

BAB 5 PENGUJIAN DAN ANALISIS

5.1 Pengujian

Pada bagian ini dijelaskan hasil pengujian terhadap implementasi yang telah dilakukan beserta analisisnya. Pengujian dilakukan untuk menilai apakah seluruh kebutuhan yang telah dispesifikasikan sebelumnya telah terpenuhi. Analisis dilakukan untuk menilai apakah sistem bekerja sesuai yang diharapkan dan sebagai acuan untuk menarik kesimpulan dari penelitian ini.

5.2 Pengujian Fungsionalitas

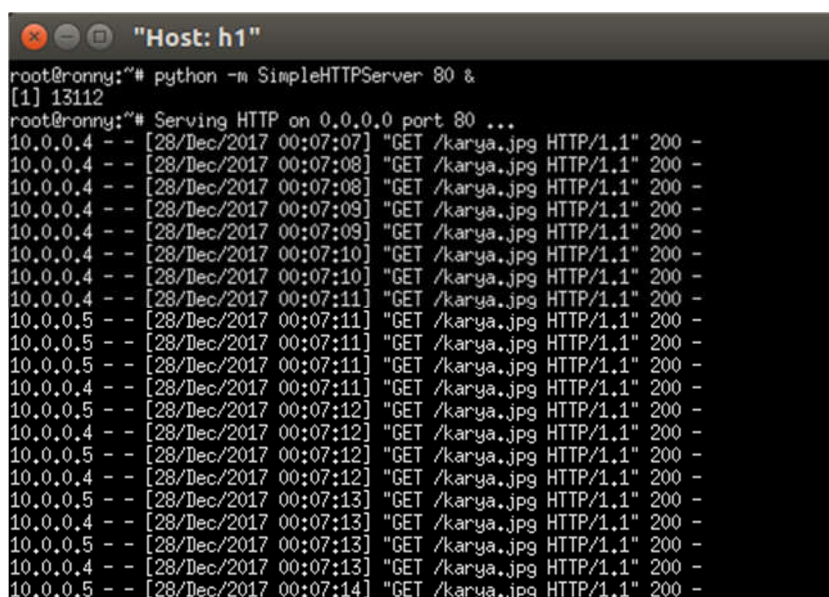
5.2.1 Tujuan Pengujian Fungsionalitas

Kebutuhan fungsionalitas merupakan kebutuhan yang dapat dilakukan oleh sistem. Pada penelitian ini dilakukan dengan membandingkan algoritme *shortest delay* dengan algoritme *round robin* dalam *mapping server*. *Mapping server* dilakukan untuk mengetahui kinerja server dalam menghandle banyaknya permintaan dari client.

5.2.2 Skenario Pengujian Fungsionalitas

Untuk mengukur *fungsionalitas*, Pengujian ini dilakukan untuk membandingkan antara algoritme *shortest delay* dan algoritme *round-robin* Pada Pengujian ini, h1, h2,h3 sebagai *server* dan h4 sampai h7, h11 sampai h18 sebagai *client*. Pengujian *fungsionalitas* ini dilakukan dengan beban traffic TCP selama 1000 detik di salah satu host, seperti h8 dan host lainnya dilakukan dengan request secara bersamaan. Pengujian ini digunakan untuk melihat server dalam menghandle request dari beberapa client yang diuji untuk mengetahui performa dari *mapping* tiap server dalam menangani banyak permintaan dari client.

5.2.3 Hasil Pengujian Fungsionalitas



```
root@ronny:~# python -m SimpleHTTPServer 80 &
[1] 13112
root@ronny:~# Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.4 - - [28/Dec/2017 00:07:07] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:08] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:08] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:09] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:09] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:10] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:10] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:11] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:11] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:11] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:11] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:11] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:12] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:12] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:12] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:12] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:13] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:13] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:13] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.4 - - [28/Dec/2017 00:07:13] "GET /karya.jpg HTTP/1.1" 200 -
10.0.0.5 - - [28/Dec/2017 00:07:14] "GET /karya.jpg HTTP/1.1" 200 -
```

Gambar 5. 1 Pengujian Mapping Server 48 requet/second pada host 1

Pada gambar diatas menunjukkan contoh *mapping server* pada 1 server saja yang ditampilkan pada gambar 5.1. Berikut hasil tabel perbandingan *mapping server* antara algoritme *shortest delay* dan algoritme *round robin*. Terdapat perbedaan pada tiap server dalam menangani *request* yang telah dibuat.

