

**MATA KULIAH
KOMUNIKASI DATA**



DISUSUN OLEH

NAMA : SRIWAHYUNI
NIM : 09011282025045
KELAS : SK4A INDRALAYA
JURUSAN : SISTEM KOMPUTER
DOSEN PENGAMPUH : ADI HERMANSYAH,M.T.

**PROGRAM STUDI SISTEM KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS SRIWIJAYA**

2022

LAPORAN QOS

TERHADAP TRAFFIC JARINGAN

I. Judul praktikum

Analisis QOS Terhadap Traffic Jaringan Menggunakan Tools Wireshark

II. Tujuan praktikum

- 1) Mengerti dan memahami QOS (Quality Of Service) pada jaringan.
- 2) Mampu mengukur QOS terhadap traffic jaringan menggunakan tools wireshark.
- 3) Mampu menganalisa QOS terhadap traffic jaringan menggunakan tools wireshark.

III. Dasar teori

Wireshark adalah satu dari sekian banyaknya tools Network Analyzer yang dipakai oleh orang-orang yang bekerja di bidang jaringan yang ingin melihat atau menganalisa paket jaringan, pengembangan protocol jaringan serta edukasi bagi yang ingin memperdalam ilmunya dalam jaringan komputer. Aplikasi ini juga dapat menangkap paket-paket data/informasi yang ada dalam jaringan yang kita ingin lihat. Semua jenis paket informasi dalam berbagai format protocol pun akan dengan mudah ditangkap dan dianalisa.

Fungsi dari aplikasi wireshark adalah dapat menganalisa transmisi paket data dalam jaringan, proses koneksi dan transmisi data antar komputer. Wireshark sendiri juga memiliki fitur yang cukup lengkap, diantaranya yaitu:

- Multiplatform, bisa dipakai untuk beberapa basis sistem operasi (Unix, Mac, Windows, serta Linux).
- Bisa dilakukan capture paket data jaringan secara real time.
- Bisa menampilkan informasi protocol jaringan dari paket data secara komplit.
- Paket data bisa disimpan jadi file serta nantinya bisa di buka Kembali untuk analisis lebih lanjut.
- Filtering paket data jaringan.
- Pencarian paket data dengan persyaratan spesifik.

- Pewarnaan paket data dengan persyaratan spesifik.
- Pewarnaan penampilan paket data untuk memudahkan analisis paket data.
- Menampilkan data statistic.
- Untuk melakukan capture paket data yang keluar maupun masuk pada jaringan, wireshark membutuhkan piranti fisik NIC (Network Interface Card).

➤ Quality Of Service (QOS)

Quality Of Service (QOS) adalah kemampuan suatu jaringan untuk menyediakan layanan yang baik dengan menyediakan bandwidth, mengatasi jitter dan delay serta paket loss. QOS sangat ditentukan oleh kualitas jaringan yang digunakan. Pengukuran parameter QOS yaitu Throughput, Delay, Packet loss, dan jitter. Berikut pengertian dari setiap parameter QOS serta cara menghitungnya:

- 1) Throughput adalah kecepatan rata-rata yang diterima oleh suatu node dalam selang waktu pengamatan tertentu.

$$\text{Rumus Throughput} = \frac{\text{Jumlah data yang dikirim (bytes)}}{\text{waktu pengiriman (time span)}} \times 8$$

- 2) Packet Loss adalah banyaknya paket yang hilang pada suatu jaringan paket yang disebabkan oleh tabrakan (collision) dan congestion pada jaringan serta penurunan paket yang disebabkan oleh habisnya TTL (Time To Live) paket.

$$\text{Rumus packet loss} = \frac{\text{jumlah paket yang dikirim} - \text{jumlah paket yang diterima}}{\text{jumlah paket yang di kirim}} \times 100\%$$

- 3) Delay adalah waktu tundaan saat paket yang diakibatkan oleh proses transmisi dari suatu titik menuju titik lain yang menjadi tujuannya.

