total *upload*. Pada tab *general*, kolom *name* adalah *TOTAL UPLOAD*, kolom *parent* adalah *GLOBAL TRAFFIC*, kolom *queue type* adalah *pcq-upload-default*, dan *max limit* adalah 100M (d disesuaikan dengan kecepatan upload dari *ISP*). *Apply* dan *ok*, seperti yang terlihat pada gambar 4.59 berikut.

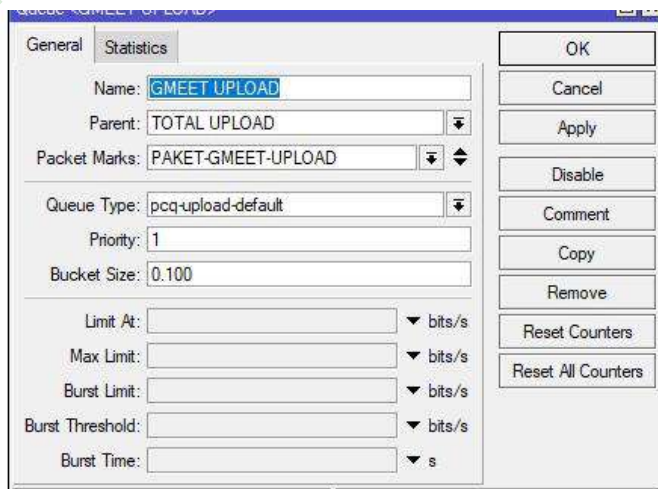
Gambar 4.59 Total upload

10. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *DNS upload*. Pada tab *general*, kolom *name* adalah *DNS UPLOAD*, kolom *parent* adalah *TOTAL UPLOAD*, kolom *Packet Mark* adalah *Packet-DNS-Upload*, dan kolom *queue type* adalah *pcq-upload-default*, seperti yang terlihat pada gambar 4.60 berikut.



Gambar 4.60 Total upload-DNS

11. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *Google Meet upload*. Pada tab *general*, kolom *name* adalah *GMEET UPLOAD*, kolom *parent* adalah *TOTAL UPLOAD*, kolom *Packet Mark* adalah *PACKET-GMEET-UPLOAD*, kolom *queue type* adalah *pcq-upload-default*, dan *priority* adalah 1, seperti yang terlihat pada gambar 4.61 berikut.



Gambar 4.61 Total upload-google meet

12. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *ICMP upload*. Pada tab *general*, kolom *name* adalah *ICMP UPLOAD*, kolom *parent* adalah *TOTAL UPLOAD*, kolom *Packet Mark* adalah *Packet-ICMP-Upload*, dan kolom *queue type* adalah *pcq-upload-default*, seperti yang terlihat pada gambar 4.62 berikut.

The screenshot shows the configuration window for a queue named 'ICMP UPLOAD'. The 'Name' field is 'ICMP UPLOAD', 'Parent' is 'TOTAL UPLOAD', 'Packet Marks' is 'Paket-ICMP-Upload', and 'Queue Type' is 'pcq-upload-default'. The 'Priority' is set to 8 and 'Bucket Size' is 0.100. There are fields for 'Limit At', 'Max Limit', 'Burst Limit', 'Burst Threshold', and 'Burst Time', all with dropdown menus for units (bits/s or s). The 'enabled' checkbox is checked. On the right, there are buttons for 'Cancel', 'Apply', 'Disable', 'Comment', 'Copy', 'Remove', 'Reset Counters', and 'Reset All Counters'.

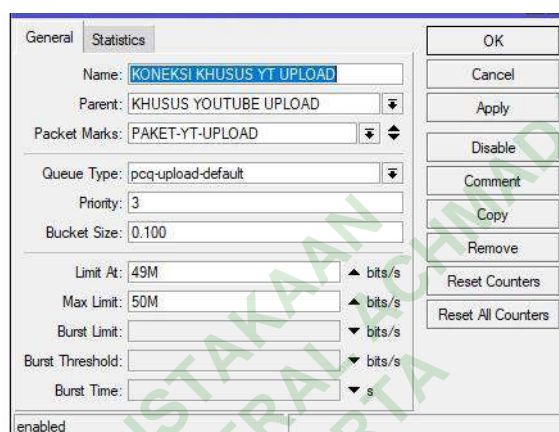
Gambar 4.62 Total upload-ICMP

13. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *parent traffic YouTube upload*. Pada tab *general*, kolom *name* adalah *KHUSUS YOUTUBE UPLOAD*, kolom *parent* adalah *TOTAL UPLOAD*, kolom *queue type* adalah *pcq-upload-default*, dan *max limit* adalah 50M, seperti yang terlihat pada gambar 4.63 berikut.

The screenshot shows the configuration window for a queue named 'KHUSUS YOUTUBE UPLOAD'. The 'Name' field is 'KHUSUS YOUTUBE UPLOAD', 'Parent' is 'TOTAL UPLOAD', and 'Queue Type' is 'pcq-upload-default'. The 'Priority' is set to 8 and 'Bucket Size' is 0.100. The 'Max Limit' is set to 50M. There are fields for 'Limit At', 'Burst Limit', 'Burst Threshold', and 'Burst Time', all with dropdown menus for units (bits/s or s). The 'enabled' checkbox is checked. On the right, there are buttons for 'OK', 'Cancel', 'Apply', 'Disable', 'Comment', 'Copy', 'Remove', 'Reset Counters', and 'Reset All Counters'.

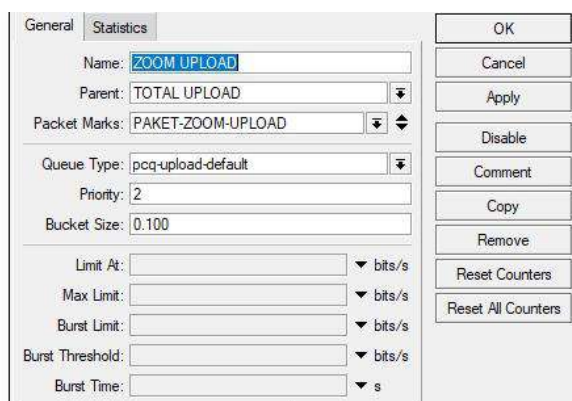
Gambar 4.63 Parent total youtube-upload

14. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk *child* koneksi YouTube *upload*. Pada tab *general*, kolom *name* adalah KONEKSI KHUSUS YT UPLOAD, kolom *parent* adalah KHUSUS YOUTUBE UPLOAD, kolom *Packet Mark* adalah PACKET-YT-UPLOAD, kolom *queue type* adalah *pcq-upload-default*, *priority* adalah 3, kolom *limit* adalah 49M, dan *max limit* adalah 50M, seperti yang terlihat pada gambar 4.64 berikut.



Gambar 4.64 Child total youtube-upload

15. Pada tab *Queue tree*, (+) digunakan untuk menambahkan konfigurasi untuk Zoom meet *upload*. Pada tab *general*, kolom *name* adalah ZOOM UPLOAD, kolom *parent* adalah TOTAL UPLOAD, kolom *Packet Mark* adalah PACKET ZOOM UPLOAD, kolom *queue type* adalah *pcq-upload-default*, dan *priority* adalah 2, seperti yang terlihat pada gambar 4.65 berikut.



Gambar 4.65 Total upload-zoom meet

4.6.1 Hasil Konfigurasi Queue tree

Berikut adalah hasil konfigurasi *Queue tree* mikrotik yang telah berhasil di buat, seperti pada gambar 4.66 berikut.

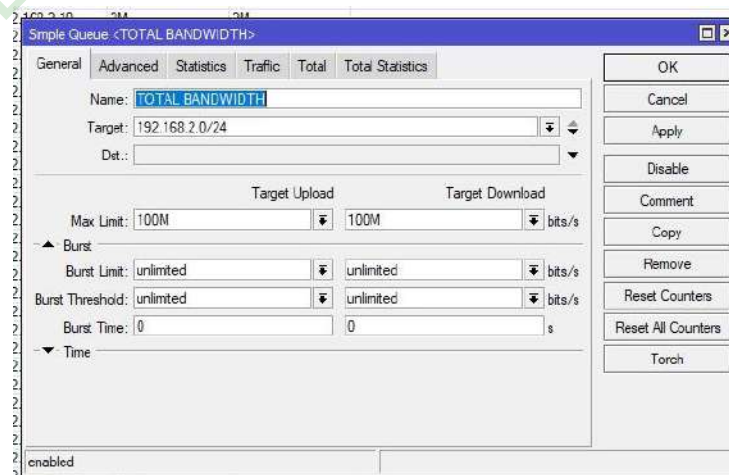
Name	Parent	Packet Mark	Limit At B.	Max Limit	Avg. Rate	Queued Bytes	Bytes	Packets
GLOBAL TRAFFIC	global				15.6 Mbps	0 B	231.9	276.02
TOTAL DOWNLOAD	GLOBAL TRAFFIC			100M	15.4 Mbps	0 B	218.9	199.55
DNS DOWNLOAD	TOTAL DOWNLOAD	Paket-DNS-Download			0bps	0 B	486.7	3.110.9
GMEET DOWNLOAD	TOTAL DOWNLOAD	PAKET-GMEET-DOWNLOAD			0bps	0 B	835.0	9.620
ICMP DOWNLOAD	TOTAL DOWNLOAD	Paket-ICMP-Download			5.3 kbps	0 B	177.2	604.503
KONEKSI UMUM YT-BROWSING DOWNLOAD	TOTAL DOWNLOAD			100M	15.3 Mbps	0 B	214.1	188.12
KONEKSI KHUSUS YT	KONEKSI UMUM YT-BROWSING DOWNLOAD	PAKET-YT-DOWNLOAD	50M	50M	0bps	0 B	205.9	181.31
UMUM DOWNLOAD	KONEKSI UMUM YT-BROWSING DOWNLOAD	PAKET-UMUM-DOWNLOAD			15.3 kbps	0 B	73.0	6.905.1
ZOOM DOWNLOAD	TOTAL DOWNLOAD	PAKET-ZOOM-DOWNLOAD			41.6 kbps	0 B	4188.9	7.702.0
TOTAL UPLOAD	GLOBAL TRAFFIC			100M	4.1 Mbps	0 B	13.0	646.76.476
DNS UPLOAD	TOTAL UPLOAD	Paket-DNS-Upload			0bps	0 B	200.9	3.184.9
GMEET UPLOAD	TOTAL UPLOAD	PAKET-GMEET-UPLOAD			0bps	0 B	446.7	8.942
ICMP UPLOAD	TOTAL UPLOAD	Paket-ICMP-Upload			8.3 kbps	0 B	145.3	687.354
KONEKSI UMUM YT-BROWSING UPLOAD	TOTAL UPLOAD			100M	3.9 Mbps	0 B	10.1	646.67.762
KONEKSI KHUSUS YT UPLOAD	KONEKSI UMUM YT-BROWSING UPLOAD	PAKET-YT-UPLOAD	49M	50M	0bps	0 B	9.1	646.63.987
UMUM UPLOAD	KONEKSI UMUM YT-BROWSING UPLOAD	PAKET-UMUM-UPLOAD			3.9 Mbps	0 B	1100.2	2.774.6
ZOOM UPLOAD	TOTAL UPLOAD	PAKET-ZOOM-UPLOAD			231.9 kbps	0 B	2634.9	4.632.2