Tabel 5. 1 Perbandingan Mapping Server Dua Algoritme

Server yang di uji	Shortest Delay	Round robin
Server 1 (10.0.0.1)	105	76
Server 2 (10.0.0.2)	0	76
Server 3 (10.0.0.3)	141	76

Pada tabel 5.1 menunjukkan perbandingan algoritme *shortest delay* dan algoritme *round-robin*. Gambar di atas dilakukan pada pengujian *mapping server* dengan 48 request/second. Pada tabel tersebut dapat dilihat tiap server dalam menangani banyaknya permintaan dari client.

5.2.4 Analisis Pengujian Fungsionalitas

Berdasarkan skenario *fungsionalitas* yang telah dilakukan, hasil analisis perbandingan algoritme *shortest delay* dan algoritme *round robin* terdapat perbedaan dalam *mapping request* ke tiap server. Pada pengujian ini dilakukan dengan pemberian beban *traffic* pada dua algoritme tersebut. Pada algoritme *round-robin* setiap server mendapatkan request yang sama sebanyak 76 *request/second*. Pada algoritme *shortest delay* setiap server mendapatkan request berbeda. Pada server 2 nilai bernilai 0, hal ini dikarenakan pada server 2 diberikan *beban traffic*.

Dari pengujian yang telah dilakukan bahwa algoritme *shortest delay* dalam *menghandle request* dari client lebih bagus dibandingkan algoritme *round-robin* karena algoritme *shortest delay* tidak akan memilih jalur ke server yang terdapat beban *traffic* dan akan memilih jalur ke server lainnya, algoritme ini mengalokasikan koneksi ke server dengan jalur *delay* terendah. Sedangkan algoritme *round-robin* akan tetap melewati jalur yang telah diberikan beban *traffic* karna algoritme tersebut tidak memperhatikan *disturbance conditions* dan tetap akan memilih jalur ke server dengan cara berurutan.

5.3 Pengujian Throughput

5.3.1 Tujuan Pengujian Throughput

Throughput menyatakan jumlah jumlah bit yang berlalu pada suatu link per satuan waktu. Pengujian ini dilakukan untuk membandingkan *throughput* antara algoritme *shortest delay* dan algoritme *round-robin*. Selanjutnya hasil dari percobaan dari *algoritme shortest delay* dan *algoritme round robin* akan dibandingkan dan dilakukan analisis dari hasil yang didapat. Sehingga dapat diketahui performa dari web server dalam melakukan respon terhadap permintaan client.

5.3.2 Skenario Pengujian Throughput

Untuk mengukur *throughput*, digunakan tool *httperf* dengan command “*httperf --server=[ip_server] --port=80 --uri=[directory yg di tuju] --num--conns=[jumlah koneksi] --rate=[rate koneksi]*” di sisi client. Pengujian ini dilakukan untuk membandingkan *throughput* antara algoritme *shortest delay* dan algoritme *round-robin* Pada Pengujian ini, h1, h2,h3 sebagai *server* dan h4 sampai h7, h11 sampai h18 sebagai *client*. Pengujian *throughput* ini dilakukan dengan memberikan beban traffic TCP selama 1000 detik di salah satu host, seperti *h8* dan host lainnya dilakukan dengan request secara bersamaan. Gambar *command-line* di sisi client adalah sebagai berikut:



Gambar 5. 2 Command-line pengujian httperf di host 4

Command-line pada gambar 5.2 diatas dijelaskan tool yang digunakan untuk melakukan pengujian load balancing dengan menggunakan *#httperf* dengan alamat web server yang diakses pada client 10.0.0.20. Port yang digunakan untuk komunikasi antara client server adalah *port 80*. file yang diakses oleh *httperf*, menggunakan *picture* dengan format *jpg* yang diberi nama “*karya.jpg*” pada *resource (uri)* serta banyaknya permintaan yang akan dilakukan ke server dan juga jumlah koneksi yang akan dibuat dalam satu waktu.

