ANALISIS PERFORMA *WEB SERVER* POLITEKNIK NEGERI LHOKSEUMAWE DENGAN METODE *STRESS TESTING*

Afil Tarick Pasha[1], Syamsul[2], Yassir[3]

[1,2,3)]Program Studi Teknologi Rekayasa Jaringan Telekomunikasi
Jurusan Teknik Elektro Politeknik Negeri Lhokseumawe
Email: filpasha70@gmail.com[1], syamsul10466@gmail.com[2], yassirjalil@gmail.com[3]

ABSTRAK

Di Politeknik Negeri Lhokseumawe, *web server* memainkan peran krusial dalam mendukung berbagai layanan online seperti portal akademik dan sistem manajemen informasi. Dengan meningkatnya jumlah pengguna dan beban kerja, performa *web server* perlu dievaluasi untuk memastikan kelancaran layanan. Penelitian ini menggunakan metode *Stress Testing* untuk menganalisis kemampuan *web server* dalam menangani beban tinggi serta untuk mengidentifikasi batasan dan kelemahan yang ada. Hasil pengujian menunjukkan bahwa *web server* memiliki performa yang sangat baik, dengan *throughput* yang meningkat dari 2,316 bps pada 4000 *thread* menjadi 3,574 bps pada 6000 *thread*. Tingkat *packet loss* yang terdeteksi sangat rendah, bahkan pada beban tinggi tidak ada *packet loss* sama sekali. Selain itu, *server* menunjukkan waktu respons yang cepat dengan *delay* menurun hingga 1,08 ms pada beban 6000 *thread*. Kesimpulannya, *web server* Politeknik Negeri Lhokseumawe mampu menangani beban kerja yang tinggi dengan reliabilitas dan efisiensi yang optimal, mempertahankan kualitas layanan yang diperlukan untuk mendukung operasional kampus.

Kata kunci : Web Server, Stress Testing, Politeknik Negeri Lhokseumawe, Throughput, Packet Loss, Delay.

I. PENDAHULUAN

Di era digital yang semakin maju, keberlanjutan layanan teknologi informasi menjadi aspek kritis dalam mendukung kegiatan akademik dan administratif di institusi pendidikan tinggi seperti Politeknik Negeri Lhokseumawe. Web server yang digunakan untuk berbagai layanan online, seperti portal akademik, sistem manajemen informasi, dan layanan lainnya, memainkan peran penting dalam memastikan kelancaran operasional kampus.

Namun, dengan semakin tingginya jumlah pengguna dan beban kerja yang dihadapi, *web server* sering kali diuji kemampuannya dalam menangani situasi ekstrem. Kondisi ini dapat menyebabkan berbagai kendala, mulai dari penurunan performa hingga terhentinya layanan, yang pada akhirnya dapat mengganggu aktivitas sehari-hari di kampus.

Untuk mengidentifikasi dan memahami batasan serta kendala yang dihadapi oleh *web server*, diperlukan suatu metode pengujian yang dapat mensimulasikan beban kerja ekstrem. Metode *Stress Testing* adalah salah satu pendekatan yang efektif dalam mengevaluasi sejauh mana kemampuan *web server* dalam menangani beban yang tinggi dan mengidentifikasi titik lemah dalam sistem.

Penelitian ini bertujuan untuk mengidentifikasi batasan dan kendala yang di hadapi oleh *web server* Politeknik negeri Lhokseumawe melalui pengujian menggunakan metode *Strees Testing*. Selain itu, penelitian ini juga bertujuan untuk mengungkap kelemahan dalam sistem *web server*, sehingga dapat memberikan rekomendasi yang konkret untuk meningkatkan reliabilitas dan efesiensi pengolahan serta pengoperasian *server* di masa depan.

II. TINJAUAN PUSTAKA

A. Konsep Dasar *Web Server*

Web server adalah sebuah *software* dalam sebuah *server* yang berfungsi menerima permintaan (*Request*) berupa halaman *Website* melalui HTTP atau HTTPs dari *Client* (*Browser*) dan mengirimkan kembali (*Response*) dalam bentuk halaman – halaman *Website* yang umumnya berbentuk HTML.

Web Server juga memilki fungsi tidak hanya mengolah data tapi dapat juga mengirimkan data berupa file foto dan video berdasarkan permintaan *Client. Web Server* dapat berjalan secara *Online*[1].

Berikut adalah fungsi dari *web server*:

1. Menyimpan Data: *Web server* menyimpan konten web, termasuk file HTML, gambar, dan dokumen lainnya.
2. Memproses Data: *Web server* dapat memproses data secara dinamis menggunakan skrip atau program, seperti PHP, untuk menghasilkan halaman web yang disesuaikan dengan permintaan klien.
3. Mengelola Permintaan: *Web server* mengelola lalu lintas permintaan HTTP/HTTPS dari klien, memprioritaskan, dan meresponnya sesuai urutan masuk.
4. Keamanan: *Web server* menyediakan fitur keamanan, seperti enkripsi SSL/TLS, untuk melindungi data yang ditransfer antara *server* dan klien.
5. *Caching*: Web *server* dapat menyimpan salinan statis dari halaman web untuk mengurangi waktu respon dan mempercepat pengiriman konten kepada pengguna yang sering mengakses halaman yang sama.

