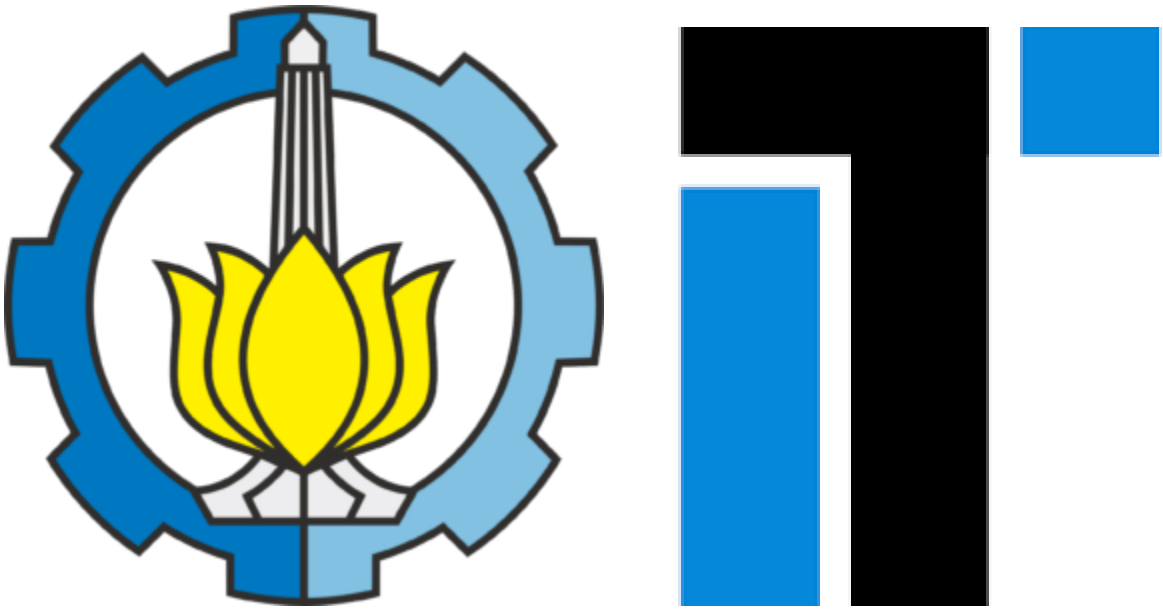
Komunikasi Data dan Jaringan Komputer
Analisis Pembahasan Soal Nomor 8 dan 9



Oleh Kelompok IT25:

Mohammad Arkananta Radithya Taratugang (5027221003)
Michael Wayne (5027221037)

Departemen Teknologi Informasi
Institut Teknologi Sepuluh Nopember
2024

Landasan Teori

Load balancing adalah proses mendistribusikan serangkaian tugas di atas serangkaian sumber daya, dengan tujuan membuat pemrosesan keseluruhannya lebih efisien.

Terdapat 4 jenis metode Load Balancing pada Nginx:

1. Round Robin
2. Least-connection
3. IP Hash
4. Generic Hash

Pada **Round Robin**, distribusi beban akan didistribusikan sesuai dengan urutan nomor dari server atau master. Jika kita memiliki 3 buah node, maka urutannya adalah dari node pertama, kemudian node kedua, dan ketiga. Setelah node ketiga menerima beban, maka akan diulang kembali dari node ke satu. Round robin sendiri merupakan metode default yang ada di Nginx.