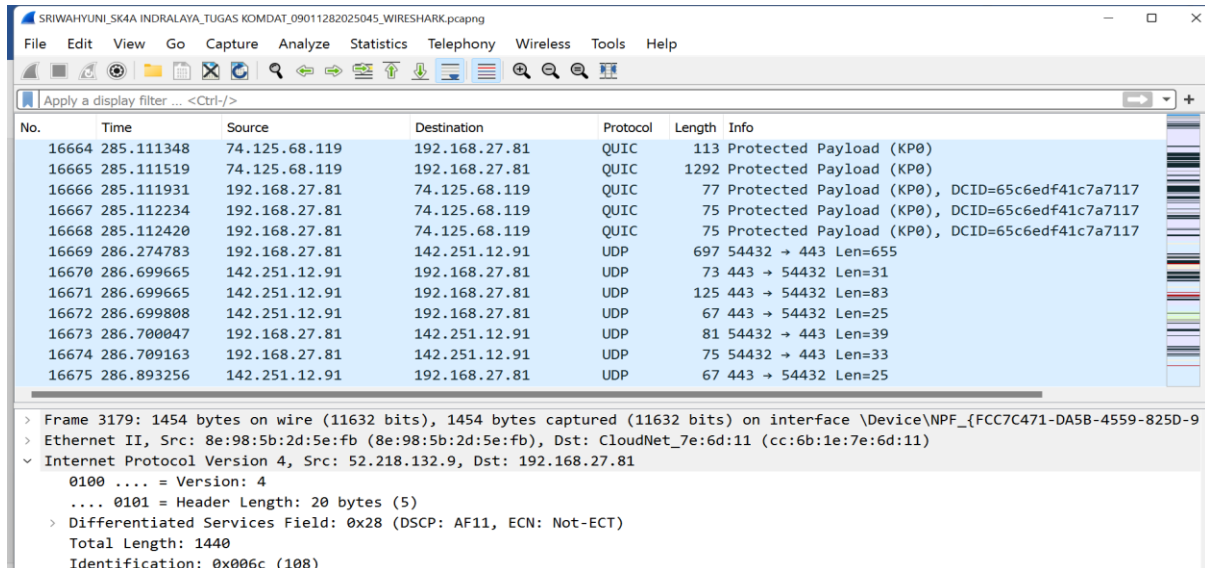
$$\text{Rumus delay} = \text{time2} - \text{time1}$$

- 4) Jitter adalah variasi atau perubahan latency dari delay atau variasi waktu kedatangan paket. Jitter juga didefinisikan sebagai gangguan pada komunikasi digital maupun analog yang disebabkan oleh perubahan sinyal karena referensi posisi waktu. Adanya jitter ini dapat mengakibatkan hilangnya data, terutama pada pengiriman hilangnya data, terutama pada pengiriman data dengan kecepatan tinggi.

$$\text{Rumus jitter} = \text{delay 1} - \text{delay 2}$$

IV. Percobaan: Menggunakan Wireshark

Pertama-tama saya merunning tools dari wireshark ini dan akan muncul data-data lalu lintas jaringan seperti pada gambar terlampir dibawah ini. Disini saya merunning tools dari wireshark ini sekitar selama 5 menit sambil membuka mengakses youtube dan dipatikanlah tampilan seperti pada gambar dibawah ini.



No.	Time	Source	Destination	Protocol	Length	Info
16664	285.111348	74.125.68.119	192.168.27.81	QUIC	113	Protected Payload (KP0)
16665	285.111519	74.125.68.119	192.168.27.81	QUIC	1292	Protected Payload (KP0)
16666	285.111931	192.168.27.81	74.125.68.119	QUIC	77	Protected Payload (KP0), DCID=65c6edf41c7a7117
16667	285.112234	192.168.27.81	74.125.68.119	QUIC	75	Protected Payload (KP0), DCID=65c6edf41c7a7117
16668	285.112420	192.168.27.81	74.125.68.119	QUIC	75	Protected Payload (KP0), DCID=65c6edf41c7a7117
16669	286.274783	192.168.27.81	142.251.12.91	UDP	697	54432 → 443 Len=655
16670	286.699665	142.251.12.91	192.168.27.81	UDP	73	443 → 54432 Len=31
16671	286.699665	142.251.12.91	192.168.27.81	UDP	125	443 → 54432 Len=83
16672	286.699808	142.251.12.91	192.168.27.81	UDP	67	443 → 54432 Len=25
16673	286.700047	192.168.27.81	142.251.12.91	UDP	81	54432 → 443 Len=39
16674	286.709163	192.168.27.81	142.251.12.91	UDP	75	54432 → 443 Len=33
16675	286.893256	142.251.12.91	192.168.27.81	UDP	67	443 → 54432 Len=25