Gambar 4.66 Hasil konfigurasi *Queue tree* mikrotik

4.7 KONFIGURASI SIMPLE QUEUE PADA MIKROTIK

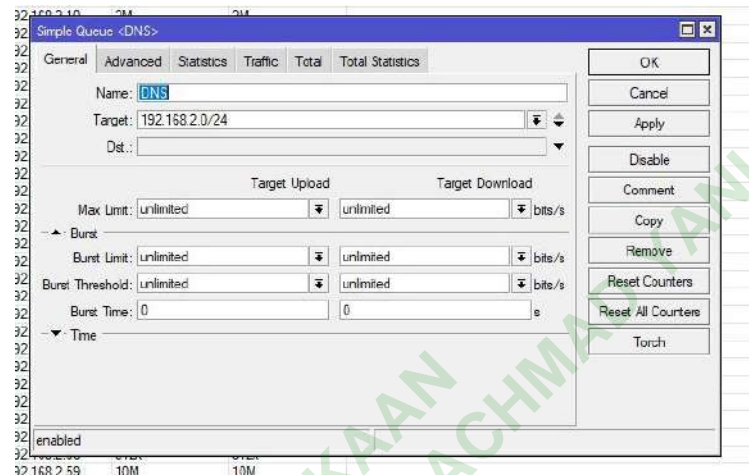
Setelah konfigurasi *Queue tree* berhasil, langkah selanjutnya yaitu konfigurasi *Simple queue* untuk limitasi dan penerapan ke masing masing segmen jaringan, berikut langkah langkah yang dilakukan :

1. Pada menu *queues*, *Simple queue* digunakan untuk menambahkan konfigurasi baru. Pada tab *general*, kolom *name* adalah TOTAL BANDWIDTH, kolom target adalah alamat *ether2-LAN* di switch mikrotik, yaitu 192.168.2.0/24. Kolom target *upload max limit* adalah 100M, dan target *download max limit* adalah 100M (d disesuaikan dengan koneksi internet yang didapat dari *ISP*). *Apply* dan ok, seperti yang terlihat pada gambar 4.67 berikut.



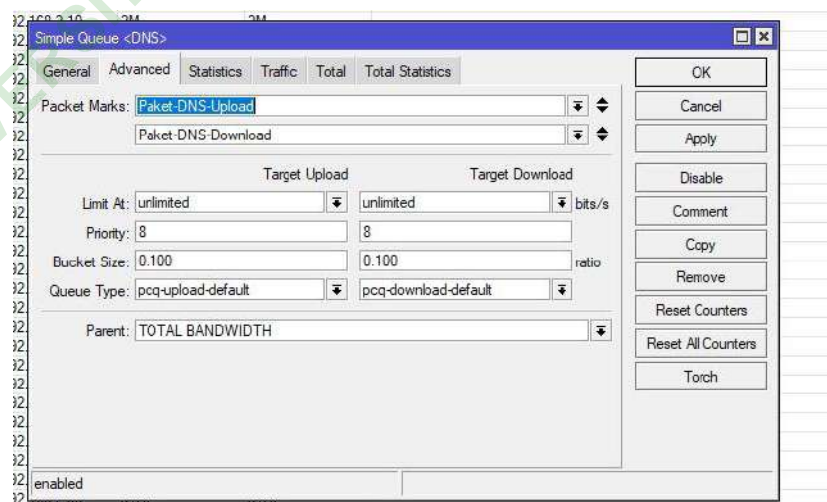
Gambar 4.67 Parent Simple queue general-total bandwidth

2. Pada menu *queues*, *Simple queue* digunakan untuk menambahkan konfigurasi baru untuk *DNS*. Pada tab *general*, kolom *name* adalah *DNS*, kolom target adalah *192.168.2.0/24*. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.68 berikut.



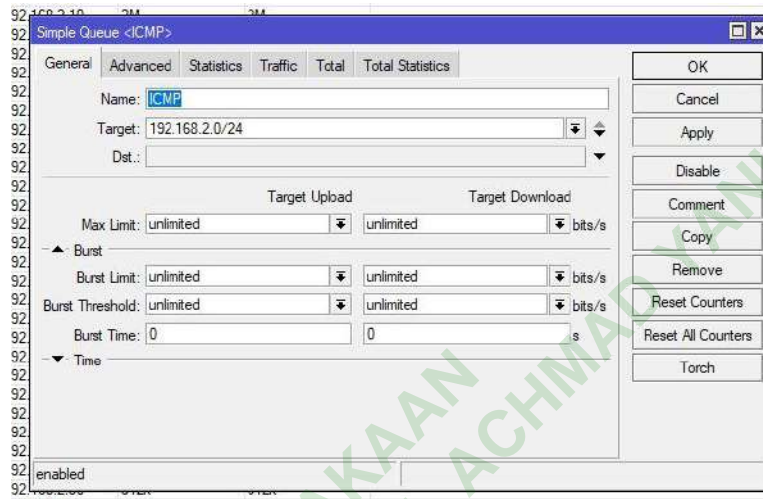
Gambar 4.68 Simple queue general-DNS

3. Pada tab *advanced*, kolom *Packet Mark* adalah *Packet-DNS-Upload* dan *Packet-DNS-Download*. Kolom *queue type* adalah *pcq-upload-default* dan *pcq-download-default*. Kolom *parent* adalah *TOTAL BANDWIDTH*. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.69 berikut.



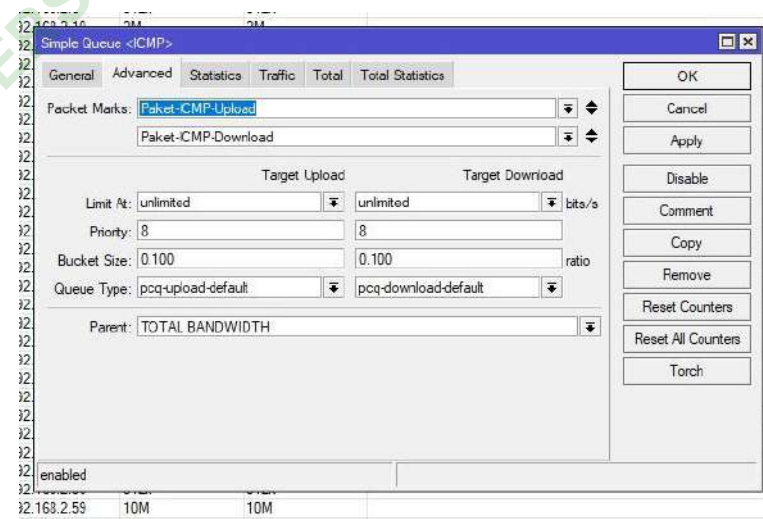
Gambar 4.69 Simple queue advanced-DNS

4. Pada menu *queues*, *Simple queue* digunakan untuk menambahkan konfigurasi baru untuk *ICMP*. Pada tab *general*, kolom *name* adalah *ICMP*, kolom target adalah 192.168.2.0/24. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.70 berikut.



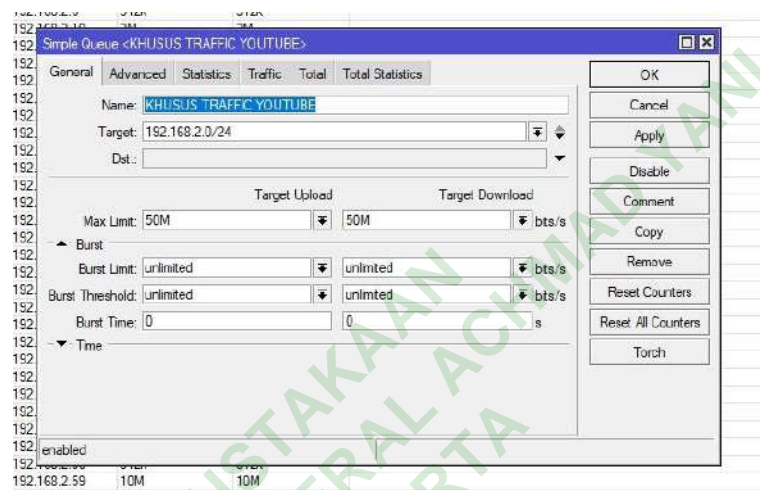
Gambar 4.70 Simple queue general-ICMP

5. Pada tab *advanced*, kolom *Packet Mark* mencakup *Packet-ICMP-Upload* dan *Packet-ICMP-Download*. *Queue type* adalah *pcq-upload-default* dan *pcq-download-default*. Kolom *parent* adalah *TOTAL BANDWIDTH*. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.71 berikut.



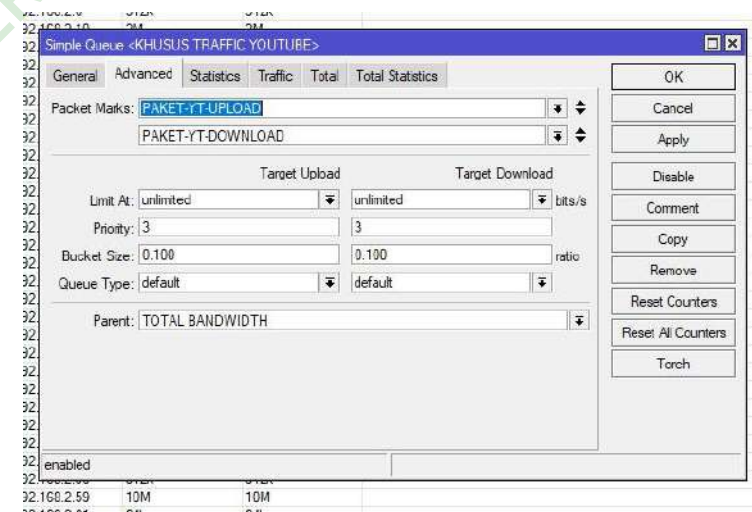
Gambar 4.71 Simple queue advanced-ICMP

6. Pada menu *queues*, *Simple queue* digunakan untuk menambahkan konfigurasi baru untuk *traffic* YouTube. Pada tab *general*, kolom *name* adalah *KHUSUS TRAFFIC YOUTUBE*, kolom *target* adalah *192.168.2.0/24*, dengan *target upload* dan *download* masing-masing diatur ke *50M* (d disesuaikan dengan *bandwidth* masing-masing). *Apply* dan *ok*, seperti yang terlihat pada gambar 4.72 berikut.



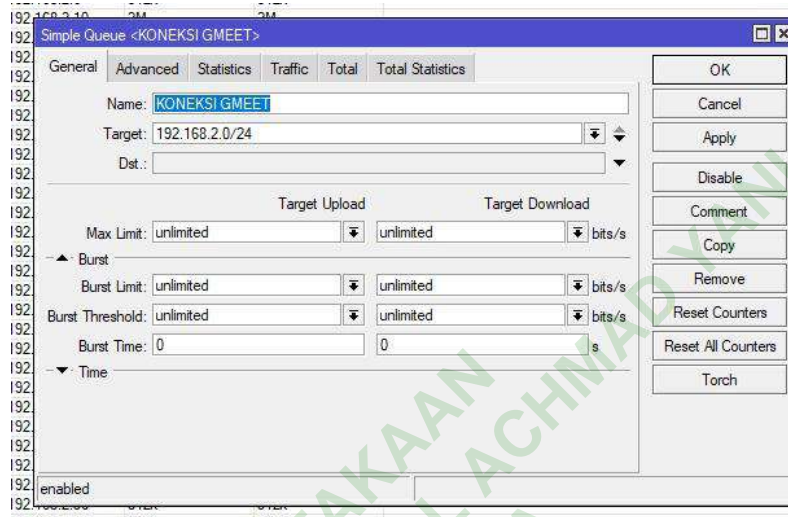
Gambar 4.72 Simple queue general-youtube

7. Pada tab *advanced*, kolom *Packet Mark* mencakup *PAKET-YT-UPLOAD* dan *PAKET-YT-DOWNLOAD*. Kolom *priority* diatur ke nilai *3*, kolom *queue type* adalah *default*, dan kolom *parent* adalah *TOTAL BANDWIDTH*, seperti yang terlihat pada gambar 4.73 berikut.