B. TCP/IP *(Transmision Control Protocol/Internet Protocol)*

TCP/IP adalah kumpulan protokol yang dirancang untuk menjalankan fungsi komunikasi penting. Kumpulan ini terdiri dari beberapa protokol, masing-masing bertanggung jawab atas bagian tertentu dalam komunikasi data. Desain protokol TCP/IP bersifat modular, memungkinkan pembagian tugas yang jelas dan sederhana untuk setiap protokol. Oleh karena itu, satu protokol tidak perlu memahami detail protokol lain, asalkan mereka dapat bertukar data secara efektif. Fleksibilitas TCP/IP sebagai protokol komunikasi data terletak pada prinsip modular ini, sehingga implementasinya mudah disesuaikan dengan berbagai jenis komputer dan antarmuka jaringan, karena sebagian besar protokolnya tidak terikat pada perangkat atau komputer tertentu. Untuk mengaktifkan TCP/IP pada antarmuka jaringan, perubahan hanya perlu dilakukan pada protokol yang relevan dengan antarmuka tersebut. Model TCP/IP yang terdiri dari empat lapisan menyediakan kerangka kerja yang kuat untuk menjelaskan interaksi dan fungsi setiap protokol dalam sistem ini[2].

C. Wireshark

Wireshark merupakan tools yang bertujuan untuk menganalisa paket data yang ada pada jaringan internet. Wireshark juga termasuk *Network Packet Analyzer* yang fungsinya untuk menangkap semua data informasi yang ada saat komunikasi data di jaringan internet dan menampilkan informasi data tersebut sedetail mungkin. Wireshark juga *tools* yang fleksibel dalam artian Wireshark bisa memerika data baik itu yang terjadi pada jaringan internet kabel maupun wireless. Wireshark adalah sebuah aplikasi yang digunakan untuk menangkap aktivitas jaringan data yang sedang berjalan untuk di pindai dan di analisa serta membuat suatu penjelasan secara detail terkait trafik data yang ada pada jaringan internet tersebut[3].

Manfaat penggunaan aplikasi Wireshark meliputi:

1. Menangkap informasi atau data paket yang dikirim dan diterima dalam jaringan komputer.
2. Memantau aktivitas yang terjadi dalam jaringan komputer.
3. Menilai dan menganalisis kinerja jaringan komputer, termasuk kecepatan akses, berbagi data, dan koneksi jaringan ke internet.
4. Mengawasi keamanan jaringan komputer[4].

Beberapa kegunaan Wireshark meliputi:

1. Wireshark digunakan oleh administrator jaringan untuk menganalisis lalu lintas dalam jaringannya.
2. Wireshark dapat menangkap paket data dan informasi yang sedang terjadi di dalam jaringan, memungkinkan analisis yang mudah terhadap semua jenis informasi yang diperoleh.

D. Apache JMeter

Apache JMeter merupakan "*open-source testing tool*" yang digunakan untuk melakukan proses pengujian kinerja dari sebuah *Website*. Hal ini berarti dapat digunakan oleh siapapun tanpa harus membayar lisensi. Apache JMeter dikembangkan oleh Apache Software Foundation (ASF)[5].

Fungsi utama JMeter adalah mengakses klien/*server* uji. Selain itu, JMeter digunakan dalam pengujian regresi dengan menghasilkan skrip pengujian. JMeter menyediakan pelaporan *offline* dari hasil pengujian yang dilakukan. JMeter dapat digunakan untuk menganalisis dan mengukur kinerja aplikasi web atau jangkauan layanan. JMeter dapat mengetahui kinerja dari sebuah *Website* terhadap tekanan kinerja ketika banyak pengguna atau pengunjung yang mengakses *Website*. JMeter awalnya digunakan untuk menguji aplikasi web atau aplikasi FTP, dalam perkembangannya JMeter juga dapat menguji fungsionalitas *server database*.

E. *Stress Testing*

Dalam *software development*, *stress testing* adalah pengujian yang penting dilakukan. Apalagi, jika aplikasi atau web yang dikembangkan akan digunakan oleh banyak orang. Menguntip *Geeks for Geeks*, *stress testing* adalah teknik pengujian terhadap sebuah *sotware*. Teknik ini dilakukan untuk mengetahui ketahanan *software* atau sistem saat digunakan melampaui batas operasional normal. Dalam kondisi ekstrem ini, *developer* jadi tahu stabilitas dan reabilitas dari sebuah sistem. Tes ini juga melihat apakah sistem bisa beroperasi normal lagi setelah diharuskan berfungsi melewati batas normalnya[6].

F. *Quality of Service (QoS)*

Quality of Service (QoS) adalah penilaian terhadap seberapa baik jaringan, dan merupakan upaya untuk menggambarkan karakteristik dan sifat suatu layanan. QoS mencakup kemampuan jaringan untuk menyediakan layanan yang unggul dengan menyediakan lebar pita, mengatasi jitter, dan mengurangi *delay*. Parameter QoS merujuk pada kinerja tingkat kecepatan dan keandalan pengiriman berbagai jenis data selama komunikasi. Parameter-parameter QoS melibatkan *throughput*, *packet loss*, *delay*, *jitter* (variasi kedatangan paket), dan MOS (*Mean Opinion Score*)[7].