> Frame 3179: 1454 bytes on wire (11632 bits), 1454 bytes captured (11632 bits) on interface \Device\NPF_{FCC7C471-DA5B-4559-825D-9...}

> Ethernet II, Src: 8e:98:5b:2d:5e:fb (8e:98:5b:2d:5e:fb), Dst: CloudNet_7e:6d:11 (cc:6b:1e:7e:6d:11)

> Internet Protocol Version 4, Src: 52.218.132.9, Dst: 192.168.27.81

0100 = Version: 4

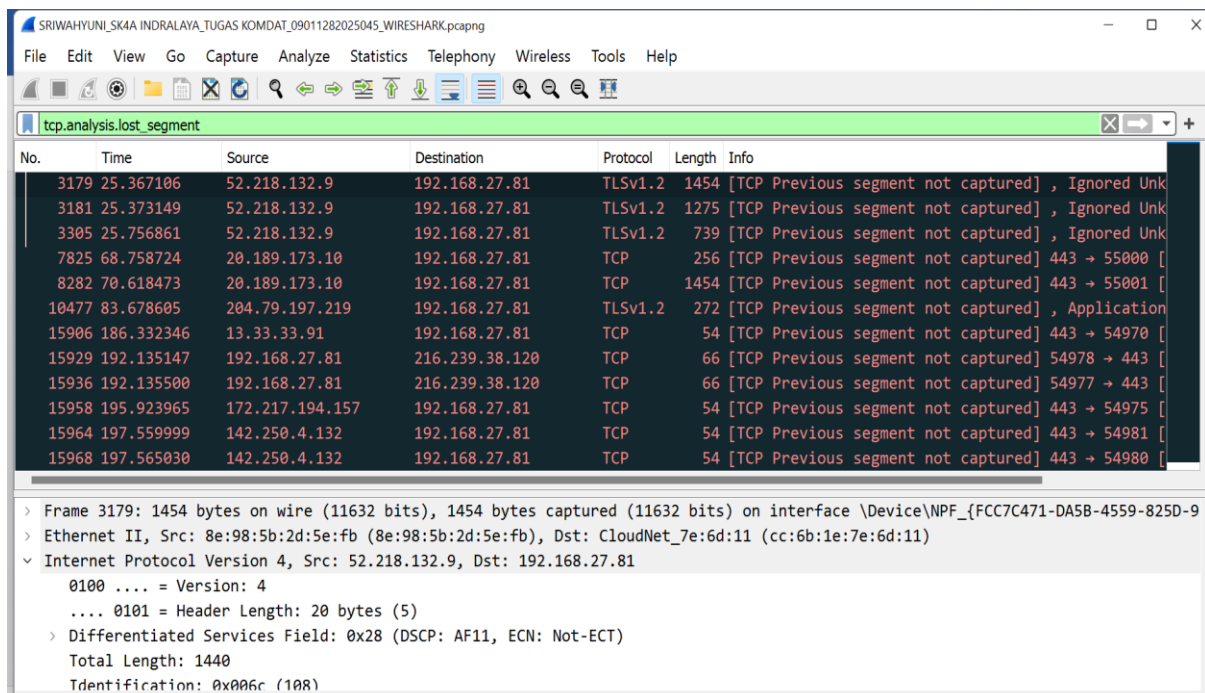
.... 0101 = Header Length: 20 bytes (5)

> Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)

Total Length: 1440

Identification: 0x006c (108)

Setelah saya menunggu sekitar 5 menit selanjutnya saya stopkan dari proses running tadi, dan didapatkanlah lalu lintas jaringan seperti gambar terlampir. Lalu saya Mengecek paket lost dengan mengetikkan *tcp.analysis.lost_segment* untuk melihat berapa banyak data lost yang didapat pada percobaan kali ini.