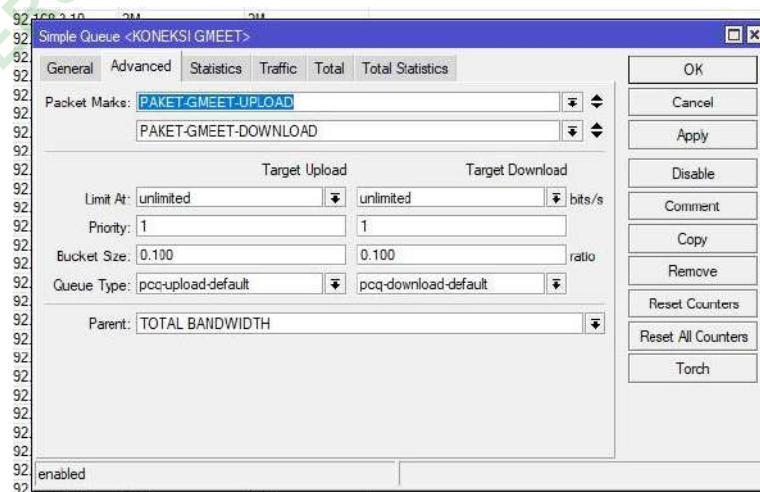
Gambar 4.73 Simple queue advanced-youtube

8. Pada menu *queues*, *Simple queue* digunakan untuk menambahkan konfigurasi baru untuk Google Meet. Pada tab *general*, kolom *name* adalah KONEKSI GMEET, kolom target adalah 192.168.2.0/24. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.74 berikut.



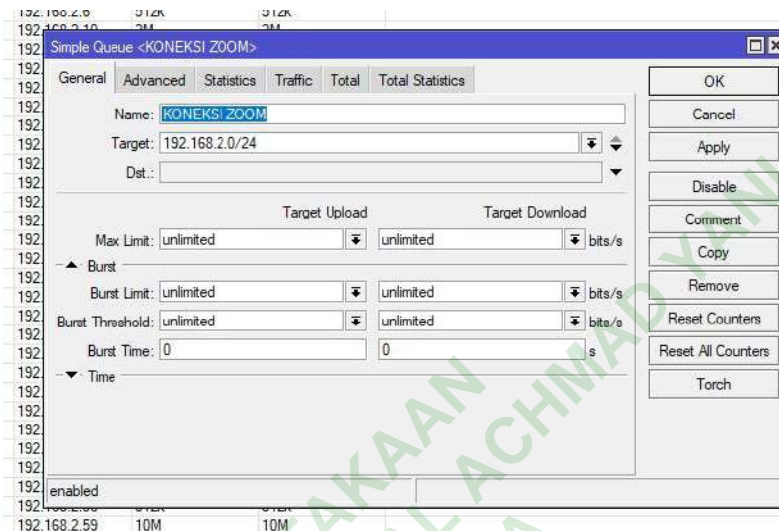
Gambar 4.74 Simple queue general-google meet

9. Pada tab *advanced*, kolom *Packet Mark* mencakup PAKET-GMEET-UPLOAD dan PAKET-GMEET-DOWNLOAD. Kolom *priority* diatur ke nilai 1, *queue type* adalah *pcq-upload-default* dan *pcq-download-default*, serta kolom *parent* adalah TOTAL BANDWIDTH, seperti yang terlihat pada gambar 4.75 berikut.



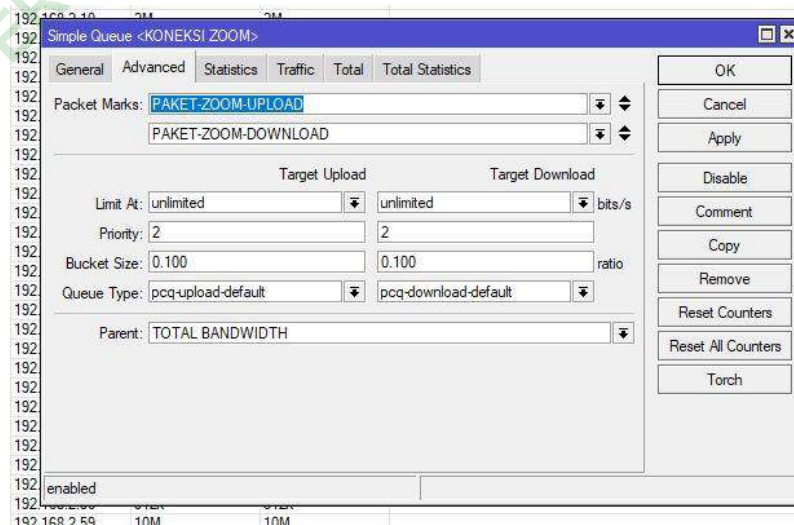
Gambar 4.75 Simple queue advanced-google meet

10. Pada menu *queues*, *Simple queue* digunakan untuk menambahkan konfigurasi baru untuk Zoom Meet. Pada tab *general*, kolom *name* adalah KONEKSI ZOOM, kolom target adalah 192.168.2.0/24. *Apply* dan *ok*, seperti yang terlihat pada gambar 4.76 berikut.



Gambar 4.76 Simple queue general-zoom meet

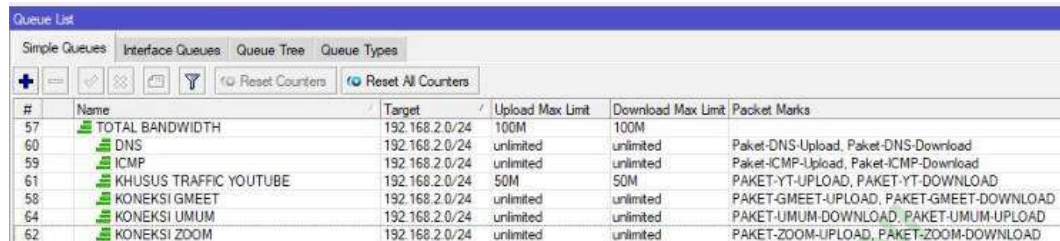
11. Pada tab *advanced*, kolom *Packet Mark* mencakup *PAKET-ZOOM-UPLOAD* dan *PAKET-ZOOM-DOWNLOAD*. Kolom *priority* diatur ke nilai 2, *queue type* adalah *pcq-upload-default* dan *pcq-download-default*, serta kolom *parent* adalah *TOTAL BANDWIDTH*, seperti yang terlihat pada gambar 4.77 berikut.



Gambar 4.77 Simple queue advanced-zoom meet

4.7.1 Hasil Konfigurasi *Simple queue*

Berikut adalah hasil konfigurasi *Simple queue* mikrotik yang telah berhasil di buat, seperti pada gambar 4.78 berikut.



#	Name	Target	Upload Max Limit	Download Max Limit	Packet Marks
57	TOTAL BANDWIDTH	192.168.2.0/24	100M	100M	
60	DNS	192.168.2.0/24	unlimited	unlimited	Paket-DNS-Upload, Paket-DNS-Download
59	ICMP	192.168.2.0/24	unlimited	unlimited	Paket-ICMP-Upload, Paket-ICMP-Download
61	KHUSUS TRAFFIC YOUTUBE	192.168.2.0/24	50M	50M	PAKET-YT-UPLOAD, PAKET-YT-DOWNLOAD
58	KONEKSI GMEET	192.168.2.0/24	unlimited	unlimited	PAKET-GMEET-UPLOAD, PAKET-GMEET-DOWNLOAD
64	KONEKSI UMUM	192.168.2.0/24	unlimited	unlimited	PAKET-UMUM-DOWNLOAD, PAKET-UMUM-UPLOAD
62	KONEKSI ZOOM	192.168.2.0/24	unlimited	unlimited	PAKET-ZOOM-UPLOAD, PAKET-ZOOM-DOWNLOAD

Gambar 4.78 Hasil konfigurasi *Simple queue* mikrotik

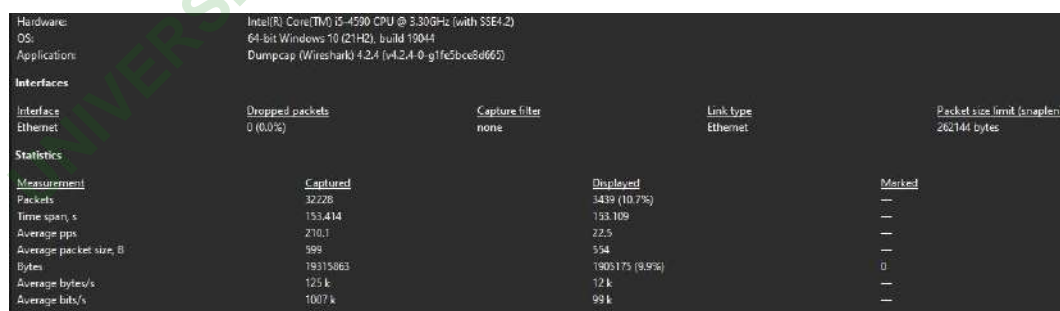
4.8 HASIL PENGUJIAN SEBELUM KONFIGURASI *MANGLE* DAN *QUEUE TREE*

Berikut adalah hasil pengujian *Quality of Service (QOS)* di Solo Technopark sebelum menggunakan konfigurasi *mangle* dan *Queue tree*. Pengujian ini dilakukan secara bertahap 5 kali mulai dari jam 9 pagi sampai jam 5 sore dengan cara membuka youtube (khususnya *live streaming*). Pengujian ini dilakukan dengan menggunakan sampel komputer bagian keuangan dan logistik..