5.3.3 Hasil Pengujian Throughput

Throughput berdasarkan manual *httperf* dapat di ambil dari NET I/O yang merupakan rata-rata *throughput* jaringan yang mempunyai satuan KB (*kilobytes*) per detik dan MB (*megabits*) per detik.

```

Host: h4"
root@ronny:~# httpperf --server=10.0.0.20 --port=80 --uri=/karya.jpg --num-conns
=60 --rate=6
httpperf --client=0/1 --server=10.0.0.20 --port=80 --uri=/karya.jpg --rate=6 --se
nd-buffer=4096 --recv-buffer=16384 --ssl-protocol=auto --num-conns=60 --num-call
s=1
Maximum connect burst length: 1

Total: connections 60 requests 60 replies 60 test-duration 9.920 s

Connection rate: 6.0 conn/s (165.3 ms/conn, <=1 concurrent connections)
Connection time [ms]: min 86.4 avg 87.3 max 131.5 median 86.5 stddev 5.8
Connection time [ms]: connect 0.8
Connection length [replies/conn]: 1.000

Request rate: 6.0 req/s (165.3 ms/req)
Request size [B]: 71.0

Reply rate [replies/s]: min 6.0 avg 6.0 max 6.0 stddev 0.0 (1 samples)
Reply time [ms]: response 0.4 transfer 86.1
Reply size [B]: header 191.0 content 1030958.0 footer 0.0 (total 1031149.0)
Reply status: 1xx=0 2xx=60 3xx=0 4xx=0 5xx=0

CPU time [s]: user 7.37 system 2.55 (user 74.3% system 25.7% total 100.0%)
Net I/O: 6091.2 KB/s (49.9*10^6 bps)

Errors: total 0 client-time 0 socket-time 0 connrefused 0 connreset 0
Errors: fd-unavail 0 addrunavail 0 ftab-full 0 other 0
root@ronny:~#

```

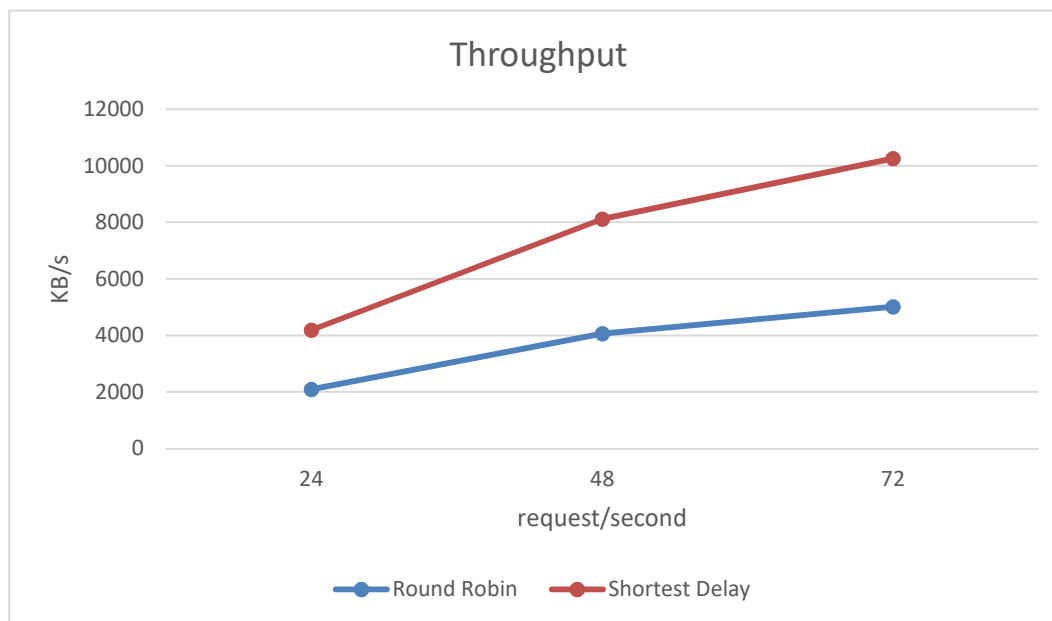
Gambar 5. 3 Pengujian Throughput 72 request/second pada host 4

Pada Gambar diatas menunjukkan contoh *throughput* dari pengujian pada 1 host saja yang ditampilkan pada gambar 5.3. Berikut hasil tabel pengujian *throughput* dengan membandingkan algoritme *shortest delay* dan algoritme *round robin*. Terdapat perbedaan hasil *throughput* pada masing-masing request yang telah dibuat.

Tabel 5. 2 Perbandingan Throughput Dua Algoritme

Request	Throughput (KB/s)	
	Shortest Delay	Round Robin
24	2096,52	2093,73
48	4062,10	4060,52
72	5242,74	5019,15
Rata-rata	3800,45	3724,46

Pada Tabel 5.2 di atas menunjukkan perbandingan algoritme *shortest delay* dan algoritme *round robin*. Pada tabel tersebut juga terdapat peningkatan nilai *throughput* dari algoritme *shortest delay* dibandingkan algoritme *round robin*. Perhitungan nilai *throughput* dilakukan untuk mengetahui kecepatan transfer paket antar *client-server*.



Gambar 5. 4 Grafik Pengujian *Throughput* Dua Algoritme

Nilai *throughput* menunjukkan perbandingan algoritme *shortest delay* dan algoritme *round robin* yang telah dilakukan pada gambar 5.4. Nilai *throughput* algoritme *shortest delay* lebih besar dibandingkan algoritme *round-robin* untuk 24 sampai 72 *request/second*. Hal ini bisa dilihat dari grafik perbandingan *throughput* pada gambar 5.4, dimana grafik yang dihasilkan bertambah naik dan cenderung stabil, sama halnya dengan algoritme *round robin*. Perubahan jumlah *request* mengakibatkan nilai dari *throughput* meningkat sesuai dengan jumlah *request* yang semakin banyak yang diterima oleh web server.