No.	Time	Source	Destination	Protocol	Length	Info
3179	25.367106	52.218.132.9	192.168.27.81	TLSv1.2	1454	[TCP Previous segment not captured], Ignored Unk
3181	25.373149	52.218.132.9	192.168.27.81	TLSv1.2	1275	[TCP Previous segment not captured], Ignored Unk
3305	25.756861	52.218.132.9	192.168.27.81	TLSv1.2	739	[TCP Previous segment not captured], Ignored Unk
7825	68.758724	20.189.173.10	192.168.27.81	TCP	256	[TCP Previous segment not captured] 443 → 55000 [
8282	70.618473	20.189.173.10	192.168.27.81	TCP	1454	[TCP Previous segment not captured] 443 → 55001 [
10477	83.678605	204.79.197.219	192.168.27.81	TLSv1.2	272	[TCP Previous segment not captured], Application
15906	186.332346	13.33.33.91	192.168.27.81	TCP	54	[TCP Previous segment not captured] 443 → 54970 [
15929	192.135147	192.168.27.81	216.239.38.120	TCP	66	[TCP Previous segment not captured] 54978 → 443 [
15936	192.135500	192.168.27.81	216.239.38.120	TCP	66	[TCP Previous segment not captured] 54977 → 443 [
15958	195.923965	172.217.194.157	192.168.27.81	TCP	54	[TCP Previous segment not captured] 443 → 54975 [
15964	197.559999	142.250.4.132	192.168.27.81	TCP	54	[TCP Previous segment not captured] 443 → 54981 [
15968	197.565030	142.250.4.132	192.168.27.81	TCP	54	[TCP Previous segment not captured] 443 → 54980 [

> Frame 3179: 1454 bytes on wire (11632 bits), 1454 bytes captured (11632 bits) on interface \Device\NPF_{FCC7C471-DA5B-4559-825D-9...}

> Ethernet II, Src: 8e:98:5b:2d:5e:fb (8e:98:5b:2d:5e:fb), Dst: CloudNet_7e:6d:11 (cc:6b:1e:7e:6d:11)

> Internet Protocol Version 4, Src: 52.218.132.9, Dst: 192.168.27.81

0100 = Version: 4

.... 0101 = Header Length: 20 bytes (5)

> Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)

Total Length: 1440

Identification: 0x006c (108)

Kemudian juga saya mengetikkan *tcp* untuk mailhat berapa protocol tcp yang didapatkan, yang bisa dilihat pada gambar terlampir dibawah ini.

No.	Time	Source	Destination	Protocol	Length	Info
2775	23.924748	142.251.12.156	192.168.27.81	TCP	66	443 → 54988 [ACK] Seq=1 Ack=2 Win=277 Len=0 SLE=1
2853	24.220021	192.168.27.81	204.79.197.219	TCP	55	54989 → 443 [ACK] Seq=1 Ack=1 Win=256 Len=1 [TCP
2872	24.272372	204.79.197.219	192.168.27.81	TCP	66	443 → 54989 [ACK] Seq=1 Ack=2 Win=2050 Len=0 SLE=
3003	24.772527	192.168.27.81	117.18.237.29	TCP	54	54986 → 80 [FIN, ACK] Seq=1 Ack=1 Win=253 Len=0
3017	24.841088	117.18.237.29	192.168.27.81	TCP	54	80 → 54986 [FIN, ACK] Seq=1 Ack=2 Win=131 Len=0
3018	24.841243	192.168.27.81	117.18.237.29	TCP	54	54986 → 80 [ACK] Seq=2 Ack=2 Win=253 Len=0
3031	24.857856	192.168.27.81	52.218.132.9	TCP	66	54998 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460
3102	25.100138	52.218.132.9	192.168.27.81	TCP	66	443 → 54998 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=
3103	25.100310	192.168.27.81	52.218.132.9	TCP	54	54998 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
3104	25.107493	192.168.27.81	52.218.132.9	TLSv1.2	375	Client Hello
3177	25.359828	52.218.132.9	192.168.27.81	TCP	54	443 → 54998 [ACK] Seq=1 Ack=322 Win=30464 Len=0
3179	25.367106	52.218.132.9	192.168.27.81	TLSv1.2	1454	[TCP Previous segment not captured] , Ignored Unk