4.8.1 Hasil Pengujian Komputer Keuangan Sebelum Konfigurasi

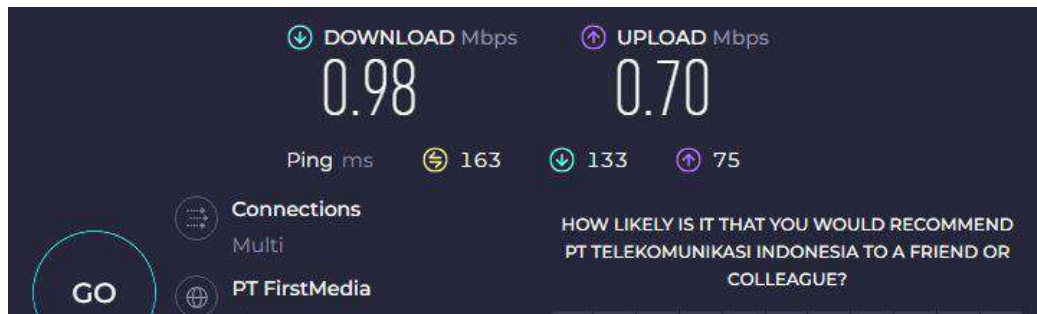
Berikut adalah hasil pengujian komputer bagian divisi keuangan Solo Technopark

1. *Test youtube streaming* sebelum konfigurasi (jam 9 pagi)



Measurement	Captured	Displayed	Marked
Packets	32228	3439 (10.7%)	—
Time span, s	153.414	133.109	—
Average pps	210.1	22.5	—
Average packet size, B	599	554	—
Bytes	19315863	1905175 (9.9%)	0
Average bytes/s	125 k	12 k	—
Average bits/s	1007 k	99 k	—

Gambar 4.79 *Capture file* wireshark (youtube 1)



Gambar 4.80 Speedtest koneksi sebelum konfigurasi (jam 9 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{1905175}{153.109} = 12 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{179}{32228} \times 100 = 0.6\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{153.109}{3439} \times 1000 = 44.5 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

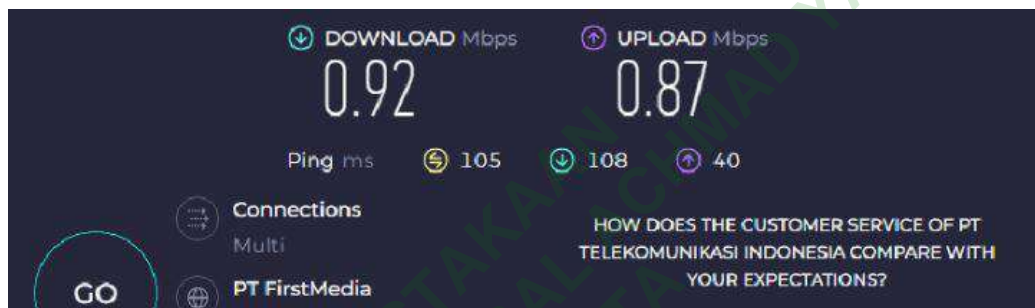
Maka *jitter* yang didapat adalah $\frac{153.109}{3439-1} \times 1000 = 43.5 \text{ ms}$

2. Test youtube streaming sebelum konfigurasi (jam 11 pagi)

Interface	Dropped packets	Capture filter	Link type	Packet size limit (optional)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes

Measurement	Captured	Displayed	Marked
Packets	48689	7124 (14.6%)	—
Time span, s	224.895	224.895	—
Average pps	215.5	31.7	—
Average packet size, B	615	670	—
Bytes	29950138	4769572 (15.9%)	0
Average bytes/s	132 k	21 k	—
Average bits/s	1060 k	169 k	—

Gambar 4.81 Capture file wireshark (youtube 2)



Gambar 4.82 Speedtest koneksi sebelum konfigurasi (jam 11 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{4769572}{224.895} = 21 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet \ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{359}{48689} \times 100 = 0.7\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

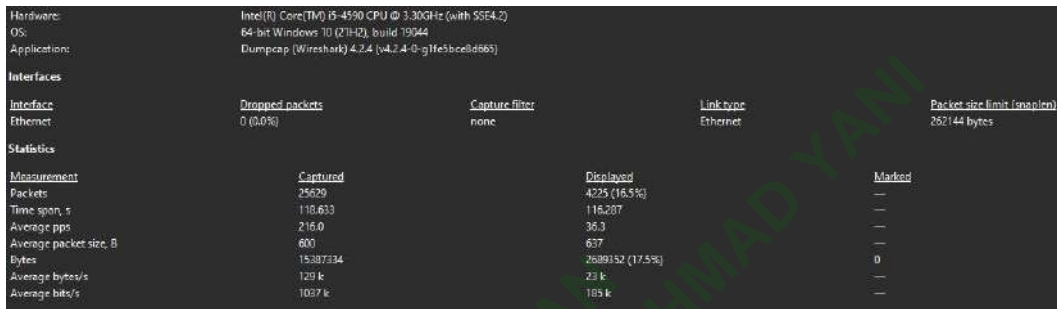
Maka *delay* yang didapat adalah $\frac{224.895}{7124} \times 1000 = 31.5 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima}-1} \times 1000$$

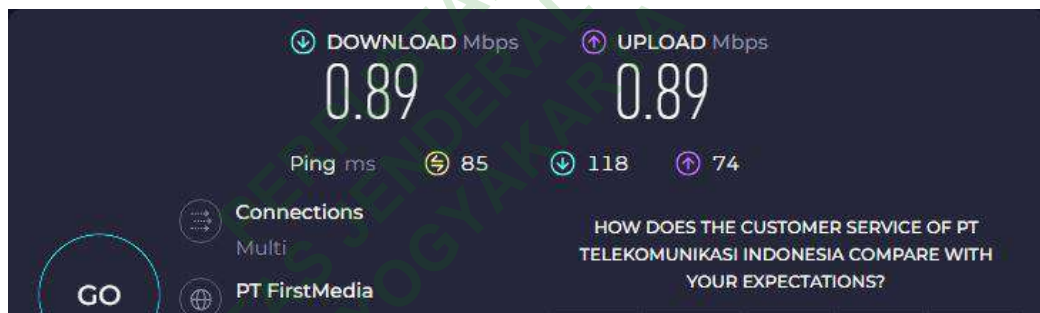
Maka *jitter* yang didapat adalah $\frac{224.895}{7124-1} \times 1000 = 30.5 \text{ ms}$

3. Test youtube streaming sebelum konfigurasi (jam 1 siang)



Measurement	Captured	Displayed	Marked
Packets	25629	4225 (16.5%)	—
Time span, s	116.633	116.287	—
Average pps	216.0	36.3	—
Average packet size, B	600	637	—
Bytes	15387334	2689352 (17.5%)	0
Average bytes/s	129 k	23 k	—
Average bits/s	1037 k	185 k	—

Gambar 4.83 Capture file wireshark (youtube 3)



Gambar 4.84 Speedtest koneksi sebelum konfigurasi (jam 1 siang)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{2689352}{116.287} = 23 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet \text{ loss} = \frac{(\text{paket dikirim}-\text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{191}{25629} \times 100 = 0.7\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{116.287}{4225} \times 1000 = 27.5 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{116.287}{4225 - 1} \times 1000 = 26.5 \text{ ms}$

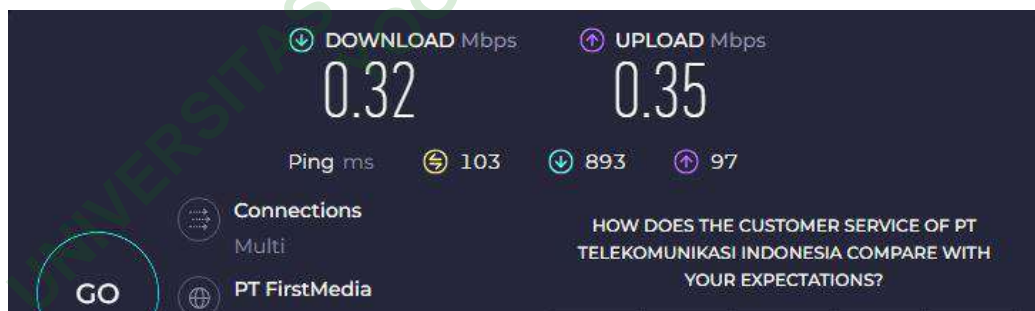
4. Test youtube streaming sebelum konfigurasi (jam 3 sore)



Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes

Measurement	Captured	Displayed	Marked
Packets	39370	2895 (7.4%)	—
Time span, s	189.286	184.695	—
Average pps	208.0	15.7	—
Average packet size, B	588	564	—
Bytes	23131517	1632082 (7.1%)	0
Average bytes/s	122 k	8836	—
Average bits/s	977 k	70 k	—

Gambar 4.85 Capture file wireshark (youtube 4)



Gambar 4.86 Speedtest koneksi sebelum konfigurasi (jam 3 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{1632082}{184.695} = 8836 \text{ (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{128}{39370} \times 100 = 0.3\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{184.695}{2895} \times 1000 = 63.7\ ms$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

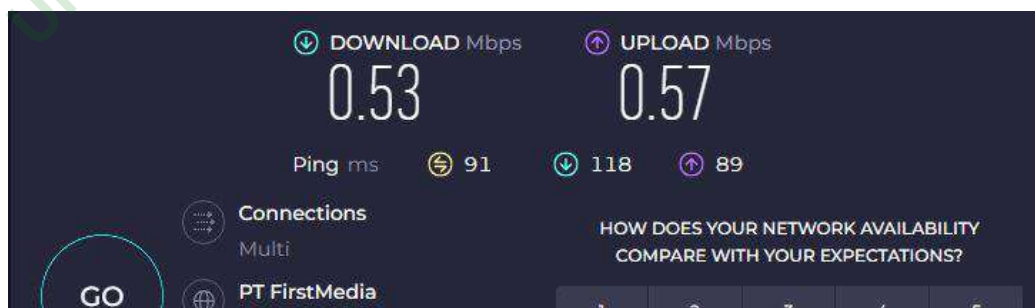
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{184.695}{2895 - 1} \times 1000 = 62.7\ ms$

5. Test youtube streaming sebelum konfigurasi (jam 5 sore)

Hardware	Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz (with SSE4.2)		
OS:	64-bit Windows 10 (21H2), build 19044		
Applications:	Dumpcap (Wireshark) 4.2.4 (v4.2.4-0-g1fe3bce6d665)		
Interfaces			
Interface	Dropped packets	Capture filter	Link type
Ethernet	0 (0.0%)	none	Ethernet
Statistics			
Measurement			
Measurement	Captured	Displayed	Marked
Packets	28477	554 (1.9%)	—
Time spent, s	120.888	124.113	—
Average pps	224.4	4.5	—
Average packet size, B	598	291	—
Bytes	17033537	161405 (0.9%)	0
Average bytes/s	134 k	1300	—
Average bits/s	1073 k	10 k	—

Gambar 4.87 Capture file wireshark (youtube 5)



Gambar 4.88 Speedtest koneksi sebelum konfigurasi (jam 5 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{161405}{124.113} = 1300$ (*bytes/s*)

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{11}{28477} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{124.113}{554} \times 1000 = 224$ *ms*

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{124.113}{554-1} \times 1000 = 223$ *ms*

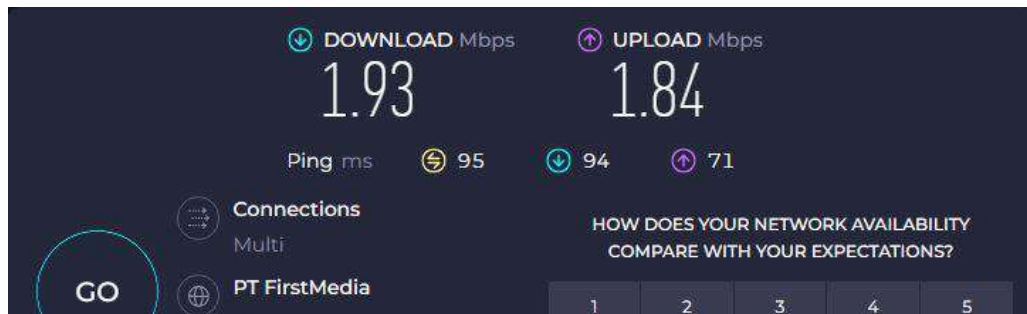
4.8.2 Hasil Pengujian Komputer Logistik Sebelum Konfigurasi

Berikut adalah hasil pengujian komputer bagian divisi logistik Solo Technopark

1. *Test youtube streaming* sebelum konfigurasi (jam 9 pagi)