1. *Throughput*

   Throughput adalah ukuran aktual dari *bandwidth* selama suatu periode waktu tertentu saat mentransmisikan berkas. Meskipun satuan yang digunakan sama dengan *bandwidth*, yaitu bits per second (bps), *throughput* lebih mencerminkan *bandwidth* yang sesungguhnya pada waktu tertentu dan dalam kondisi serta jaringan tertentu saat mengunduh suatu *file* dengan ukuran tertentu. *Throughput* menggambarkan jumlah total paket yang berhasil tiba di tujuan selama interval waktu

tertentu, dibagi oleh durasi interval waktu tersebut. Nilai *throughput* dapat dihitung dengan menggunakan suatu persamaan. *Throughput* dapat dihitung dengan menggunakan rumus sebagai berikut.

$$Throughput = \frac{packet\ data\ yang\ diterima}{lama\ pengamatan} \quad (1)$$

2. *Packet Loss*

Packet loss adalah persentase paket yang tidak berhasil dikirim saat mentransmisikan data. Ini dapat disebabkan oleh berbagai faktor seperti penurunan sinyal dalam media jaringan, kesalahan perangkat keras jaringan, atau radiasi dari lingkungan sekitar. Parameter *packet loss* mengindikasikan kondisi di mana jumlah total paket yang hilang dapat terjadi, baik karena *collision* maupun *congestion* dalam jaringan. Dampaknya dirasakan oleh semua aplikasi karena retransmisi dapat mengurangi efisiensi jaringan secara keseluruhan, bahkan jika jumlah *bandwidth* yang cukup tersedia untuk aplikasi tersebut. Untuk menghitung nilai *packet loss*, dapat menggunakan suatu persamaan. Untuk menghitung *packet loss* dapat digunakan rumus sebagai berikut.

$$Packet\ loss = \frac{paket\ data\ dikirim - paket\ data\ diterima}{paket\ data\ yang\ dikirim} x100 \ldots\ldots (2)$$

3. *Delay*

Delay merujuk pada waktu yang dibutuhkan oleh data untuk mencapai tujuan dari suatu paket yang dikirimkan dari satu komputer ke komputer lainnya. Dalam konteks transmisi paket dalam jaringan komputer, *delay* dapat disebabkan oleh adanya antrian yang panjang atau pengambilan rute alternatif untuk menghindari kemacetan pada routing. Faktor-faktor yang mempengaruhi *delay* melibatkan jarak, media fisik, kongesti, dan waktu proses yang dibutuhkan. Untuk menghitung *delay* pada paket yang ditransmisikan, dapat menggunakan pembagian panjang paket (dalam satuan bit) dengan *bandwidth* link (dalam satuan bit per detik). Rata-rata *delay* dapat dihitung menggunakan rumus tertentu. Rumus menghitung Rata-rata *delay* dapat dilihat pada rumus sebagai berikut[8].

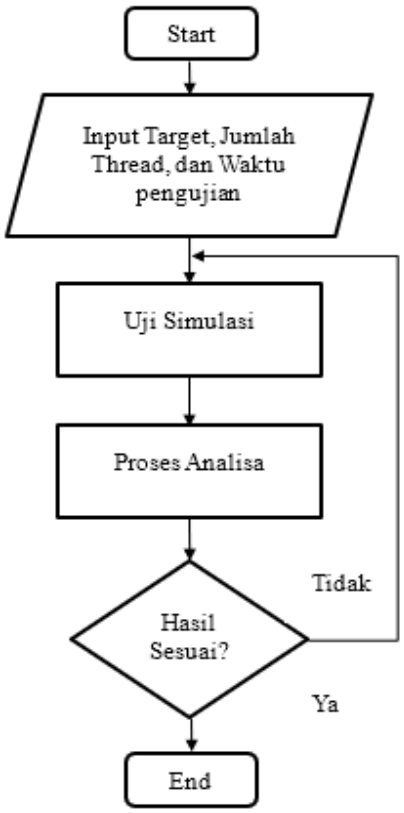
$$Delay\ rata - rata = \frac{total\ delay}{total\ paket\ yang\ diterima} \ldots\ldots (3)$$

G. Performa *Web Server*

Pengujian performa merupakan fase penting dari siklus pengembangan suatu sistem ketika hendak diluncurkan. Salah satu metode pengujian awal yang dilakukan adalah uji performa. Uji performa dipilih karena dapat melakukan diagnosa awal tentang kerangka sistem dilihat dari sudut pandang atau parameter pengukuran tertentu serta dapat memberikan solusi atau rekomendasi agar sistem tersebut menjadi lebih optimal. Uji tingkat stres bertujuan untuk mengukur dan memverifikasi performa sistem secara spesifik, seperti waktu respon dan ketersediaan layanan yang dijalankan, dengan mensimulasikan banyak pengguna mengakses secara bersamaan selama interval waktu yang ditentukan[9].

III. METODOLOGI

Pada bagian ini akan dibahas metodologi yang merupakan tahapan-tahapan yang akan dilakukan oleh peneliti dari perumusan masalah sampai dengan kesimpulan, yang membentuk alur sistematis. Metodologi penelitian digunakan sebagai pedoman peneliti dalam melaksanakan penelitian ini agar hasil yang dicapai tidak menyimpang dari tujuan penelitian.