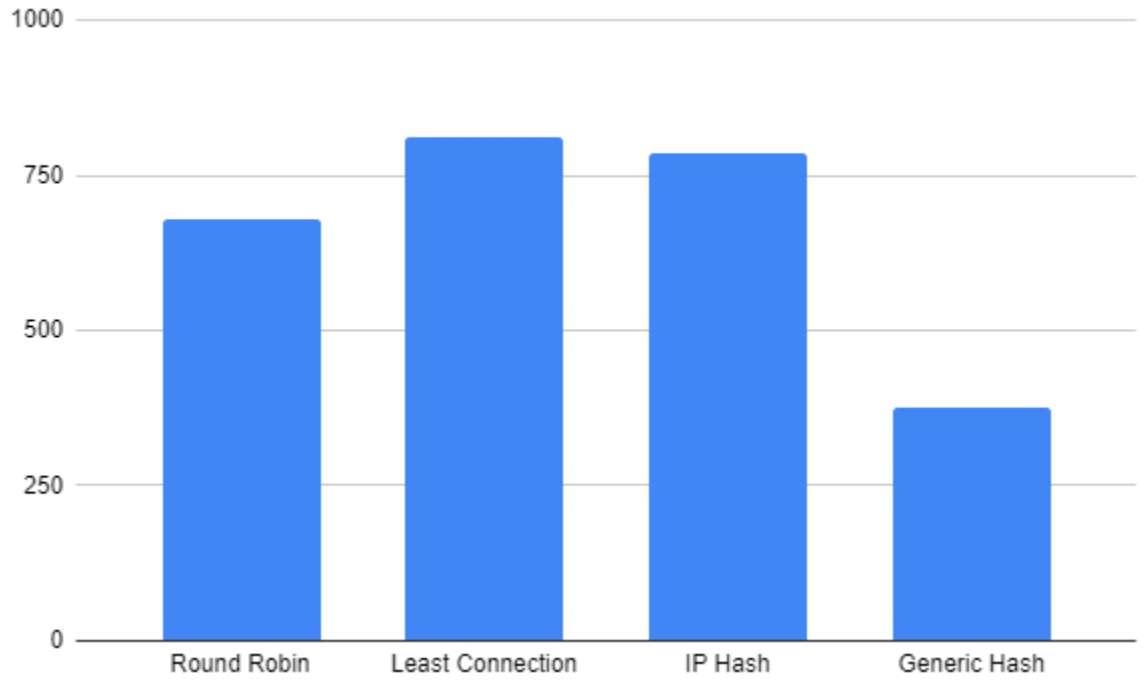
Least-connection akan melakukan prioritas pembagian dari beban kinerja yang paling rendah. Node master akan mencatat semua beban dan kinerja dari semua node, dan akan melakukan prioritas dari beban yang paling rendah. Sehingga diharapkan tidak ada server dengan beban yang rendah

Algoritma **IP Hash** ini akan melakukan hash berdasarkan request dari pengguna (menggunakan alamat IP dari pengguna). Sehingga server akan selalu menerima request dari alamat IP yang berbeda. Ketika server ini tidak tersedia, permintaan dari klien ini akan dilayani oleh server lain

Metode **Generic Hash** memetakan beban ke masing-masing node dengan cara membuat hashing berdasarkan text dan atau Nginx Variables yang ditentukan dalam hash config

Nomor 8

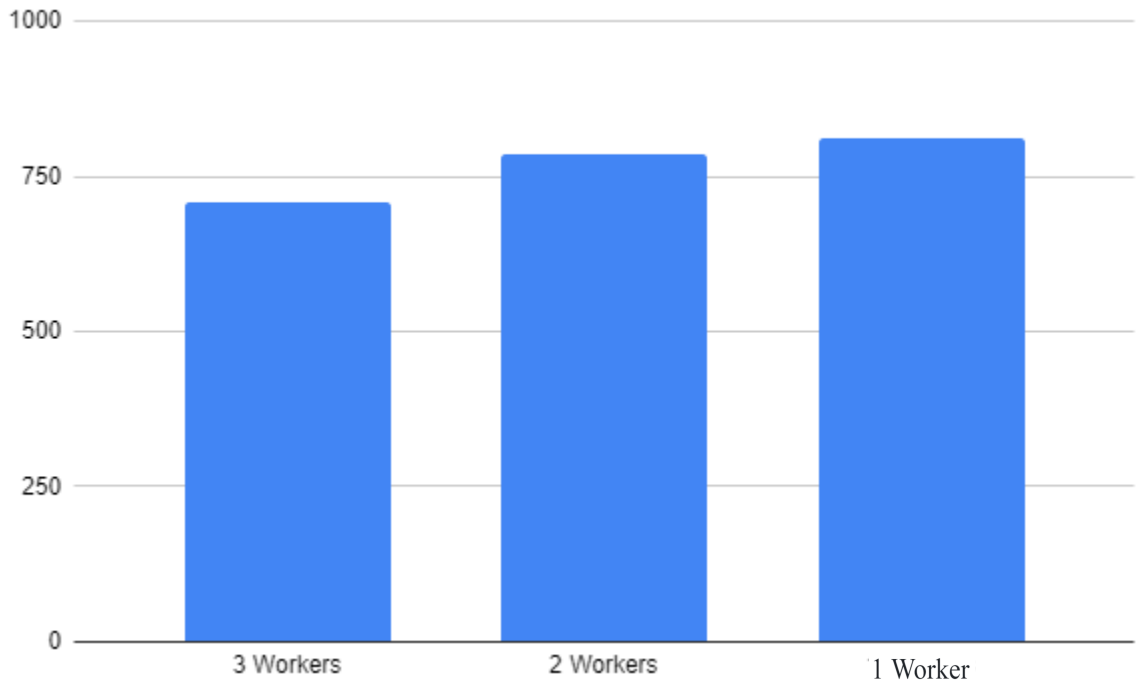
c. Grafik request per second untuk masing masing algoritma.
d. Analisis (8)