Hardware:	Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE42)		
OS:	64-bit Windows 10 (22H2), build 19045		
Application:	Dumpcap (Wireshark) 4.2.5 (v4.2.5-0-g4aa814ac25a1)		
Interfaces			
Interface	Drop/cap packets	Capture filter	Link type
Ethernet	0 (0.0%)	none	Ethernet
			Packet size limit (maxlen): 262144 bytes
Statistics			
Measurement	Captured	Displayed	Marked
Packets	34177	5948 (17.4%)	—
Time span, s	151.582	148.001	—
Average pps	225.5	40.2	—
Average packet size, B	720	864	—
Bytes	24606265	5140240 (20.9%)	0
Average bytes/s	162 k	34 k	—
Average bits/s	1290 k	277 k	—

Gambar 4.89 *Capture file* wireshark (youtube 1)



Gambar 4.90 Speedtest koneksi sebelum konfigurasi (jam 9 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{5140240}{148.001} = 34 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet \ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{290}{34177} \times 100 = 0.8\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{148.001}{5948} \times 1000 = 24.8 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

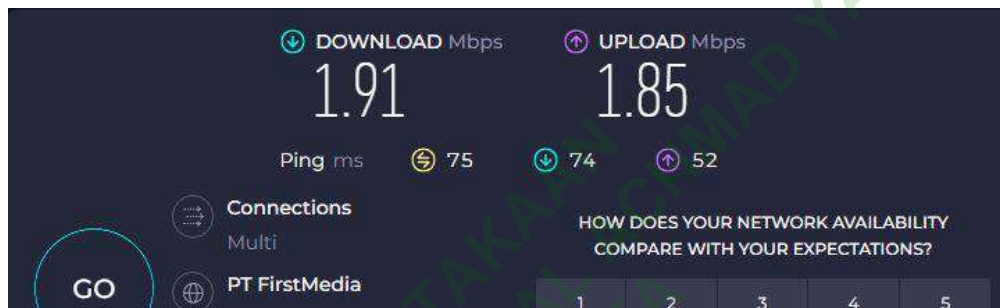
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{148.001}{5948-1} \times 1000 = 23.8 \text{ ms}$

2. Test youtube streaming sebelum konfigurasi (jam 11 pagi)

Hardware		Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE4.2)		
OS		64-bit Windows 10 (22H2, build 19045)		
Application		Dumpcap (Wireshark) 4.2.5 (v4.2.5-0-g1ae814ac25a1)		
Interfaces				
Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	252144 bytes
Statistics				
Measurement	Captured	Discarded	Marked	
Packets	39884	8396 (21.1%)	—	
Time span, s	151.674	141.606	—	
Average pps	262.6	59.3	—	
Average packet size, B	769	896	—	
Bytes	30678535	7522390 (24.5%)	0	
Average bytes/s	201 k	53 k	—	
Average bits/s	1615 k	424 k	—	

Gambar 4.91 Capture file wireshark (youtube 2)



Gambar 4.92 Speedtest koneksi sebelum konfigurasi (jam 11 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{7522390}{141.606} = 53 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{315}{39884} \times 100 = 0.8\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{141.606}{8396} \times 1000 = 16.8 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

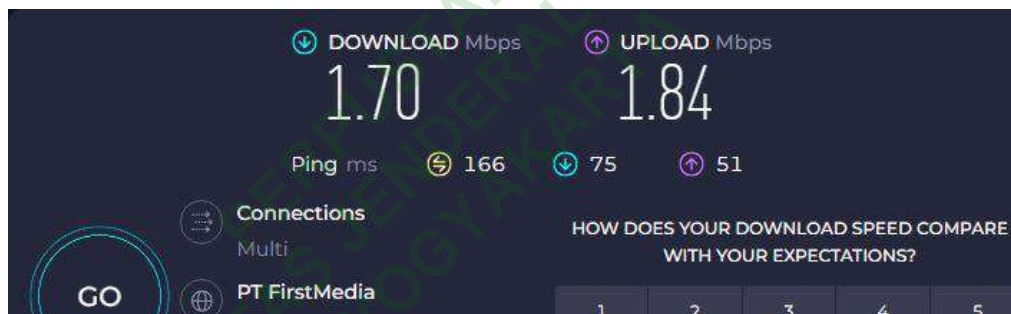
$$jitter = \frac{\text{total variasi } delay}{\text{paket data diterima}-1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{141.606}{8396-1} \times 1000 = 15.8 \text{ ms}$

3. Test youtube streaming sebelum konfigurasi (jam 1 siang)

Hardware	Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE4.2)		
OS	64-bit Windows 10 (22H2), build 19045		
Application	Dumpcap (Wireshark) 4.2.5 (v4.2.5-0-g4aa814ac25a1)		
Interfaces			
Interface	Dropped packets	Capture filter	Link type
Ethernet	0 (0.0%)	none	Ethernet
			Packet size limit (snaplen)
			262144 bytes
Statistics			
Measurement	Captured	Displayed	Marked
Packets	33294	5324 (16.0%)	—
Time span, s	131.739	127.127	—
Average pps	252.7	41.9	—
Average packet size, B	731	773	—
Bytes	24322387	4115141 (16.9%)	0
Average bytes/s	184 k	32 k	—
Average bits/s	1476 k	258 k	—

Gambar 4.93 Capture file wireshark (youtube 3)



Gambar 4.94 Speedtest koneksi sebelum konfigurasi (jam 1 siang)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{4115141}{127.127} = 32 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet \text{ loss} = \frac{(\text{paket dikirim}-\text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{197}{33294} \times 100 = 0.6\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{127.127}{5324} \times 1000 = 23.8 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

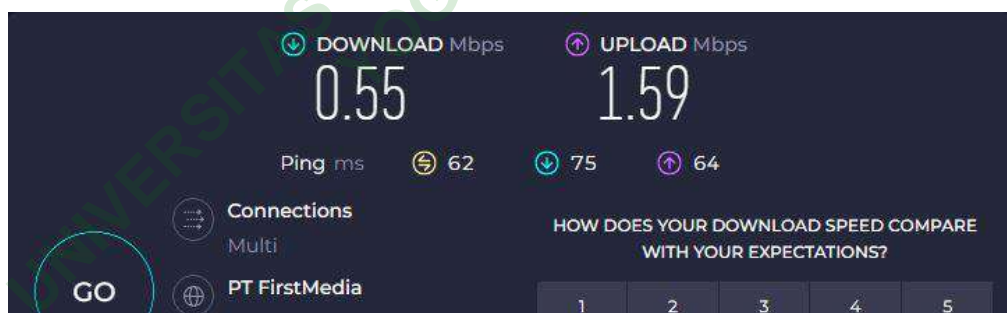
Maka *jitter* yang didapat adalah $\frac{127.127}{5324 - 1} \times 1000 = 22.8 \text{ ms}$

4. Test youtube streaming sebelum konfigurasi (jam 3 sore)

Interface	Dropped packets	Capture filter	Link type	Packet size limit (capture)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes

Measurement	Captured	Displayed	Marked
Packets	41676	8391 (20.0%)	—
Time span, s	168.228	163.173	—
Average pps	247.7	51.2	—
Average packet size, B	750	856	—
Bytes	31297222	7147826 (22.9%)	0
Average bytes/s	185 k	43 k	—
Average bits/s	1485 k	350 k	—

Gambar 4.95 Capture file wireshark (youtube 4)



Gambar 4.96 Speedtest koneksi sebelum konfigurasi (jam 3 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{7147826}{163.173} = 43k \text{ (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{324}{41676} \times 100 = 0.8\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{163.173}{8351} \times 1000 = 19.5\ ms$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

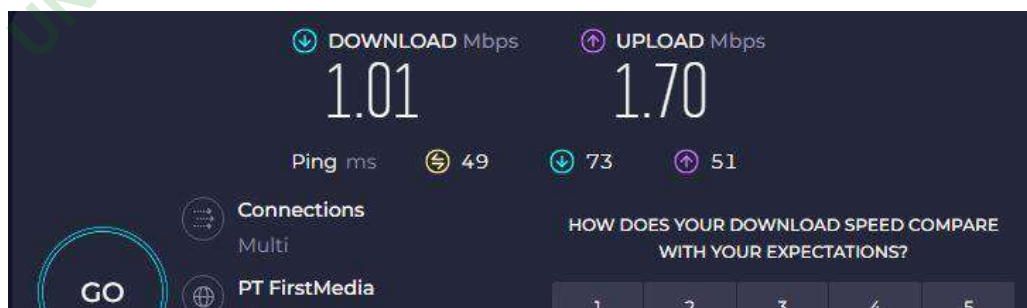
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{163.173}{8351 - 1} \times 1000 = 18.5\ ms$

5. Test youtube streaming sebelum konfigurasi (jam 5 sore)

Hardware:		Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE4.2)	
OS:		64-bit Windows 70 (22H2), build 19045	
Application:		Dumpcap (Wireshark) 4.2.5 (v4.2.5-0-g1ea814ec25a1)	
Interfaces			
Interface	Dropped packets	Capture filter	Link type
Ethernet	0 (0.0%)	none	Ethernet
Packet size limit (capture): 262144 bytes			
Statistics			
Measurement	Captured	Displayed	Marked
Packets	46985	2352 (5.0%)	—
Time span, s	144.870	135.156	—
Average pps	321.6	18.9	—
Average packet size, B	767	761	—
Bytes	35710256	1941142 (5.4%)	0
Average bytes/s	246 k	14 k	—
Average bits/s	1972 k	114 k	—

Gambar 4.97 Capture file wireshark (youtube 5)



Gambar 4.98 Speedtest koneksi sebelum konfigurasi (jam 5 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{1941142}{135.156} = 14k \text{ (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{132}{46585} \times 100 = 0.3\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{135.156}{2552} \times 1000 = 52.9 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{135.156}{2552-1} \times 1000 = 51.9 \text{ ms}$

4.9 HASIL PENGUJIAN SETELAH KONFIGURASI MANGLE DAN QUEUE TREE

Berikut adalah hasil pengujian *Quality of Service (QOS)* di Solo Technopark setelah menggunakan konfigurasi *mangle* dan *Queue tree*. Pengujian ini dilakukan secara bertahap 5 kali mulai dari jam 9 pagi sampai jam 5 sore dengan cara membuka youtube (khususnya *live streaming*). Pengujian ini dilakukan dengan menggunakan sampel komputer bagian keuangan dan logistik..