Gbr 1 Diagram Alir

A. Alat dan Bahan

Alat dan bahan pada penelitian kali ini terbagi dua jenis yaitu *Hardware* dan *Software*.

1. *Hardware*
   - Laptop HP Notebook 14-an029au
   - DDR3
   - AMD Radeon R3 Graphics 1.80 GHz
   - RAM 8GB
   - AMD A4-7210 APU
2. *Software*
   - Apache Jmeter Versi 5.6.3
   - Wireshark

B. Teknik Pengumpulan Data

Teknik pengumpulan data menggunakan metode *Stress Testing* dilakukan menggunakan *tools* Wireshark dimana Wireshark menangkap lalu lintas jaringan antara laptop peneliti dan *web server* saat *Stress Testing* dilakukan menggunakan Jmeter. Data yang dikumpulkan merupakan lalu lintas dari *thread* yang dikirim oleh Jmeter ke *Webserver.*

C. Prosedur Pengujian

Berikut adalah tahapan Pengujian Jmeter:

Gbr 2 Tahapan Pengujian Menggunakan Jmeter

1. Membuat *Test Plan*
   Setelah berhasil membukan JMeter, kita dapat membuat *test plan* yang berisi urutan komponen – komponen uji yang berfungsi sebagai penentu bagaimana sebuah *server* akan disimulasikan.
   Di percobaan kali ini, ubah nama *test plan* menjadi "Pengujian" (100, 250, 500 100) lalu *save* dengan menekan ctrl+ s.

2. Membuat *Thread Group*
   Thread Group merupakan salah satu komponen uji yang ada pada *test plan*. Cara menambah *Thread Group*:
   a. Klik kanan pada nama *Test Plan* yang sudah disimpan, "Pengujian"
   b. Pilih *Add* > *Threads* (*Users*) > *Threads Group*
   c. Pilih *Thread Group*

   Di *thread group* terdapat beberapa properties yang dapat mempengaruhi skenario pengujian performance yang dijalankan;
   a. *Number of Threads* (*users*) : jumlah user virtual yang akan disimulasikan.
   b. *Ramp-up period* (*seconds*) : total durasi yang dibutuhkan seluruh skenario dijalankan dari awal sampai akhir.
   c. *Loop Count* : jumlah percobaan pengujian yang dijalankan. Bisa juga membuat pengujian yang *loop*-nya tak terhingga dengan meng-klik *checkbox* "*Infinite*".

   Jadi, apabila kita memasukkan *number of threads* = 10, *ramp-up period* = 10, dan *loop count* = 1, itu artinya ada 1 *thread* yang dijalankan setiap 1 detik. Apabila memasukkan *ramp-up period* = 120, itu artinya ada 1 *thread* yang dijalankan setiap 10 detik karena 100 *ramp-up period*/10 *threads* = 10. *Kalau ramp-up period* = 200 dan *threads* = 10, artinya ada 1 *thread* yang dijalankan setiap 20 detik.
3. Menambah HTTP *Request Sampler*
   Setelah membuat *Thread Group*, dapat menambahkan http *request sampler*. Http *request sampler* ini merupakan tempat untuk menambahkan informasi berupa *protocol*, IP *Address*, *port number*, dan *method* serta *path* dari *web server* yang hendak diuji. Cara menambahkan http *request sampler*:
   a. Pada *Thread Group*, klik kanan
   b. Klik *"Add"*
   c. Klik *"Sampler"*
   d. Pilih *"HTTP Request"*

   Saat tampilan HTTP *Request* terbuka, masukkan protokol *web server* yang kita gunakan (HTTP/HTTPS) kemudian masukkan IP *Address* atau *server name*. Setelah itu pilih method yang ingin kita gunakan (GET, POST, etc.).
   Di pengujian ini saya tambahkan *protocol* > https, *server name* or IP > 103.248.196.100, http *request* -> GET, dan *path* > https://elearning.pnl.ac.id/.

4. Menambah *Listener*
   Setelah menyusun kerangka mulai dari http *request*, kita butuh *listener*. *Listener* adalah komponen pada JMeter yang menunjukkan hasil dari skenario yang di susun. *Listener* memiliki banyak jenis, apa hasil pengujian itu disajikan. Bisa dalam bentuk *tree*, *graph*, *summary report*, atau dalam bentuk log. Cara menambahkan *listener*:
   a. Pada *Thread Group*, klik kanan
   b. Klik "Add"
   c. Klik "Listener"
   d. Pilih "View Results Tree"
   e. Menambah Graph Result

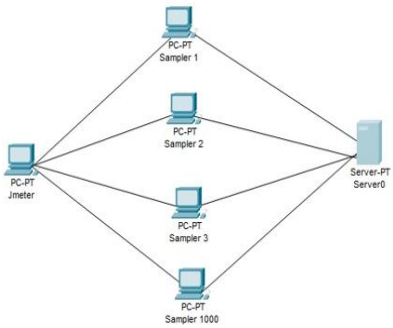
   JMeter dapat menampilkan hasil tes dalam format Grafik. Cara menambahkan *Graph Results*:
   a. Pada *Thread Group*, klik kanan
   b. Klik "Add"
   c. Klik "Listener"
   d. Pilih "Graph Results"

D. Teknik Pengolahan Data

Data yang diambil dari lalu lintas *server* yang didapat dari Wireshark akan diolah melalui proses *filtering*, menghitung *Delay*, *Throughput,* dan *Packet loss* menggunakan Rumus *Quality of Service.* Jumlah data *Thread* yang akan diolah adalah dimulai dari 5000, 5500, dan 6000 *Thread* (*users*).

