

I. Hasil Testing

a. Algoritma Round Robin

```
root@Revolte:~# cat result.txt
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.57.2.2 (be patient)

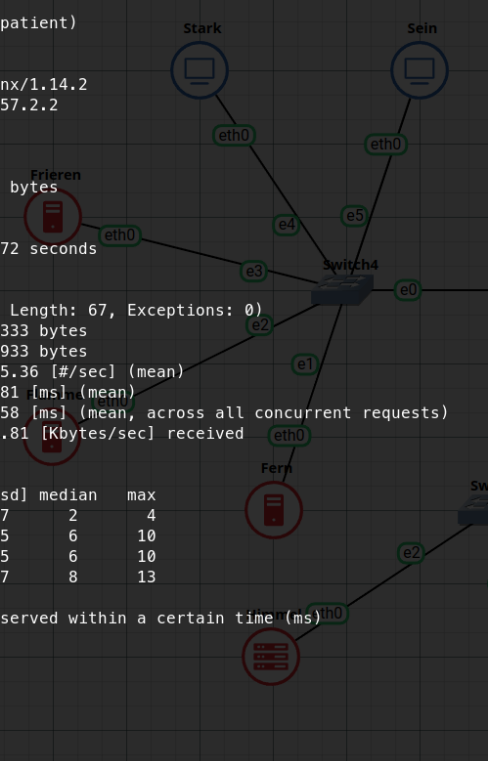
Server Software:      nginx/1.14.2
Server Hostname:      10.57.2.2
Server Port:          80

Document Path:        /
Document Length:      625 bytes

Concurrency Level:    10
Time taken for tests:  0.172 seconds
Complete requests:    200
Failed requests:       67
  (Connect: 0, Receive: 0, Length: 67, Exceptions: 0)
Total transferred:    152333 bytes
HTML transferred:     124933 bytes
Requests per second:  1165.36 [#/sec] (mean)
Time per request:     8.581 [ms] (mean)
Time per request:     0.858 [ms] (mean, across all concurrent requests)
Transfer rate:        866.81 [Kbytes/sec] received

Connection Times (ms)
  min   mean[+/-sd] median   max
Connect:  0       2   0.7      2      4
Processing: 2       7   1.5      6     10
Waiting:  2       7   1.5      6     10
Total:    3       8   1.7      8     13

Percentage of the requests served within a certain time (ms)
 50%    8
 66%    9
 75%   10
 80%   10
 90%   11
 95%   11
 98%   12
 99%   13
100%   13 (longest request)
```



b. Algoritma Least Connection

```
root@Revolt:~# cat result.txt
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.57.2.2 (be patient)

+-----+
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
|      |      |      |      |      |      |      |      |      |      |
+-----+

Server Software:      nginx/1.14.2
Server Hostname:      10.57.2.2
Server Port:          80

Document Path:        /
Document Length:      625 bytes

Concurrency Level:    10
Time taken for tests:  0.152 seconds
Complete requests:    200
Failed requests:      67
  (Connect: 0, Receive: 0, Length: 67, Exceptions: 0)
Total transferred:    152333 bytes
HTML transferred:     124933 bytes
Requests per second:  1312.02 [#/sec] (mean)
Time per request:     7.622 [ms] (mean)
Time per request:     0.762 [ms] (mean, across all concurrent requests)
Transfer rate:        975.90 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0    2  0.6      2    4
Processing:  2    6  1.0      6    9
Waiting:    1    6  1.0      6    9
Total:      2    7  1.3      7   11

Percentage of the requests served within a certain time (ms)
 50%    7
 66%    8
 75%    8
 80%    8
 90%    9
 95%   10
 98%   11
 99%   11
100%   11 (longest request)
```