> Frame 3179: 1454 bytes on wire (11632 bits), 1454 bytes captured (11632 bits) on interface \Device\NPF_{FCC7C471-DA5B-4559-825D-9
 > Ethernet II, Src: 8e:98:5b:2d:5e:fb (8e:98:5b:2d:5e:fb), Dst: CloudNet_7e:6d:11 (cc:6b:1e:7e:6d:11)
 > Internet Protocol Version 4, Src: 52.218.132.9, Dst: 192.168.27.81
 > 0100 = Version: 4
 > 0101 = Header Length: 20 bytes (5)
 > Differentiated Services Field: 0x28 (DSCP: AF11, ECN: Not-ECT)
 > Total Length: 1440
 > Identification: 0x006c (108)

V. Data percobaan

Menghitung Parameter QOS dari penggunaan percobaan wireshark

1. Troughput :

$$\begin{aligned}
 \text{Rumus Throughput} &= \text{Jumlah data yang dikirim (bytes)} / \text{waktu pengiriman} \\
 &= (\text{time span}) \times 8 \text{ (untuk mengubah byte ke kilobyte)} \\
 &= 11565099 / 286.893 \times 8 \\
 &= 40,31154 \text{ b} \times 8 \\
 &= 322 \text{ k}
 \end{aligned}$$

2. packet loss :

$$\begin{aligned}
 \text{Rumus packet loss} &= (\text{jumlah paket yang dikirim} - \text{jumlah paket yang diterima}) / (\text{jumlah paket yang di kirim}) \times 100\% \\
 &= (16675 - 16634) / (16675) \times 100\% \\
 &= 0,2
 \end{aligned}$$

3. delay :

$$\begin{aligned}
 \text{Rumus delay} &= \text{time2} - \text{time1} \\
 \text{Total Delay} &= 286,893256 \text{ s} \\
 \text{Rata-rata Delay} &= 0,017206025 \text{ s}
 \end{aligned}$$

4. jitter :

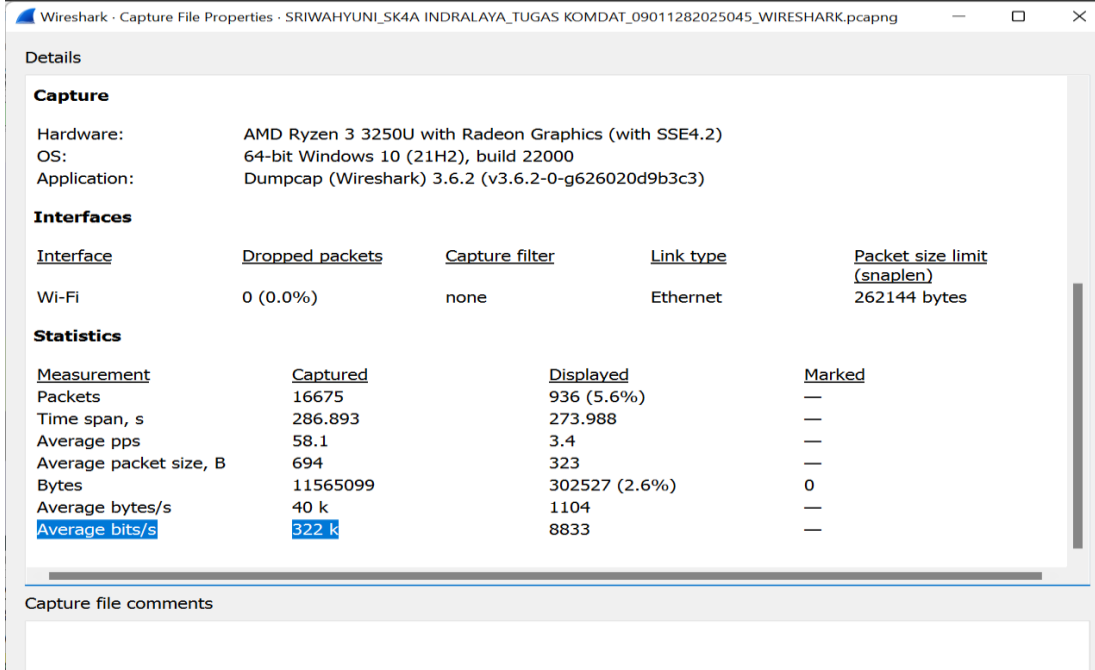
$$\begin{aligned}
 \text{Rumus jitter} &= \text{delay 1} - \text{delay 2} \\
 \text{Total Jitter} &= 287,077349 \text{ s} \\
 \text{Rata-rata Jitter} &= 0,017217065 \text{ s}
 \end{aligned}$$