E. Metode Simulasi

Simulasi dilakukan dengan menggunakan Jmeter untuk mensimulasikan beban pengguna dengan jumlah yang berbeda, dimulai dengan beban terendah hingga beban tertinggi untuk mengetahui bagaimana *server* merespon beban yang meningkat secara bertahap. Skenario simulasi dapat divisualisasikan pada gambar 3.



Gbr 3 Skenario Simulasi

F. Metode Analisis

Analisis dilakukan berdasarkan *Throughput, packet loss,* dan *Delay* yang telah dihitung. Kinerja *Server* dapat dianalisis dari tiap-tiap variabel yang dhitung seperti *Throughput* dalam satuan bps (*bit per second*), *Packet loss* dalam satuan persen (%), dan *Delay* dalam satuan *mili second* (ms).

Semakin besar *Throughput* maka semakin baik kinerja *server* tersebut dan untuk *packet loss,* dan *Delay* semakin kecil nilainya maka, semakin bagus kinerja *server* tersebut.

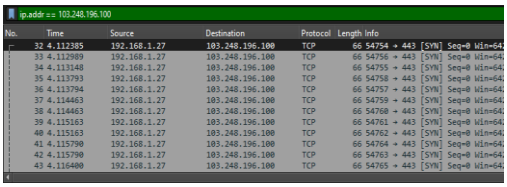
IV. HASIL DAN PEMBAHASAN

A. Hasil Pengujian

Adapun hasil dari pengujian yang dilakukan berdasarkan jumlah *Thread* bervariasi yang dikirimkan dalam waktu 300 detik dengan target "elearning.pnl.ac.id" yang didapatkan adalah sebagai berikut:

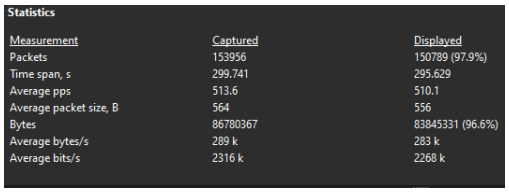
1. Pengujian dengan 4000 *Thread*

   Pada pengujian 4000 *Thread* langkah pertama yang dilakukan melakukan proses *filtering* pada Wireshark berdasarkan IP *Address web server* elearning.pnl.ac.id yaitu 103.248.196.100. Proses *Filtering* dapat dilihat pada gambar 4.



Gbr 4 Proses *Filtering* pada data 4000 *Thread*

Adapun hasil dari pengujian 4000 *Thread* dapat dilihat pada gambar 5.



Gbr 5 Hasil Pengujian 4000 *Thread*

Dimana hasil yang didapat dalam waktu pengujian 300 detik dapat dilihat pada tabel 1.

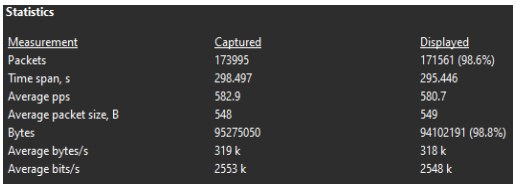
TABEL I
Hasil Perhitungan Qos 4000 Thread

Throughput	Jumlah Byte / Time Span = 86780367 x 8 = 694,242,936 / 299,741 = 2,316 bps
Packet loss	Paket dikirim - Paket diterima / Paket dikirim x 100% (153956 - 150742) : 153956 x 100 (3214 : 153956) x 100 0,0208760945 x 100 = 2,08%
Delay	*Delay* Rata – Rata = 1,92 ms

Pada pengujian 4000 *Thread* diatas web "elearning.pnl.ac.id" masih dapat diakses dan berjalan normal. Ini menenujukkan *web server* dapat menahan beban 4000 *Thread* dalam waktu 300 detik.

2. Pengujian dengan 4500 Thread

   Adapun hasil pengujian 4500 *Thread* dalam waktu 300 detik dapat dilihat pada gambar 6 dibawah.



Gbr 6 Hasil Pengujian 4500 *Thread*

Berdasarkan hasil pengujian diatas, dapat dilihat perhitungan *Quality of Service* pada pengujian 4500 *Thread* selama 300 detik dapat dilihat pada tabel 2.

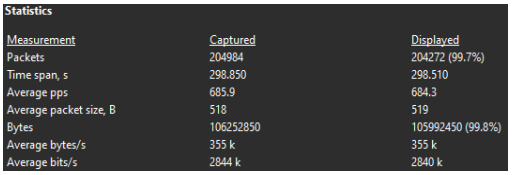
TABEL II
Hasil Perhitungan Qos 4500 Thread

Throughput	Jumlah Byte / Time Span = 95275050 x 8 = 762.200.400 / 298,497 = 2.553 bps
Packet loss	Paket dikirim - Paket diterima / Paket dikirim x 100% (173995- 170397) : 173995 x 100 (3558 : 173995) x 100 0,0204488635 x 100 = 2,04%
Delay	*Delay* Rata – Rata = 1,69 ms

Pada pengujian 4500 *Thread* dalam 300 detik, *web* "elearning.pnl.ac.id" masih dapat diakses dan berjalan normal. Dan berdasarkan Tabel II *web server* mampu menangani beban dengan baik.