c. IP Hash

```
Finished 200 requests
root@Revolte:~# cat result.txt
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.57.2.2 (be patient)

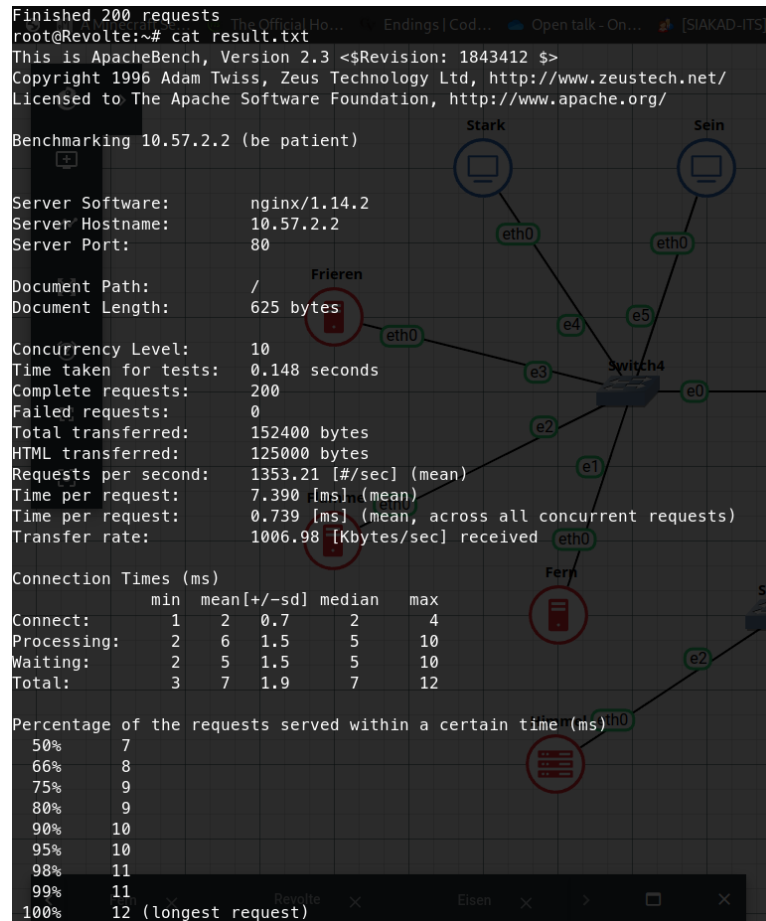
Server Software:      nginx/1.14.2
Server Hostname:      10.57.2.2
Server Port:          80

Document Path:        /
Document Length:       625 bytes

Concurrency Level:     10
Time taken for tests:   0.148 seconds
Complete requests:     200
Failed requests:        0
Total transferred:     152400 bytes
HTML transferred:      125000 bytes
Requests per second:   1353.21 [#./sec] (mean)
Time per request:      7.390 [ms] (mean)
Time per request:      0.739 [ms] (mean, across all concurrent requests)
Transfer rate:         1006.98 [Kbytes/sec] received

Connection Times (ms)
              min      mean[+/-sd] median   max
Connect:        1         2    0.7      2       4
Processing:      2         6    1.5      5      10
Waiting:         2         5    1.5      5      10
Total:          3         7    1.9      7      12

Percentage of the requests served within a certain time (ms)
 50%    7
 66%    8
 75%    9
 80%    9
 90%   10
 95%   10
 98%   11
 99%   11
100%   12 (longest request)
```