5.3.4 Analisis Pengujian *Throughput*

Berdasarkan hasil pengujian *throughput* pada algoritme *shortest delay*, dapat diketahui dari *request* yang dilakukan nilai *throughput* yang diperoleh menunjukkan 2096,52 KB/s pada 24 *request/second* dan kemudian mengalami kenaikan dengan banyaknya *request* oleh user. Semakin banyaknya *request* yang dilakukan maka nilai *throughput* akan semakin besar dengan nilai rata-rata 3800,45 KB/s. Hal ini disebabkan perubahan jumlah *request* berpengaruh pada nilai *throughput* yang otomatis akan meningkat dengan jumlah *request* yang semakin banyak yang diterima oleh web server.

Ketika pengujian *throughput* pada algoritme *round robin*, dapat diketahui dari *request* yang dilakukan nilai *throughput* yang diperoleh menunjukkan 2093,73 KB/s pada request 24 *request/second* dan kemudian mengalami kenaikan dengan banyaknya *request* per detik oleh user. Semakin banyaknya *request* yang

dilakukan maka nilai *throughput* akan semakin besar dengan nilai rata-rata 3724,46 KB/s. Sama halnya algoritme *shortest delay*, Hal ini disebabkan perubahan jumlah request berpengaruh pada nilai throughput yang otomatis akan meningkat dengan jumlah *request* yang semakin banyak yang diterima oleh web server.

Pada pengujian dua algoritme tersebut terjadi peningkatan nilai yang signifikan, hal ini disebabkan jalur yang digunakan untuk pengiriman data lebih banyak karena banyaknya request dari user. Sehingga semakin besar nilai parameter ini, maka semakin baik pula kinerja dari web server tetapi dalam pengujian *throughput* yang telah dilakukan menunjukkan kinerja algoritme *shortest delay* lebih baik dibandingkan algoritme *round robin*.

5.4 Pengujian Connection Rate

5.4.1 Tujuan Pengujian Connection Rate

Connection rate berdasarkan banyaknya koneksi yang dapat dilakukan oleh user dalam satu waktu yang mempunyai satuan *conn/s*. Pengujian ini dilakukan untuk membandingkan *connection rate* antara algoritme *shortest delay* dan algoritme *round-robin*. Selanjutnya hasil dari percobaan dari algoritme *shortest delay* dan algoritme *round robin* akan dibandingkan dan dilakukan analisis dari hasil yang didapat. Sehingga dapat diketahui performa dari web server dalam melakukan respon terhadap permintaan client.

5.4.2 Skenario Pengujian Connection Rate

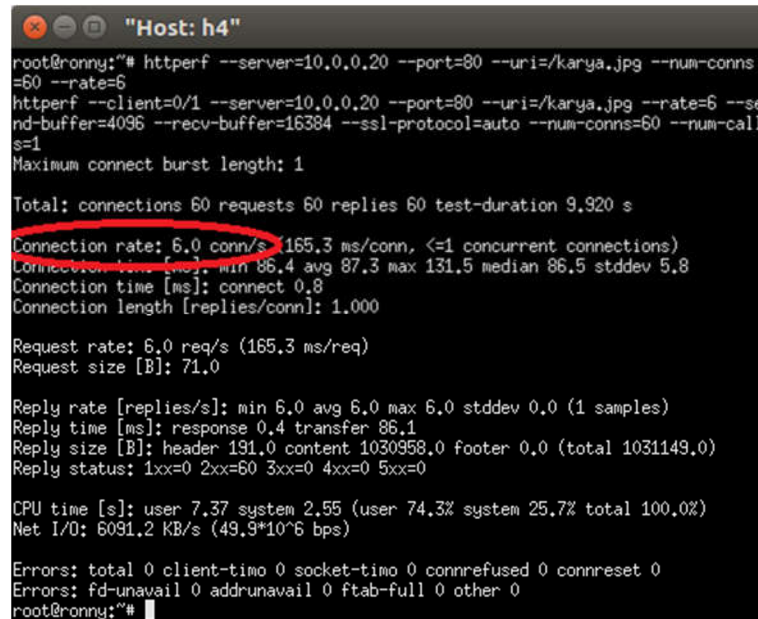
Untuk mengukur *Connection rate*, digunakan tool *httperf* dengan command "*httperf --server=[ip_server] --port=80 --uri=[directory yg di tuju] --num--conns=[jumlah koneksi] --rate=[rate koneksi]*" di sisi client. Pengujian ini dilakukan untuk membandingkan *connection rate* antara algoritme *shortest delay* dan algoritme *round-robin*. Pada Pengujian ini, h1, h2,h3 sebagai *server* dan h4 sampai h7, h11 sampai h18 sebagai *client*. Pengujian *connection rate* ini dilakukan dengan memberikan beban traffic TCP selama 1000 detik di salah satu host, seperti h8 dan host lainnya dilakukan dengan request secara bersamaan. Gambar *command-line* di sisi client adalah sebagai berikut:



```
root@ronny:~# httperf --server=10.0.0.20 --port=80 --uri=/karya.jpg --num-conns=60 --rate=6
```