3. Pengujian dengan 5000 *Thread*

   Hasil Pengujian 5000 *Thread* dengan waktu 300 detik dapat dilihat pada gambar 7.


Statistics

Measurement	Captured	Displayed
Packets	204984	204272 (99.7%)
Time span, s	298.850	298.510
Average pps	685.9	684.3
Average packet size, B	518	519
Bytes	106252850	105992450 (99.8%)
Average bytes/s	355 k	355 k
Average bits/s	2844 k	2840 k

Gbr 7 Hasil Pengujian 5000 *Thread*

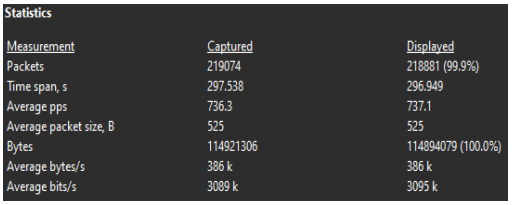
Dan berdasarkan gambar 7 dapat dihitung nilai *Quality of Service* seperti pada tabel 3.

TABEL III
Hasil Perhitungan Qos 5000 Thread

Throughput	Jumlah Byte / Time Span = 106252850 x 8 = 850.022.800 / 298,850 = 2.844 bps
Packet loss	Paket dikirim - Paket diterima / Paket dikirim x 100% (204984 - 204399) : 204984 x 100 (585 : 204984) x 100 0,0028538813 x 100 = 0,28%
Delay	*Delay* Rata – Rata = 1,45 ms

Berdasarkan hasil perhitungan dapat dinilai performa *web server* sangat baik dalam menangani beban yang dikirim sebesar 5000 *Thread* dalam waktu 300 detik sehingga web "elearning.pnl.ac.id" masih dapat diakses dengan normal tanpa ada kendala.

4. Pengujian dengan 5500 *Thread*
Hasil pengujian 5500 *Thread* pada wireshark dapat dilihat pada gambar 8.


Statistics

Measurement	Captured	Displayed
Packets	219074	218881 (99.9%)
Time span, s	297.538	296.949
Average pps	736.3	737.1
Average packet size, B	525	525
Bytes	114921306	114894079 (100.0%)
Average bytes/s	386 k	386 k
Average bits/s	3089 k	3095 k

Gbr 8 Hasil Pengujian 5500 *Thread*

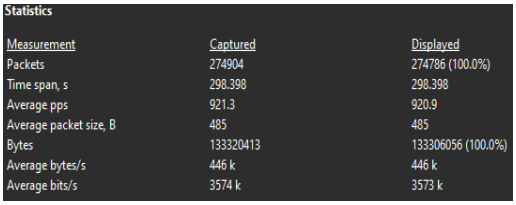
Berdasarkan hasil pengujian 5500 *Thread* dalam waktu 300 detik dapat dihitung *Quality of Service* pada *web server* di tabel 4.

TABEL IV
Hasil Perhitungan Qos 5500 Thread

Throughput	Jumlah Byte / Time Span = 114921306 x 8 = 919.380.448 / 298,850 = 3.076 bps
Packet loss	Paket dikirim - Paket diterima / Paket dikirim x 100% (219074 - 219066) : 219074 x 100 (8 : 219074) x 100 0,0000076088 x 100 = 0,00%
Delay	*Delay* Rata – Rata = 1,35 ms

Berdasarkan hasil perhitungan *web server* Politeknik Negeri Lhokseumawe dapat mengelola beban dengan baik. Dan saat proses *Stress Testing* 5500 *Thread* selama 300 detik, *web* "elearning.pnl.ac.id" masih dapat diakses dengan normal dan tanpa kendala.

5. Pengujian dengan 6000 *Thread*
Hasil pengujian 6000 *Thread* pada wireshark dapat dilihat pada gambar 9 dibawah.


Statistics

Measurement	Captured	Displayed
Packets	274904	274786 (100.0%)
Time span, s	298.398	298.398
Average pps	921.3	920.9
Average packet size, B	485	485
Bytes	133320413	133306056 (100.0%)
Average bytes/s	446 k	446 k
Average bits/s	3574 k	3573 k

Gbr 9 Hasil Pengujian 6000 *Thread*

Berdasarkan hasil pengujian 6000 *Thread* dalam waktu 300 detik dapat dihitung *Quality of Service* pada *web server* di tabel 5.

TABEL V
Hasil Perhitungan Qos 5000 Thread

Throughput	Jumlah Byte / Time Span = 133320413 x 8 = 1.066.563.304 / 298,398 = 3.574 bps
Packet loss	Paket dikirim - Paket diterima / Paket dikirim x 100% (274904 - 274886) : 274904 x 100 (18 : 274904) x 100 0,0000654774 x 100 = 0,00%
Delay	*Delay* Rata – Rata = 1,08 ms

Berdasarkan hasil perhitungan diatas dapat kita nilai *web server* Politeknik Negeri Lhokseumawe dapat mengelola beban dengan baik. Dan saat proses *Stress Testing* 6000 *Thread* selama 300 detik, *web* "elearning.pnl.ac.id" masih dapat diakses dengan normal dan tanpa kendala.