d. Generic Hash

```
This is ApacheBench, Version 2.3 <$Revision: 1843412 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.57.2.2 (be patient)

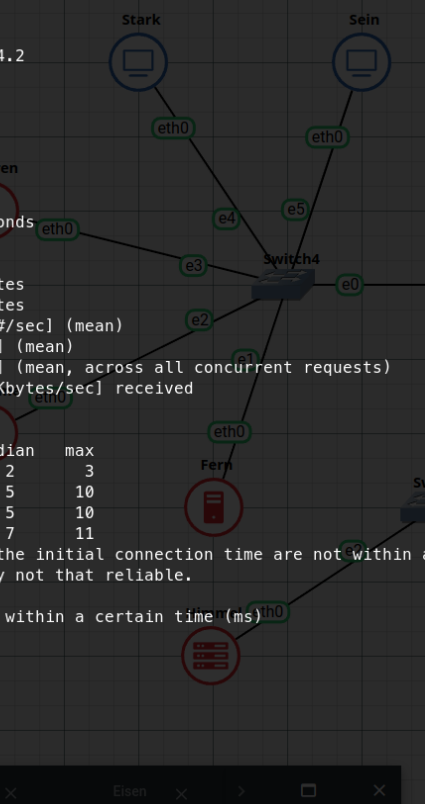
Server Software:      nginx/1.14.2
Server Hostname:      10.57.2.2
Server Port:          80

Document Path:        /
Document Length:      625 bytes

Concurrency Level:    10
Time taken for tests:  0.146 seconds
Complete requests:    200
Failed requests:       0
Total transferred:    152400 bytes
HTML transferred:     125000 bytes
Requests per second:  1369.38 [#/sec] (mean)
Time per request:      7.303 [ms] (mean)
Time per request:      0.730 [ms] (mean, across all concurrent requests)
Transfer rate:         1019.01 [Kbytes/sec] received

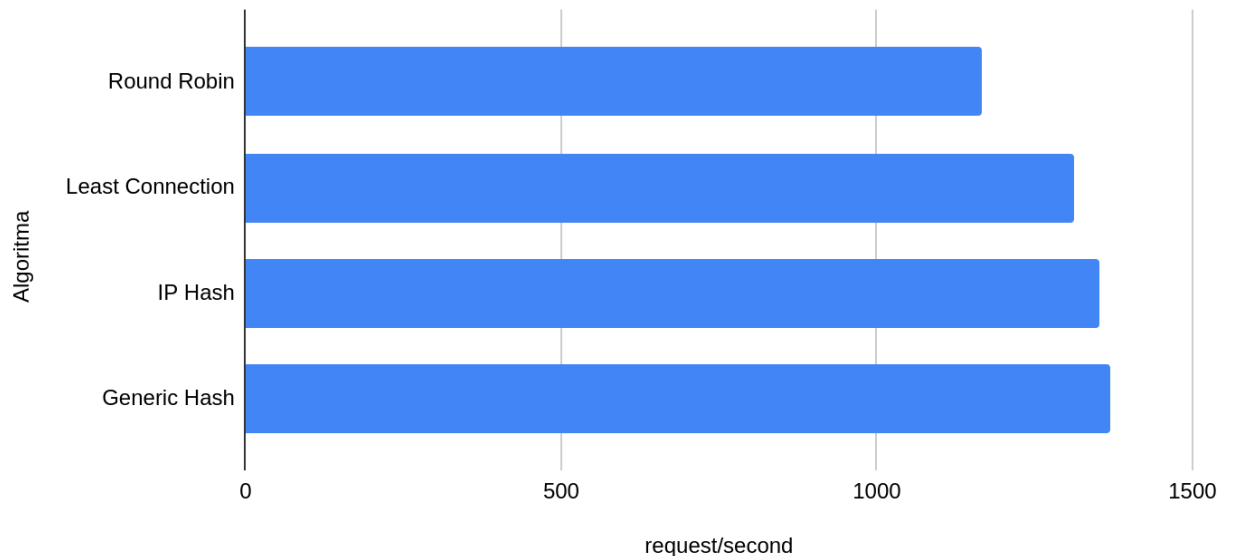
Connection Times (ms)
  min   mean[+/-sd] median   max
Connect:    0      1  0.6      2      3
Processing:  2      6  1.2      5     10
Waiting:    2      6  1.2      5     10
Total:       3      7  1.3      7     11
WARNING: The median and mean for the initial connection time are not within a
         These results are probably not that reliable.

Percentage of the requests served within a certain time (ms)
 50%      7
 66%      7
 75%      8
 80%      8
 90%      9
 95%     10
 98%     10
 99%     11
100%     11 (longest request)
```



II. Grafik

request/second vs. Algoritma



III. Analisis

Algoritma-algoritma penjadwalan yang digunakan oleh Nginx untuk memproses permintaan ke server-server terdiri dari round robin, ip hash, least connection, dan generic hash. Ketika empat server memiliki spesifikasi yang serupa dalam hal kinerja dan kapasitas, algoritma-algoritma ini akan memberikan prioritas berbeda dalam penyebaran beban kerja.