Commend-line diatas dijelaskan tool yang digunakan untuk melakukan pengujian load balancing dengan menggunakan *#httperf* dengan alamat web server yang diakses pada client 10.0.0.20. Port yang digunakan untuk komunikasi antara client server adalah *port 80*. file yang diakses oleh *httperf*, menggunakan *picture* dengan format *jpg* yang diberi nama "*karya.jpg*" pada *resource (uri)* serta banyaknya permintaan yang akan dilakukan ke server dan juga jumlah koneksi yang akan dibuat dalam satu waktu.

5.4.3 Hasil Pengujian Connection Rate



```

root@ronny:~# httpperf --server=10.0.0.20 --port=80 --uri=/karya.jpg --num-conns=60 --rate=6
httpperf --client=0/1 --server=10.0.0.20 --port=80 --uri=/karya.jpg --rate=6 --send-buffer=4096 --recv-buffer=16384 --ssl-protocol=auto --num-conns=60 --num-calls=1
Maximum connect burst length: 1

Total: connections 60 requests 60 replies 60 test-duration 9.920 s

Connection rate: 6.0 conn/s (165.3 ms/conn, <=1 concurrent connections)
Connection time [ms]: min 86.4 avg 87.3 max 131.5 median 86.5 stddev 5.8
Connection time [ms]: connect 0.8
Connection length [replies/conn]: 1.000

Request rate: 6.0 req/s (165.3 ms/req)
Request size [B]: 71.0

Reply rate [replies/s]: min 6.0 avg 6.0 max 6.0 stddev 0.0 (1 samples)
Reply time [ms]: response 0.4 transfer 86.1
Reply size [B]: header 191.0 content 1030958.0 footer 0.0 (total 1031149.0)
Reply status: 1xx=0 2xx=60 3xx=0 4xx=0 5xx=0

CPU time [s]: user 7.37 system 2.55 (user 74.3% system 25.7% total 100.0%)
Net I/O: 6091.2 KB/s (49.9*10^6 bps)

Errors: total 0 client-time 0 socket-time 0 connrefused 0 connreset 0
Errors: fd-unavail 0 addrunavail 0 ftab-full 0 other 0
root@ronny:~#

```

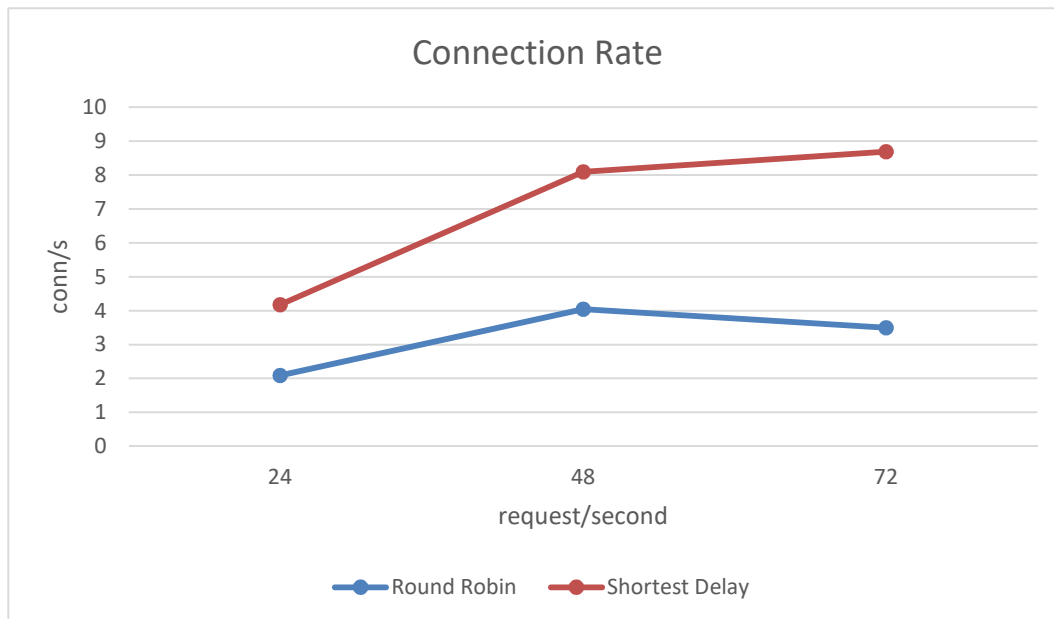
Gambar 5. 5 Pengujian Connection Rate 72 request/second pada host 4