B. Performa *Web Server* Saat *Stress Testing*

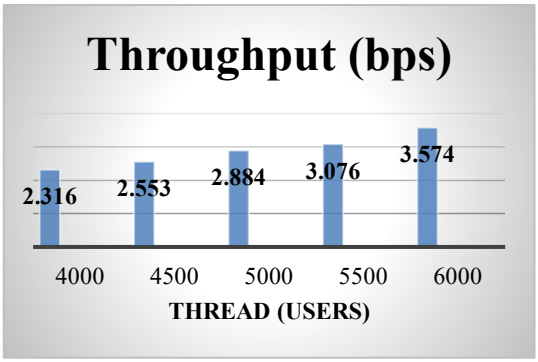
Performa *Web Server* saat *stress testing* dapat kita rangkum dari beban 4000, 4500, 5000, 5500, hingga 6000 *Thread* dalam jangka waktu 300 detik dapat kita rangkum pada tabel 6.

TABEL VI
Hasil Uji QoS 4000, 4500, 5000, 5500, dan 6000 *Thread*

Parameter	**4000**	**4500**	**5000**	**5500**	**6000**
***Throughput* (bps)**	2.316 bps	2.553 bps	2.884 bps	3.076 bps	3.574 bps
***Packet loss* (%)**	2,08%	2,04%	0,28%	0,00%	0,00%
***Delay* (ms)**	0,002 ms	0,002 ms	0,001 ms	0,001 ms	0,001 ms

1. Analisis Performa Berdasarkan *Throughput*

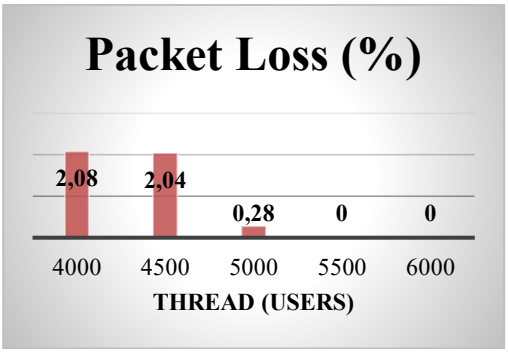
Berdasarkan gambar tabel 4.6 hasil uji QoS 4000, 4500, 5000, 5500, dan 6000 *Thread*, terlihat bahwa semakin banyak *thread* yang digunakan maka semakin tinggi nilai *throughput* yang didapatkan. *Throughput* menunjukkan jumlah data yang berhasil dikirimkan dalam waktu tertentu. Hasil uji *Throughput* dapat divisualisasikan pada gambar 10.



Gbr 10 Perbandingan Performa Berdasarkan *Throughput*

2. Analisis Performa Berdasarkan *Packet Loss*

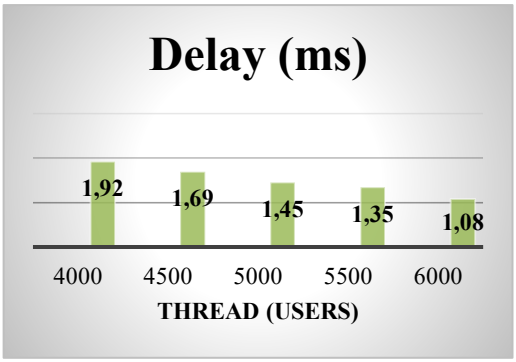
Packet loss menunjukkan tingkat kehilangan paket data yang dikirimkan. Semakin kecil nilai *packet loss*, semakin baik kualitas jaringan. Pada semua *thread* yang diuji, didapatkan nilai *packet loss* yang sangat kecil, yaitu antara 0,00% sampai 2,08%. Hasil uji *Packet loss* dapat di visualisasikan pada gambar grafik 11.



Gbr 11 Perbandingan Performa Berdasarkan *Packet Loss*

3. Analisis Performa Berdasarkan *Delay*

Nilai *delay* cenderung fluktuatif pada setiap *thread* yang diuji. *Thread* 4000 memiliki nilai *delay* sebesar 0.002 ms, *thread* 4500 memiliki nilai *delay* sebesar 0.002 ms, *thread* 5000 memiliki nilai *delay* sebesar 0.001 ms, *thread* 5500 memiliki nilai *delay* sebesar 0.001 ms, dan *thread* 6000 memiliki nilai *delay* sebesar 0,001 ms. Sebagai perbandingan dapat di visualisasikan pada gambar grafik 12.



Gbr 12 Perbandingan Performa Berdasarkan *Delay*

C. Performa *Web Server*

Berdasarkan hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa web *server* Politeknik Negeri Lhokseumawe menunjukkan performa yang sangat baik, bahkan ketika dihadapkan dengan beban kerja yang tinggi. Hal ini ditunjukkan oleh peningkatan *throughput* yang signifikan seiring dengan bertambahnya jumlah *thread* yang digunakan dalam pengujian. Pada beban 4000 *thread*, *throughput* yang dicapai adalah 2,316 bps, sementara pada beban 6000 *thread*, *throughput* meningkat tajam hingga mencapai 3,574 bps. Peningkatan *throughput* ini mengindikasikan bahwa *server* memiliki kapasitas yang memadai untuk menangani lebih banyak permintaan secara efektif.