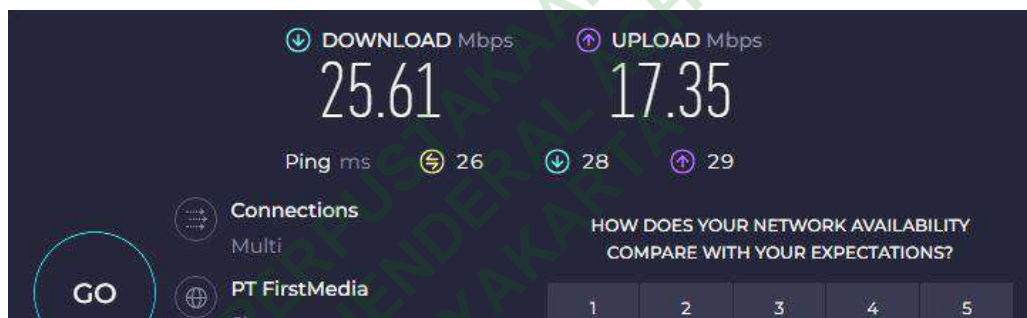
4.9.1 Hasil Pengujian Komputer Keuangan Setelah Konfigurasi

Berikut adalah hasil pengujian komputer bagian divisi keuangan Solo Technopark

1. *Test youtube streaming* setelah konfigurasi (jam 9 pagi)

Measurement	Captured	Displayed	Marked
Packets	17798	17107 (96.1%)	—
Time span, s	86.637	85.608	—
Average pps	205.4	199.8	—
Average packet size, B	1044	1083	—
Bytes	18572784	18531449 (99.8%)	0
Average bytes/s	214 k	216 k	—
Average bits/s	1715 k	1731 k	—

Gambar 4.99 Capture file wireshark (youtube 1)



Gambar 4.100 Speedtest koneksi setelah konfigurasi (jam 9 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{18531449}{85.608} = 216 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{2}{17798} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

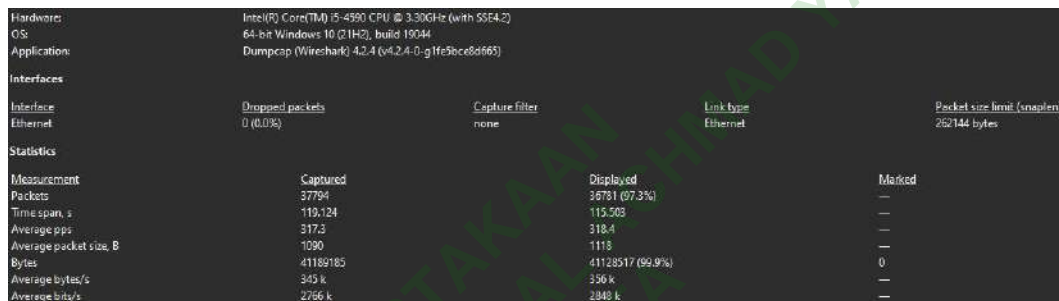
Maka *delay* yang didapat adalah $\frac{85.608}{17107} \times 1000 = 5.0 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{85.608}{17107-1} \times 1000 = 4.0 \text{ ms}$

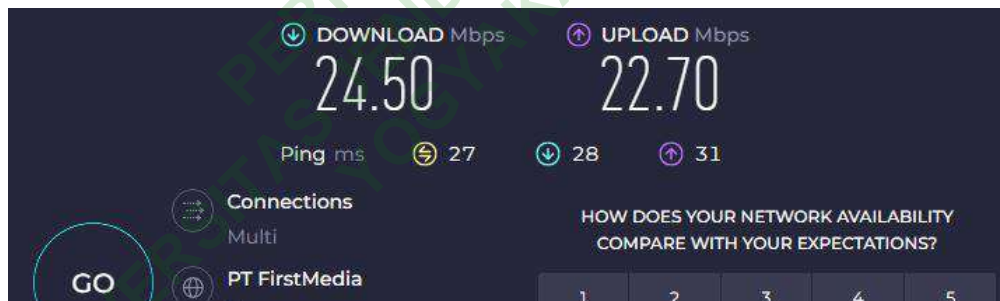
2. Test youtube streaming setelah konfigurasi (jam 11 pagi)



Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes

Measurement	Captured	Displayed	Masked
Packets	37794	36781 (97.3%)	—
Time span, s	118.124	115.503	—
Average pps	317.3	318.4	—
Average packet size, B	1090	1113	—
Bytes	41189185	41128517 (99.9%)	0
Average bytes/s	343 k	336 k	—
Average bits/s	2766 k	2348 k	—

Gambar 4.101 Capture file wireshark (youtube 2)



Gambar 4.102 Speedtest koneksi setelah konfigurasi (jam 11 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{41128517}{115.503} = 356 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet \text{ loss} = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{2}{37794} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{115.503}{36781} \times 1000 = 3.1 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

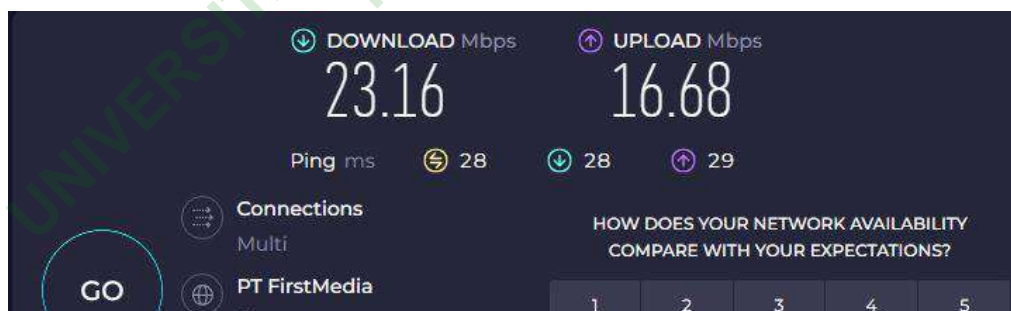
Maka *jitter* yang didapat adalah $\frac{115.503}{36781-1} \times 1000 = 2.1 \text{ ms}$

3. *Test youtube streaming* setelah konfigurasi (jam 1 siang)

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes

Measurement	Captured	Displayed	Marked
Packets	46577	45970 (98.7%)	—
Time span, s	81.868	77.003	—
Average pps	568.0	597.0	—
Average packet size, B	1106	1119	—
Bytes	51494949	51452458 (99.9%)	0
Average bytes/s	629 k	668 k	—
Average bits/s	5032 k	5345 k	—

Gambar 4.103 Capture file wireshark (youtube 3)



Gambar 4.104 Speedtest koneksi setelah konfigurasi (jam 1 siang)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{51452458}{77.003} = 668 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{1}{46577} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{77.003}{45970} \times 1000 = 1.6\ ms$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

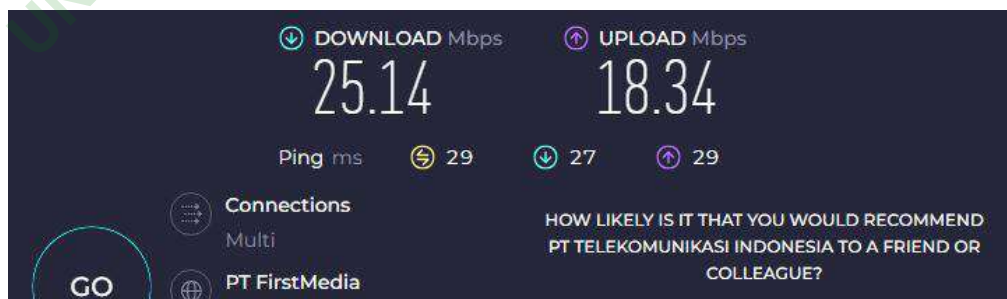
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{77.003}{45970 - 1} \times 1000 = 0.6\ ms$

4. Test youtube streaming setelah konfigurasi (jam 3 sore)

Hardware:		Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz (with SSE4.2)		
OS:		64-bit Windows 10 (21H2), build 19044		
Application:		Dumpcap (Wireshark) 4.2.4 (v4.2.4-0-g1e5ace1d865)		
Interfaces				
Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes
Statistics				
Measurement	Captured	Displayed	Marked	
Packets	52827	52089 (98.6%)	—	
Time span, s	64.355	63.899	—	
Average pps	826.2	820.9	—	
Average packet size, B	1122	1137	—	
Bytes	59271527	59227050 (99.9%)	0	
Average bytes/s	702 k	705 k	—	
Average bits/s	5621 k	5647 k	—	

Gambar 4.105 Capture file wireshark (youtube 4)



Gambar 4.106 Speedtest koneksi setelah konfigurasi (jam 3 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{59227050}{83.899} = 705k \text{ (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{0}{52827} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{83.899}{52089} \times 1000 = 1.6 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

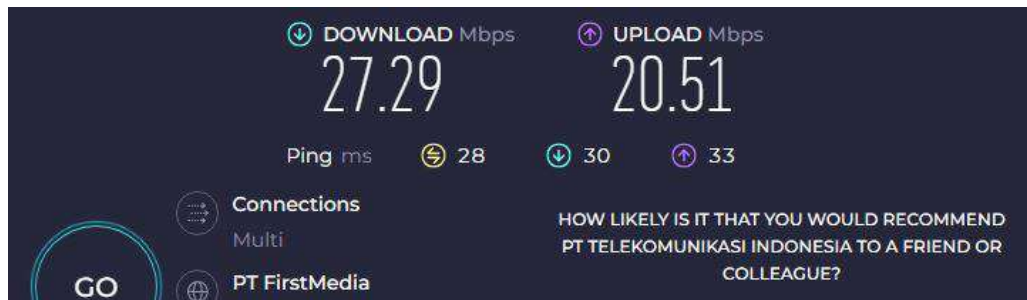
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{83.899}{52089-1} \times 1000 = 0.6 \text{ ms}$

5. Test youtube streaming setelah konfigurasi (jam 5 sore)

Hardware	Intel(R) Core(TM) i5-4590 CPU @ 3.50GHz (with SSE4.2)		
OS	64-bit Windows 10 (21H2), build 19044		
Applications	Dump.cap (Wireshark) 4.2.4 (v4.2.4-0-g1fe3bc8d065)		
Interfaces			
Interface	Dropped packets	Capture filter	Link type
Ethernet	0 (0.0%)	none	Ethernet
			Packet size limit (snaplen): 262144 bytes
Statistics			
Measurement	Counted	Discarded	Marked
Packets	8735	8073 (92.4%)	—
Time span, s	77.532	76.575	—
Average pps	112.7	105.4	—
Average packet size, B	803	864	—
Bytes	7016537	6978882 (99.4%)	0
Average bytes/s	90 k	91 k	—
Average bits/s	723 k	728 k	—

Gambar 4.107 Capture file wireshark (youtube 5)



Gambar 4.108 Speedtest koneksi setelah konfigurasi (jam 5 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{6976882}{76.575} = 91k$ (*bytes/s*)

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{0}{8735} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{76.575}{8073} \times 1000 = 9.4\ ms$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{76.575}{8073 - 1} \times 1000 = 8.4\ ms$

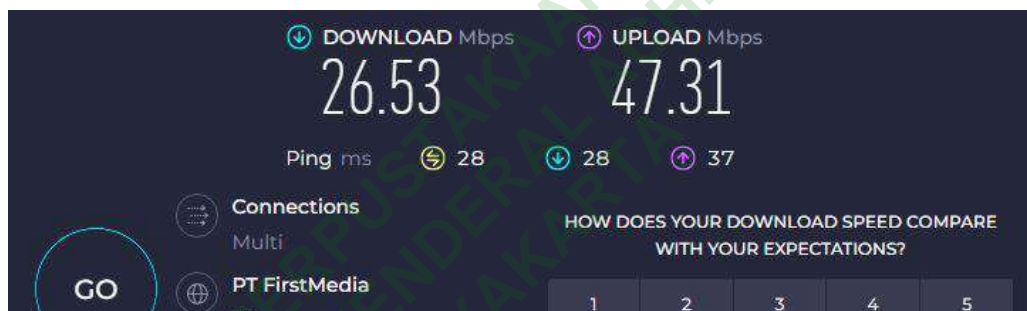
4.9.2 Hasil Pengujian Komputer Logistik Setelah Konfigurasi