Algoritma yang cenderung memberikan kinerja paling cepat di dunia nyata adalah ip hash. Ini karena ip hash menggunakan alamat IP pengguna sebagai kunci untuk menentukan server mana yang akan melayani permintaan tersebut. Dengan cara ini, setiap pengguna akan terhubung ke server yang sama selama alamat IP mereka tetap tidak berubah, yang dapat mengoptimalkan caching dan meminimalkan overhead.

Setelah itu adalah round robin, di mana Nginx akan secara berurutan mengirimkan permintaan ke setiap server yang tersedia secara bergantian. Meskipun sederhana, pendekatan ini dapat berfungsi dengan baik untuk membagi beban secara merata di antara server-server yang setara.

Naterus ada Least connection. Algoritma ini akan mengarahkan permintaan ke server dengan jumlah koneksi aktif terendah pada saat permintaan itu dibuat. Ini membantu mengurangi tekanan pada server yang mungkin sudah sibuk dengan koneksi yang ada.

Dan yg terakhir adalah generic hash, yang biasanya digunakan untuk penyebaran beban berdasarkan kunci tertentu (misalnya, header khusus), dapat memakan waktu

lebih lama karena proses hashing yang diperlukan untuk memutuskan server mana yang harus menangani permintaan.

Namun, penting untuk diingat bahwa prioritas performa algoritma penjadwalan ini juga dapat dipengaruhi oleh kondisi jaringan, beban server, dan kebutuhan spesifik dari aplikasi atau layanan yang di host. Jadi, dalam situasi di mana spesifikasi server serupa, penyebaran beban akan bergantung pada kebutuhan unik dan karakteristik lingkungan sistem yang berjalan.

Dalam dunia nyata, lingkungan komputasi tidak pernah ideal. Berbagai faktor dapat memengaruhi performa suatu algoritma load balancing. Nggak stabilnya dalam kinerja algoritma dapat disebabkan oleh sejumlah faktor kompleks. Misalnya, variasi dalam jumlah dan tipe permintaan yang diterima oleh server, beban jaringan yang berubah-ubah, perbedaan dalam spesifikasi perangkat keras, serta faktor lingkungan eksternal seperti kecepatan koneksi internet atau latensi jaringan.

Selain itu, karakteristik aplikasi yang berubah-ubah juga dapat mempengaruhi kinerja algoritma. Misalnya, jika aplikasi tiba-tiba mengalami lonjakan permintaan, algoritma yang sebelumnya terlihat stabil dapat menunjukkan ketidakstabilan karena harus menangani beban yang lebih besar dari biasanya.

Selain itu, adanya overhead dalam penghitungan hash atau pemrosesan tambahan dalam algoritma tertentu juga dapat mempengaruhi stabilitas performa. Ini menjadikan kondisi di dunia nyata tidak selalu sesuai dengan pengujian yang dilakukan dalam lingkungan yang terkendali.

Oleh karena itu, penting untuk mempertimbangkan variabilitas dan ketidakpastian dalam lingkungan nyata saat memilih algoritma load balancing. Pengujian yang dilakukan harus melibatkan keadaan yang mewakili keadaan sebenarnya dengan sebanyak mungkin faktor yang mungkin mengaruhi performance agar dapat mengambil keputusan yang lebih tepat dalam penggunaan algoritma load balancing.

Namun, penting untuk diingat bahwa dalam situasi dunia nyata, performa sebenarnya dari masing-masing algoritma ini dapat bervariasi tergantung pada beban kerja yang sebenarnya, karakteristik permintaan, dan konfigurasi lingkungan server.