VI. Pembahasan/ Analisa percobaan

Pada percobaan wireshark ini saya mengakses youtube untuk mengamati lalu lintas jaringannya. Didapatkan jumlah data yang didapat sebanyak 16675 data. Lalu, saya menghitung parameter QOS dan didapatkan pembahasan seperti berikut ini.

1. Troughput

pada data percobaan yang sudah kita lakukan sebelumnya didapatkan bahwa hasil troughput sebesar 322 k. nilai yang didapat pada nilai throughput ini sama dengan data yang didapat pada wireshark yaitu pada average bits/s adalah 322 k. ini menandakan bahwa perhitungan diatas benar sesuai dengan data yang tertera pada throughput di wireshark. Bisa dilihat pada gambar terlampir dibawah ini.



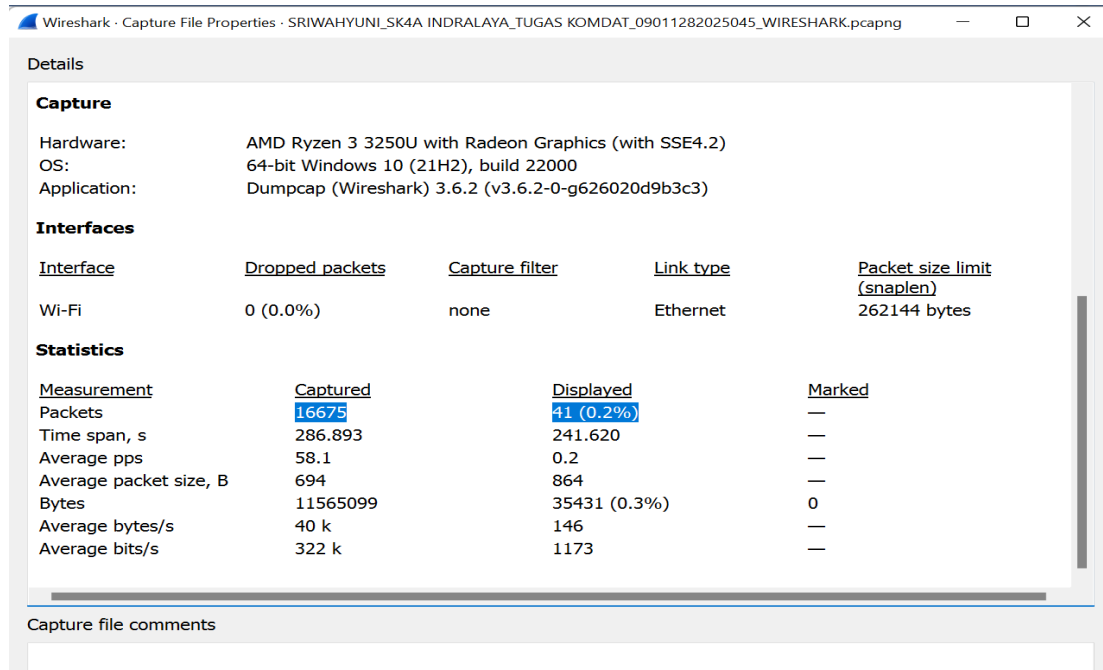
The screenshot shows the 'Details' pane of the Wireshark 'Capture File Properties' window. It displays hardware, OS, and application information under 'Capture', and interface details under 'Interfaces'. The 'Statistics' section contains a table with four columns: Measurement, Captured, Displayed, and Marked. The 'Average bits/s' row is highlighted in blue, showing a value of 322 k.