Berikut adalah hasil pengujian komputer bagian divisi logistik Solo Technopark

1. Test youtube streaming setelah konfigurasi (jam 9 pagi)

Hardware		Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE4.2)	
OS:		64-bit Windows 10 (22H2), build 19045	
Application:		Dumpcap (Wireshark) 4.2.5 (v4.2.5-0-g4aa814ac25af)	
Interfaces			
Interface	Droped packets	Capture filter	Link type
Ethernet	0 (0.0%)	none	Ethernet
			Packet size limit (capture)
			262144 bytes
Statistics			
Measurement	Captured	Displayed	Marked
Packets	79128	78111 (98.7%)	—
Time span, s	125.350	124.280	—
Average pps	631.3	628.5	—
Average packet size, B	1109	1122	—
Bytes	87729377	87656095 (99.9%)	0
Average bytes/s	699 k	705 k	—
Average bits/s	5598 k	5642 k	—

Gambar 4.109 Capture file wireshark (youtube 1)



Gambar 4.110 Speedtest koneksi setelah konfigurasi (jam 9 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{87656095}{124.280} = 705 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet \text{ loss} = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{0}{79128} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{124.280}{78111} \times 1000 = 1.5 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

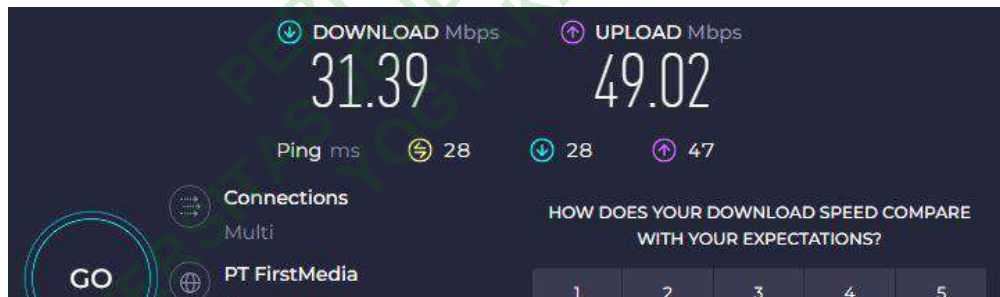
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{124.280}{78111-1} \times 1000 = 0.5 \text{ ms}$

2. Test youtube streaming setelah konfigurasi (jam 11 pagi)

Hardware:		Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE4.2)		
OS:		64-bit Windows 10 (22H2), build 19045		
Application:		Dumpcap (Wireshark) 4.2.5 (4.2.5-0-g4ea814c25e1)		
Interfaces				
Interface	Dropped packets	Capture filter	Link type	Packet size limit (max pLen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes
Statistics				
Measurement		Captured	Displayed	Marked
Packets		42054	40923 (97.33%)	—
Time span, s		122.640	121.980	—
Average pps		342.9	335.5	—
Average packet size, B		1051	1073	—
Bytes		44142249	44144249 (99.9%)	0
Average bytes/s		360 k	361 k	—
Average bits/s		2884 k	2893 k	—

Gambar 4.111 Capture file wireshark (youtube 2)



Gambar 4.112 Speedtest koneksi setelah konfigurasi (jam 11 pagi)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{44144249}{121.980} = 361 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet \text{ loss} = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{0}{42054} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{121.980}{40923} \times 1000 = 2.9 \text{ ms}$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

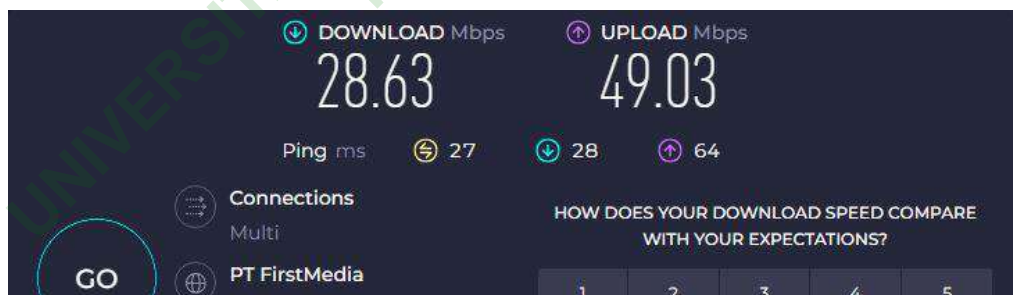
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{121.980}{40923 - 1} \times 1000 = 1.9 \text{ ms}$

3. Test youtube streaming setelah konfigurasi (jam 1 siang)

Hardware:		Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE4.2)		
OS:		64-bit Windows 10 (21H2), build 19045		
Application:		Dumpcap (Wireshark) 4.2.5 (v4.2.5-0-gdaa814ec25a1)		
Interfaces				
Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes
Statistics				
Measurement	Captured	Displayed	Marked	
Packets:	46177	44751 (96.9%)	—	
Time span, s:	169.249	166.276	—	
Average pps:	272.8	265.8	—	
Average packet size, B:	1041	1073	—	
Bytes:	48072438	47985209 (99.8%)	0	
Average bytes/s:	284 k	285 k	—	
Average bits/s:	2272 k	2281 k	—	

Gambar 4.113 Capture file wireshark (youtube 3)



Gambar 4.114 Speedtest koneksi setelah konfigurasi (jam 1 siang)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{47985209}{168.278} = 285 \text{ k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{24}{46177} \times 100 = 0.1\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{168.278}{44731} \times 1000 = 3.7\ ms$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

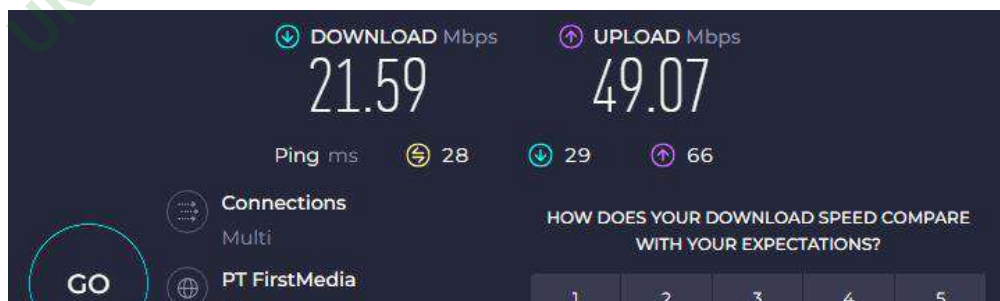
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{168.278}{44731 - 1} \times 1000 = 2.7\ ms$

4. Test youtube streaming setelah konfigurasi (jam 3 sore)

Measurement	Captured	Displayed	Marked
Packets	77780	76033 (97.8%)	—
Time span, s	185.463	185.463	—
Average pps	419.3	410.0	—
Average packet size, B	1069	1692	—
Bytes	83135258	83031421 (99.9%)	0
Average bytes/s	448 k	447 k	—
Average bits/s	3586 k	3581 k	—

Gambar 4.115 Capture file wireshark (youtube 4)



Gambar 4.116 Speedtest koneksi setelah konfigurasi (jam 3 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{83031421}{185.463} = 447\text{k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{2}{77760} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{185.463}{76033} \times 1000 = 2.4\ ms$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

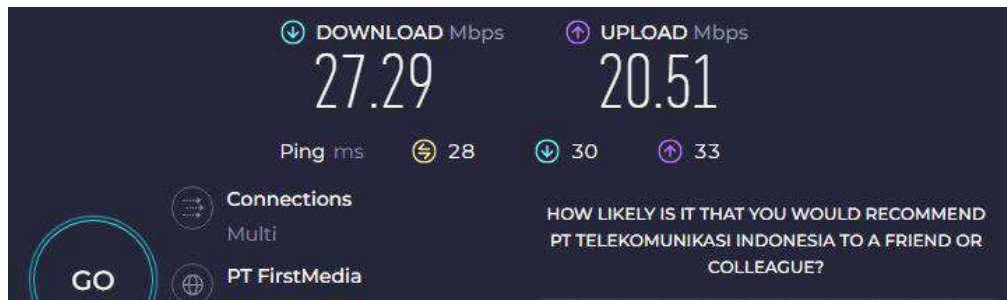
$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{185.463}{76033-1} \times 1000 = 1.4\ ms$

5. Test youtube streaming setelah konfigurasi (jam 5 sore)

Hardware		Intel(R) Core(TM) i5 CPU M 480 @ 2.67GHz (with SSE4.2)		
OS:		64-bit Windows 10 (22H2), build 19045		
Applications		Dumpcap (Wireshark) 4.2.5 (v4.2.5-0-g4aa814a125a1)		
Interfaces				
Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Ethernet	0 (0.0%)	none	Ethernet	262144 bytes
Statistics				
Measurement	Captured	Displayed	Marked	
Packets	34761	33722 (97.1%)	—	
Time span, s	127.896	126.781	—	
Average pps	272.0	265.1	—	
Average packet size, B	1069	1090	—	
Bytes	36832215	36765215 (99.8%)	0	
Average bytes/s	289 k	289 k	—	
Average bits/s	2312 k	2319 k	—	

Gambar 4.117 Capture file wireshark (youtube 5)



Gambar 4.118 Speedtest koneksi setelah konfigurasi (jam 5 sore)

Hasil perhitungan *throughput* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$throughput = \frac{\text{jumlah bytes}}{\text{time span}}$$

Maka *throughput* yang didapat adalah $\frac{36765215}{126.781} = 289\text{k (bytes/s)}$

Hasil perhitungan *packet loss* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan

$$packet\ loss = \frac{(\text{paket dikirim} - \text{paket diterima})}{\text{paket dikirim}} \times 100$$

Maka *packet loss* yang didapat adalah $\frac{0}{34751} \times 100 = 0.0\%$

Hasil perhitungan *delay* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$delay = \frac{\text{total delay}}{\text{paket data diterima}}$$

Maka *delay* yang didapat adalah $\frac{126.781}{33732} \times 1000 = 3.7\ ms$

Hasil perhitungan *jitter* berdasarkan data pada gambar diatas dapat diketahui dan dihitung menggunakan persamaan :

$$jitter = \frac{\text{total variasi delay}}{\text{paket data diterima} - 1} \times 1000$$

Maka *jitter* yang didapat adalah $\frac{126.781}{33732 - 1} \times 1000 = 2.7\ ms$

4.10 PERBANDINGAN HASIL SEBELUM DAN SETELAH PENERAPAN KONFIGURASI

Berikut adalah hasil pengujian dan pengukuran *Quality of Service (QOS)* di Solo Technopark sebelum dan setelah diterapkan konfigurasi. Dalam hal ini dihitung rata-rata dari setiap percobaan baik dari ruang keuangan dan ruang logistik.