Pada Gambar diatas menunjukkan contoh *connection rate* dari pengujian pada 1 host saja yang ditampilkan pada gambar 5.5. Berikut hasil tabel pengujian *connection rate* dengan membandingkan algoritme *shortest delay* dan algoritme *round robin*. Terdapat perbedaan hasil *connection rate* pada masing-masing request yang telah dibuat.

Tabel 5. 3 Perbandingan Connection Rate Dua Algoritme

Request	Connection rate (conn/s)	
	Shortest Delay	Round Robin
24	2,09	2,09
48	4,05	4,04
72	5,19	4,98
Rata-rata	3,78	3,70

Pada Tabel 5.3 di atas menunjukkan perbandingan algoritme *shortest delay* dan *round robin*. Pada tabel tersebut nilai *connection rate* terdapat peningkatan dengan bertambahnya request dari algoritme *shortest delay* dibandingkan algoritme *round robin*. Perhitungan *connection rate* dilakukan untuk mengetahui banyaknya koneksi yang dapat dilakukan oleh user.



Gambar 5. 6 Grafik pengujian *connection rate* dua algoritme

Dari hasil analisis gambar 5.6 di atas, perbandingan antara dua algoritme menunjukkan nilai *connection rate* dari algoritme *shortest delay* lebih besar dibandingkan algoritme *round robin* untuk 24 sampai 72 *request/second*. Hal ini bisa dilihat dari grafik perbandingan *connection rate* pada gambar 5.6, dimana grafik yang dihasilkan cenderung stabil, sama halnya dengan algoritme *round robin*. Perubahan jumlah *request* mengakibatkan nilai dari *connection rate* meningkat sesuai dengan jumlah *request* yang semakin banyak yang diterima oleh web server.

5.4.4 Analisis Pengujian Connection Rate

Berdasarkan hasil pengujian *connection rate* pada algoritme *shortest delay*, dapat diketahui dari *request* yang dilakukan nilai *connection rate* yang diperoleh menunjukkan 2,09 *conn/s* pada 24 *request/second* dan kemudian mengalami kenaikan dengan banyaknya *request* oleh user. Semakin banyaknya *request* yang dilakukan maka nilai *reply time* akan semakin besar dengan nilai rata-rata 3,78 *conn/s*. Hal ini disebabkan banyaknya koneksi sangat berpengaruh pada nilai *connection rate* yang otomatis akan meningkat dengan jumlah koneksi yang semakin banyak yang diterima oleh web server.

Ketika pengujian *connection rate* pada algoritme *round robin*, dapat diketahui dari *request* yang dilakukan nilai *connection rate* yang diperoleh menunjukkan 2,09 *conn/s* pada request 24 *request/second* dan kemudian mengalami kenaikan dengan banyaknya request per detik oleh user. Semakin banyaknya *request* yang dilakukan maka nilai *reply time* akan semakin besar dengan nilai rata-rata 3,70 *conn/s*. Sama halnya algortima *shortest delay*, Hal ini disebabkan banyaknya koneksi sangat berpengaruh pada nilai *connection rate*

yang otomatis akan meningkat dengan jumlah koneksi yang semakin banyak yang diterima oleh web server.

Pada pengujian dua algoritme yang telah dilakukan, algoritme *shortest delay* lebih baik dibandingkan algoritme *round robin* dengan banyaknya koneksi yang dapat dilakukan oleh user maka semakin baik tetapi jika semakin kecil semakin buruk pada kualitas sistem walaupun nilai hasil analisis *connection rate* yang dihasilkan tidak signifikan.

5.5 Pengujian Reply Time

5.5.1 Tujuan Pengujian Reply Time

Reply time menggambarkan response dari web server dalam menanggapi request dari client yang mempunyai satuan *millisecond*. Pengujian ini dilakukan untuk membandingkan *reply time* antara algoritme *shortest delay* dan algoritme *round-robin*. Selanjutnya hasil dari percobaan dari *algoritme shortest delay* dan *algoritme round robin* akan dibandingkan dan dilakukan analisis dari hasil yang didapat. Sehingga dapat diketahui performa dari web server dalam melakukan respon terhadap permintaan client.