Capture				
Hardware:	AMD Ryzen 3 3250U with Radeon Graphics (with SSE4.2)			
OS:	64-bit Windows 10 (21H2), build 22000			
Application:	Dumpcap (Wireshark) 3.6.2 (v3.6.2-0-g626020d9b3c3)			
Interfaces				
Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Wi-Fi	0 (0.0%)	none	Ethernet	262144 bytes
Statistics				
Measurement	Captured	Displayed	Marked	
Packets	16675	936 (5.6%)	—	
Time span, s	286.893	273.988	—	
Average pps	58.1	3.4	—	
Average packet size, B	694	323	—	
Bytes	11565099	302527 (2.6%)	0	
Average bytes/s	40 k	1104	—	
Average bits/s	322 k	8833	—	

2. Packet loss

Untuk data packet loss yang sudah diperoleh pada data percobaan diatas didapatkan bahwa nilai packet loss sebesar 0,2. Nilai tersebut sama dengan data pada tools di wireshark yaitu di bagian packets displayed yaitu 0,2. Untuk nilai 0,2

ini termasuk dalam kategori sangat bagus. Ini berarti perhitungan kita sudah benar karena sama dengan data pada wireshark seperti pada gambar dibawah ini.



3. Delay

Telah didapatkan dari percobaan yang sudah dilakukan bahwa:

$$\text{Rumus delay} = \text{time2} - \text{time1}$$

$$\text{Total Delay} = 286,893256 \text{ s}$$

$$\text{Rata-rata Delay} = 0,017206025 \text{ s}$$

$$= 17,206025 \text{ ms} \rightarrow \text{masuk dalam kategori excellent.}$$

4. Jitter

Telah didapatkan dari percobaan yang sudah dilakukan bahwa:

$$\text{Rumus jitter} = \text{delay 1} - \text{delay 2}$$

$$\text{Total Jitter} = 287,077349 \text{ s}$$

$$\text{Rata-rata Jitter} = 0,017217065 \text{ s}$$

$$= 17,2170565 \text{ ms} \rightarrow \text{masuk kedalam kategori bagus.}$$

Dari percobaan wireshark untuk traffic jaringan ini didapatkan jaringan saya sesuai dengan standart berarti sudah termasuk jaringan bagus.

VII. Kesimpulan

1. Parameter Packet Loss dari keseluruhan perhitungan menunjukkan rata-rata nilai packet loss termasuk dengan katagori sangat bagus karena didapatkan nilai rata-rata packet loss sebesar 0,2 yang artinya packet loss berkatagori sangat bagus.
2. Parameter Delay dari keseluruhan perhitungan menunjukkan rata-rata nilai delay termasuk dengan kategori excellent. Karena didapatkan nilai rata-rata delay sebesar 17,206025 ms yang artinya berkatagori excellent.
3. Parameter Jitter dari keseluruhan perhitungan menunjukkan rata-rata nilai Jitter termasuk ke dalam kategori bagus. Karena didapatkan nilai rata-rata jitter sebesar 17,2170565 ms yang artinya berkatagori bagus.
4. Dari percobaan wireshark untuk traffic jaringan ini didapatkan jaringan saya sesuai dengan standart berarti sudah termasuk jaringan bagus.

- **Lampiran :**

<https://github.com/SRIWAHYUNISK20/SRI>