4.10.1 Hasil Pengukuran Komputer Keuangan Sebelum Konfigurasi

Berikut adalah hasil pengukuran komputer bagian divisi keuangan Solo Technopark

Tabel 4.1 Pengukuran komputer keuangan sebelum konfigurasi

Percobaan	Throughput (bps)	Delay (ms)	Jitter (ms)	Packet loss (%)
Percobaan 1	12.000	44.5	43.5	0.6
Percobaan 2	21.000	31.5	30.5	0.7
Percobaan 3	23.000	27.5	26.5	0.7
Percobaan 4	8836	63.7	62.7	0.3
Percobaan 5	1300	224	223	0.0
Rata-rata	13.227	78.2	77.2	0.5

4.10.2 Hasil Pengukuran Komputer Logistik Sebelum Konfigurasi

Berikut adalah hasil pengukuran komputer bagian divisi logistik Solo Technopark

Tabel 4.2 Pengukuran komputer logistik sebelum konfigurasi

Percobaan	Throughput (bps)	Delay (ms)	Jitter (ms)	Packet loss (%)
Percobaan 1	34.000	24.8	23.8	0.8
Percobaan 2	53.000	16.8	15.8	0.8
Percobaan 3	32.000	23.8	22.8	0.6
Percobaan 4	43.000	19.5	18.5	0.8
Percobaan 5	14.000	52.9	51.9	0.3
Rata-rata	35.200	27.6	26.6	0.7

4.10.3 Hasil Pengukuran Komputer Keuangan Setelah Konfigurasi

Berikut adalah hasil pengukuran komputer bagian divisi keuangan Solo Technopark

Tabel 4.3 Pengukuran komputer keuangan setelah konfigurasi

Percobaan	<i>Throughput</i> (bps)	<i>Delay</i> (ms)	<i>Jitter</i> (ms)	<i>Packet loss</i> (%)
Percobaan 1	216.000	5.0	4.0	0.0
Percobaan 2	356.000	3.1	2.1	0.0
Percobaan 3	668.000	1.6	0.6	0.0
Percobaan 4	705.000	1.6	0.6	0.0
Percobaan 5	91.000	9.4	8.4	0.0
Rata-rata	407.200	4.1	3.1	0.0

4.10.4 Hasil Pengukuran Komputer Logistik Setelah Konfigurasi

Berikut adalah hasil pengukuran komputer bagian divisi logistik Solo Technopark

Tabel 4.4 Pengukuran komputer logistik setelah konfigurasi

Percobaan	<i>Throughput</i> (bps)	<i>Delay</i> (ms)	<i>Jitter</i> (ms)	<i>Packet loss</i> (%)
Percobaan 1	705.000	1.5	0.5	0.0
Percobaan 2	361.000	2.9	1.9	0.0
Percobaan 3	285.000	3.7	2.7	0.1
Percobaan 4	447.000	2.4	1.4	0.0
Percobaan 5	289.000	3.7	2.7	0.0
Rata-rata	417.400	2.8	1.8	0.0

4.10.5 Hasil Pengukuran Sebelum Dan Setelah Penerapan Konfigurasi

Berikut adalah hasil pengukuran komputer divisi keuangan dan logistik sebelum dan sesudah penerapan kombinasi *Simple queue* dan *Queue tree*. Hasil ini merupakan rekap nilai rata-rata *QoS* baik itu *throughput*, *delay*, *jitter* dan *packet loss*.

Tabel 4.5 Pengukuran sebelum dan setelah penerapan konfigurasi

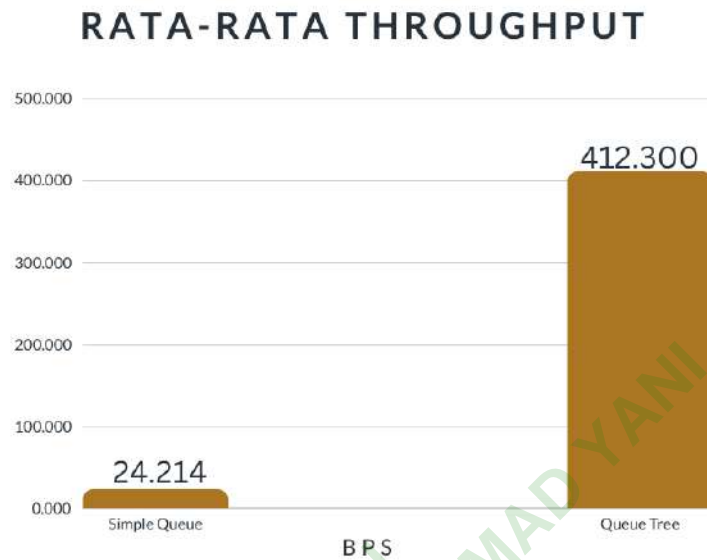
Ruangan	<i>Simple queue</i>				<i>Queue tree</i>			
	<i>Throughput</i> (bps)	<i>Delay</i> (ms)	<i>Jitter</i> (ms)	<i>Packet loss</i> (%)	<i>Throughput</i> (bps)	<i>Delay</i> (ms)	<i>Jitter</i> (ms)	<i>Packet loss</i> (%)
Keuangan	13.227	78.2	77.2	0.5	407.200	4.1	3.1	0.0
Logistik	35.200	27.6	26.6	0.7	417.400	2.8	1.8	0.0
Rata-rata	24.214	52.9	51.9	0.6	412.300	3.5	2.5	0.0

4.11 PERBANDINGAN PARAMETER QoS PADA SIMPLE QUEUE DAN QUEUE TREE

Setelah didapatkan hasil rata-rata dari perbandingan manajemen *bandwidth* metode *Simple queue* dan *Queue tree* sebagaimana terdapat pada tabel 4.5, maka dalam hal ini akan dilakukan tahap analisis terhadap nilai *throughput*, *delay*, *jitter* dan *packet loss*, berdasarkan tabel 4.5 diatas maka dapat dianalisis bahwa:

1. *Throughput*

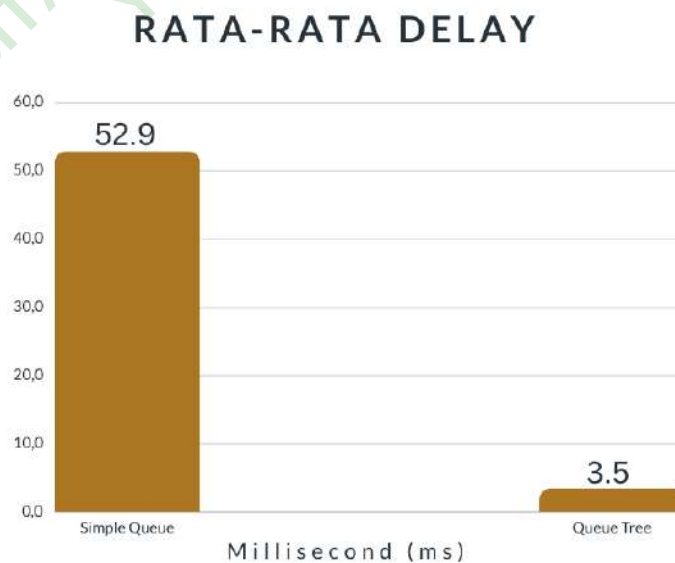
Hasil perbandingan *QoS* pada tabel 4.5, menunjukkan bahwa nilai total rata-rata *throughput* pada *Queue tree* sebesar 412.300 bps lebih bagus jika dibandingkan dengan nilai *throughput* pada *Simple queue* yang hanya 24.214 bps. pemakaian *throughput* menggunakan metode *Queue tree* termasuk dalam kategori ‘sangat bagus’ dengan indeks 4, sedangkan pemakaian *throughput* menggunakan metode *Simple queue* termasuk dalam kategori ‘jelek’ dengan indeks 1.



Gambar 4.119 Grafik perbandingan *throughput* *Simple queue* dan *Queue tree*

2. Delay

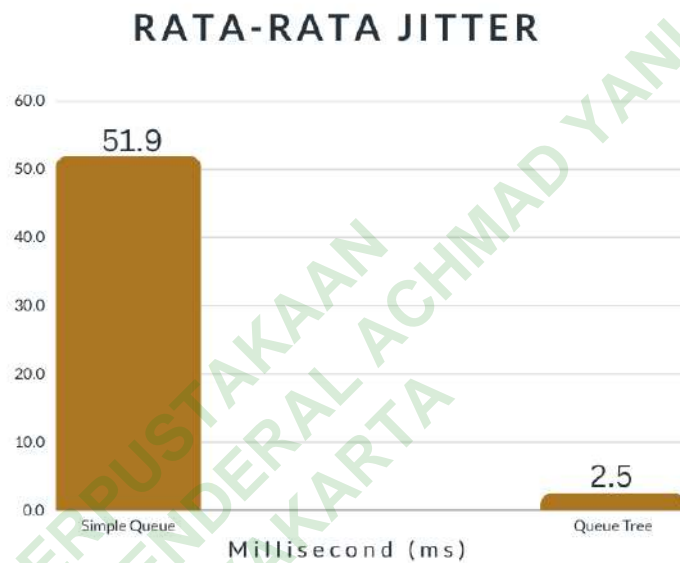
Hasil perbandingan *QoS* pada tabel 4.5, menunjukkan bahwa nilai total rata-rata *delay* pada *Queue tree* sebesar 3.5 ms lebih bagus jika dibandingkan dengan nilai *delay* pada *Simple queue* yaitu 52.9 ms. Pemakaian *delay* menggunakan metode *Queue tree* dan *Simple queue* termasuk dalam kategori 'sangat bagus' dengan indeks 4.



Gambar 4.120 Grafik perbandingan *delay* *Simple queue* dan *Queue tree*

3. *Jitter*

Hasil perbandingan *QoS* pada tabel 4.5, menunjukkan bahwa nilai total rata-rata *jitter* pada *Queue tree* sebesar 2.5 ms lebih bagus jika dibandingkan dengan nilai *jitter* pada *Simple queue* yaitu 51.9 ms. Pemakaian *jitter* menggunakan metode *Queue tree* dan *Simple queue* termasuk dalam kategori ‘bagus’ dengan indeks 3.

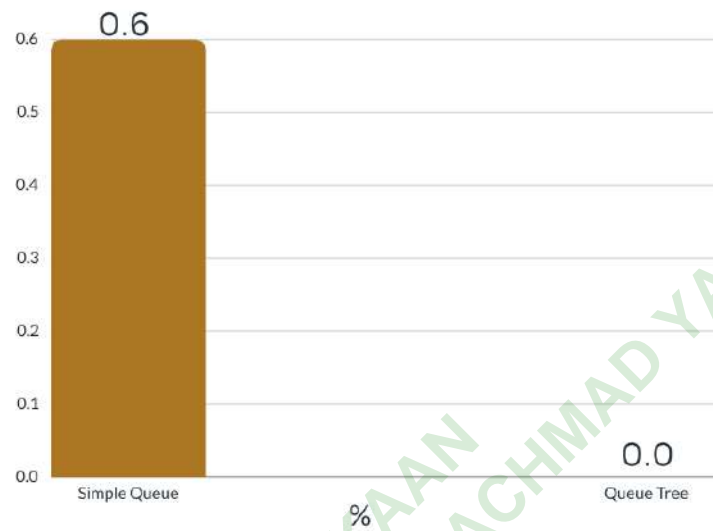


Gambar 4.121 Grafik perbandingan *jitter* *Simple queue* dan *Queue tree*

4. *Packet loss*

Hasil perbandingan *QoS* pada tabel 4.5, menunjukkan bahwa nilai total rata-rata *packet loss* pada *Queue tree* sebesar 0.0% lebih bagus jika dibandingkan dengan nilai *packet loss* pada *Simple queue* yaitu 0.6%. Pemakaian *packet loss* menggunakan metode *Queue tree* termasuk dalam kategori ‘sangat bagus’ dengan indeks 4, sedangkan *Simple queue* termasuk dalam kategori ‘bagus’ dengan indeks 3.

RATA-RATA PACKET LOSS



Gambar 4.122 Grafik perbandingan *packet loss* Simple queue dan Queue tree