5.5.2 Skenario Pengujian Reply Time

Untuk mengukur *reply time*, digunakan tool *httperf* dengan command “*httperf --server=[ip_server] --port=80 --uri=[directory yg di tuju] --num--conns=[jumlah koneksi] --rate=[rate koneksi]*” di sisi client. Pengujian ini dilakukan untuk membandingkan *reply time* antara algoritme *shortest delay* dan algoritme *round-robin*. Pada Pengujian ini, h1, h2,h3 sebagai *server* dan h4 sampai h7, h11 sampai h18 sebagai *client*. Pengujian *reply time* ini dilakukan dengan memberikan beban traffic TCP selama 1000 detik di salah satu host, seperti *h8* dan host lainnya dilakukan dengan request secara bersamaan. Gambar *command-line* di sisi client adalah sebagai berikut:



```
root@ronny:~# httperf --server=10.0.0.20 --port=80 --uri=/karya.jpg --num-conns=60 --rate=60
```

Commend-line diatas dijelaskan tool yang digunakan untuk melakukan pengujian load balancing dengan menggunakan *#httperf* dengan alamat web server yang diakses pada client 10.0.0.20. Port yang digunakan untuk komunikasi antara client server adalah *port 80*. file yang diakses oleh *httperf*, menggunakan *picture* dengan format *jpg* yang diberi nama “*karya.jpg*” pada *resource (uri)* serta banyaknya permintaan yang akan dilakukan ke server dan juga jumlah koneksi yang akan dibuat dalam satu waktu.

5.5.3 Hasil Pengujian Reply Time

```

Host: h4"
root@ronny:~# httpperf --server=10.0.0.20 --port=80 --uri=/karya.jpg --num-conns
=60 --rate=6
httpperf --client=0/1 --server=10.0.0.20 --port=80 --uri=/karya.jpg --rate=6 --se
nd-buffer=4096 --recv-buffer=16384 --ssl-protocol=auto --num-conns=60 --num-call
s=1
Maximum connect burst length: 1

Total: connections 60 requests 60 replies 60 test-duration 9.920 s

Connection rate: 6.0 conn/s (165.3 ms/conn, <=1 concurrent connections)
Connection time [ms]: min 86.4 avg 87.3 max 131.5 median 86.5 stddev 5.8
Connection time [ms]: connect 0.8
Connection length [replies/conn]: 1.000

Request rate: 6.0 req/s (165.3 ms/req)
Request size [B]: 71.0

Reply rate [replies/s]: min 6.0 avg 6.0 max 6.0 stddev 0.0 (1 samples)
Reply time [ms]: response 0.4 transfer 86.1
Reply size [B]: header 131.0 content 1030958.0 footer 0.0 (total 1031149.0)
Reply status: 1xx=0 2xx=60 3xx=0 4xx=0 5xx=0

CPU time [s]: user 7.37 system 2.55 (user 74.3% system 25.7% total 100.0%)
Net I/O: 6091.2 KB/s (49.9*10^6 bps)

Errors: total 0 client-timo 0 socket-timo 0 connrefused 0 connreset 0
Errors: fd-unavail 0 addrunavail 0 ftab-full 0 other 0
root@ronny:~#

```

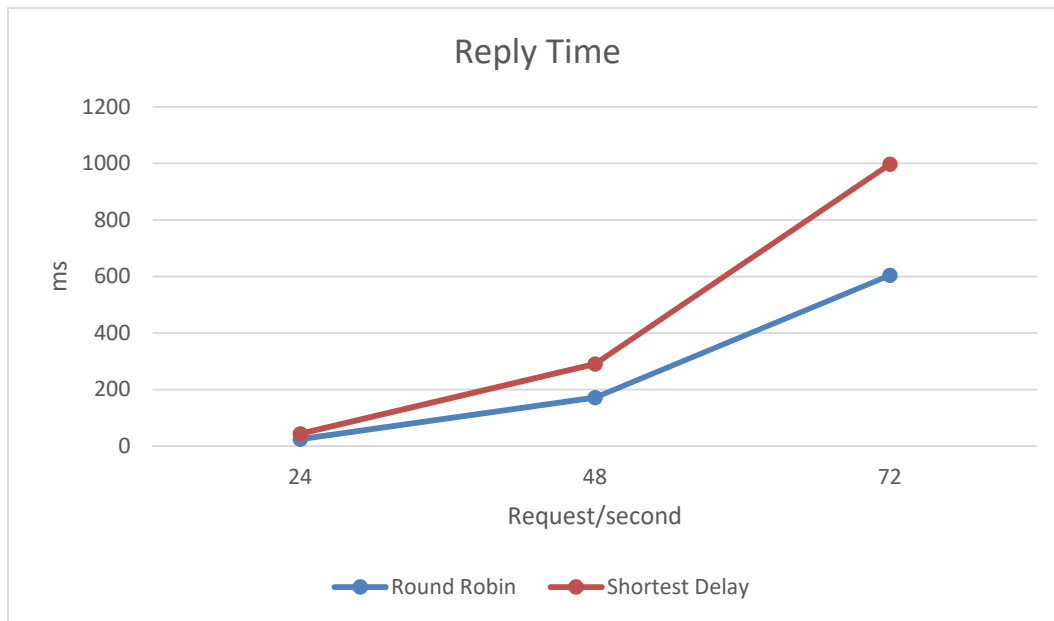
Gambar 5. 7 Pengujian Connection Rate 72 request/second pada host 4