Selain itu, tingkat *packet loss* yang terdeteksi selama pengujian juga sangat rendah. Pada pengujian dengan 4000 *thread*, *packet loss* tercatat sebesar 2,08%, dan pada 5000 *thread*, *packet loss* hanya sebesar 0,28%. Pada pengujian dengan 5000 dan 6000 *thread*, tidak ditemukan adanya *packet loss* sama sekali. Hal ini menunjukkan bahwa *server* mampu mengirimkan dan menerima hampir semua paket data dengan baik, sehingga kualitas layanan dapat terjaga dengan optimal.

Dari sisi *delay* atau *latency*, hasil pengujian menunjukkan bahwa *server* mampu memberikan waktu respons yang sangat cepat. Terdapat fluktuasi kecil dalam nilai *delay*, namun secara keseluruhan, nilai *delay* menurun seiring dengan meningkatnya jumlah *thread*. Pada 4000 *thread*, *delay* tercatat sebesar 1,922 ms, sementara pada 6000 *thread*, *delay* menurun hingga 1,08 ms. Waktu respons yang rendah ini merupakan indikasi positif bahwa *server* mampu menangani permintaan dengan cepat dan efisien, meskipun beban kerja meningkat.

Secara keseluruhan, hasil pengujian ini menunjukkan bahwa *web server* Politeknik Negeri Lhokseumawe memiliki performa yang baik dan dapat diandalkan dalam menghadapi beban kerja yang tinggi. *Server* mampu mempertahankan *throughput* yang tinggi, *packet loss* yang minimal, dan *delay* yang rendah, yang semuanya berkontribusi terhadap kualitas layanan yang optimal. Hal ini menunjukkan bahwa *server* memiliki kapasitas dan konfigurasi yang memadai untuk melayani kebutuhan pengguna dengan efisiensi dan kehandalan yang tinggi.

V. KESIMPULAN

Berdasarkan hasil pengujian yang dilakukan pada *web server* Politeknik Negeri Lhokseumawe, dapat disimpulkan bahwa *server* tersebut menunjukkan performa yang sangat baik saat dihadapkan dengan beban kerja yang tinggi. Beberapa poin penting yang dapat ditarik dari hasil pengujian adalah sebagai berikut:

1. Server mampu meningkatkan *throughput* secara signifikan seiring dengan bertambahnya jumlah *thread* yang digunakan dalam pengujian. Pada puncaknya dengan 6000 *thread*, *throughput* mencapai 3,574 bps, menunjukkan kapasitas *server* yang memadai untuk menangani permintaan dengan efektif.
2. Tingkat *packet loss* selama pengujian sangat rendah, bahkan pada beban tertinggi sekalipun (5000 dan 6000 *thread*), tidak ditemukan adanya *packet loss*.
3. Nilai *delay* menunjukkan penurunan seiring dengan peningkatan jumlah *thread*, dengan nilai terendah mencapai 1,08 ms pada 6000 *thread*. Hal ini menunjukkan efisiensi server dalam menanggapi permintaan pengguna dengan cepat.

REFERENSI

[1] F. Fachri, A. Fadlil, and I. Riadi, **Analisis Keamanan Webserver menggunakan Penetration Test** *Jurnal Informatika*, vol. 8, no. 2, pp. 183–190, Aug. 2021, doi: 10.31294/ji.v8i2.10854.

[2] **Internet-TCPIP Konsep Dan Implementasi** (httpslms.onnocenter.or.idpustakaREVIEW-BUKU2018-Internet TCPIP%20Konsep%20Dan%20Implementasi.pdf). pdf.

[3] Z. M. Luthfansa and U. D. Rosiani, **Pemanfaatan Wireshark untuk Sniffing Komunikasi Data Berprotokol HTTP pada Jaringan Internet** *JIEET*, vol. 5, no. 1, pp. 34–39, Jun. 2021, doi: 10.26740/jieet.v5n1.p34-39.

[4] R. M. Farhan and G. H. A. Kusuma, **Teknik Sniffing Jaringan Menggunakan Wireshark** vol. 4, no. 1, 2023.

[5] M. Reza Maulana, E. Budi Susanto, and S. Sattriedi, **Analisis Kinerja Website Pemerintah Kota Pekalongan** *JLKP*, vol. 20, Jun. 2021, doi: 10.54911/litbang.v20i.144.

[6] N. Rahmalia, **Stress Testing: Definisi, Tipe, dan Tools yang Digunakan** Glints Blog. Accessed: Aug. 14, 2024. [Online]. Available: https://glints.com/id/lowongan/stress-testing-adalah/

[7] P. R. Utami, **Analisis Perbandingan Quality Of Service Jaringan Internet Berbasis Wireless Pada Layanan Internet Service Provider (Isp) Indihome Dan First Media** *tekno*, vol. 25, no. 2, pp. 125–137, 2020, doi: 10.35760/tr.2020.v25i2.2723.

[8] M. Hasbi and N. R. Saputra, **Analisis Quality Of Service (Qos) Jaringan Internet Kantor Pusat King Bukopin Dengan Menggunakan Wireshark**.

[9] **Performa Dan Stresstesting dalam Upaya Mengoptimalkan Webgis Opensource Studi Kasus WebGIS Ekowisata Sungai Mudal Kulon Progo**.pdf.