Pada Gambar diatas menunjukkan contoh *reply time* dari pengujian pada 1 host saja yang ditampilkan pada gambar 5.7. *Reply time* didapatkan dari *Reply Section*. Berikut hasil tabel pengujian *reply time* dengan membandingkan algoritme *shortest delay* dan algoritme *round robin*. Terdapat perbedaan hasil *reply time* pada masing-masing request yang telah dibuat.

Tabel 5. 4 Perbandingan Reply Time Dua Algoritme

Request	Reply time (milisecond)	
	Shortest Delay	Round Robin
24	20,29	23,61
48	118,4	171,34
72	393,69	603,72
Rata-rata	177,46	266,22

Pada Tabel 5.4 di atas menunjukkan perbandingan algoritme *shortest delay* dan *round robin*. Pada tabel tersebut terjadi peningkatan *reply time* dari algoritme *shortest delay* lebih dibandingkan algoritme *round robin*. Perhitungan *reply time* dilakukan untuk mengetahui kecepatan web server dalam menanggapi request dari client.



Gambar 5. 8 Pengujian Reply Time Dua Algoritme

Pada gambar 5.8 di atas, Perubahan nilai signifikan terjadi di awal percobaan. Pergerakan grafik di mulai dari 24 *request/second* dengan nilai response sangat rendah berarti pada response time pada 24 *request/second* lebih baik dan cepat. Hal ini disebabkan batas kemampuan web server dalam menangani 24 *request/second*. Pergerakan grafik pada 48 *request/second* sampai 72 *request/second* mengalami peningkatan yang signifikan. Hal tersebut karena server banyaknya antrian request dari client dan banyaknya koneksi juga berpengaruh dalam menanggapi permintaan client.

5.5.4 Analisis Pengujian Reply time

Berdasarkan hasil pengujian *reply time* pada algoritme *shortest delay*, dapat diketahui dari *request* yang dilakukan nilai *reply time* yang diperoleh sangat kecil menunjukkan 20,29 ms pada 24 *request/second* disebabkan seluruh paket dilayani dengan baik oleh web server. Selanjutnya mengalami kenaikan dengan 48 *request/second* sampai 72 *request/second* dan cenderung meningkat dengan bertambahnya request oleh user. Semakin banyaknya *request* yang dilakukan maka nilai *reply time* akan semakin besar dengan nilai rata-rata 177,46 ms. Hal ini disebabkan perubahan jumlah request berpengaruh pada nilai *reply time* dalam menanggapi banyak *request* dari user.

Ketika pengujian *reply time* pada algoritme *round robin*, dapat diketahui dari *request* yang dilakukan nilai *reply time* yang diperoleh menunjukkan 23,61 ms pada request 24 *request/second* dan kemudian mengalami kenaikan dengan banyaknya request per detik oleh user. Semakin banyaknya *request* yang dilakukan maka nilai *reply time* akan semakin besar dengan nilai rata-rata 266,22 ms. Sama halnya dengan algoritme *shortest delay*, hal ini disebabkan perubahan jumlah request berpengaruh pada nilai *reply time* dalam menanggapi banyak *request* dari user.

Pada pengujian dua algoritme ini terjadi peningkatan nilai yang signifikan dari awal percobaan, Hal ini disebabkan server mengalami lonjakan sehingga mengakibatkan nilai response time menjadi tinggi dan cenderung stabil, karena server mampu beradaptasi dengan banyaknya request yang telah diberikan. Sehingga semakin kecil nilai dari parameter ini, maka semakin cepat sebuah web server dalam menanggapi *request* dari client. Selanjutnya hasil analisis *reply time* yang telah dilakukan menunjukkan kinerja algoritme *shortest delay* lebih baik dibandingkan algoritme *round robin*.