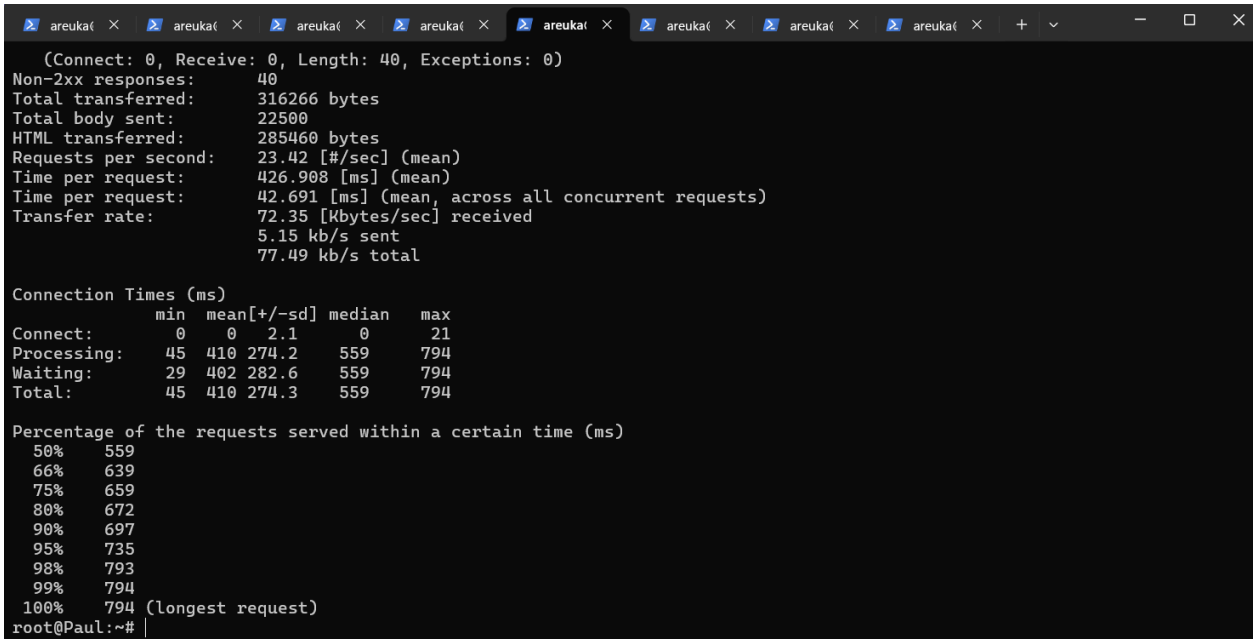
Berdasarkan data di atas, Least Connection merupakan pilihan load balancing yang optimal karena mempertimbangkan jumlah koneksi yang sedang ditangani oleh setiap server secara dinamis, menghindari overload dengan menyesuaikan pembagian beban, mengoptimalkan kinerja sistem dengan memastikan responsivitas yang seimbang, dan meningkatkan skalabilitas dengan fleksibilitas dalam mengatasi fluktuasi jumlah server dan permintaan.

Nomor 9

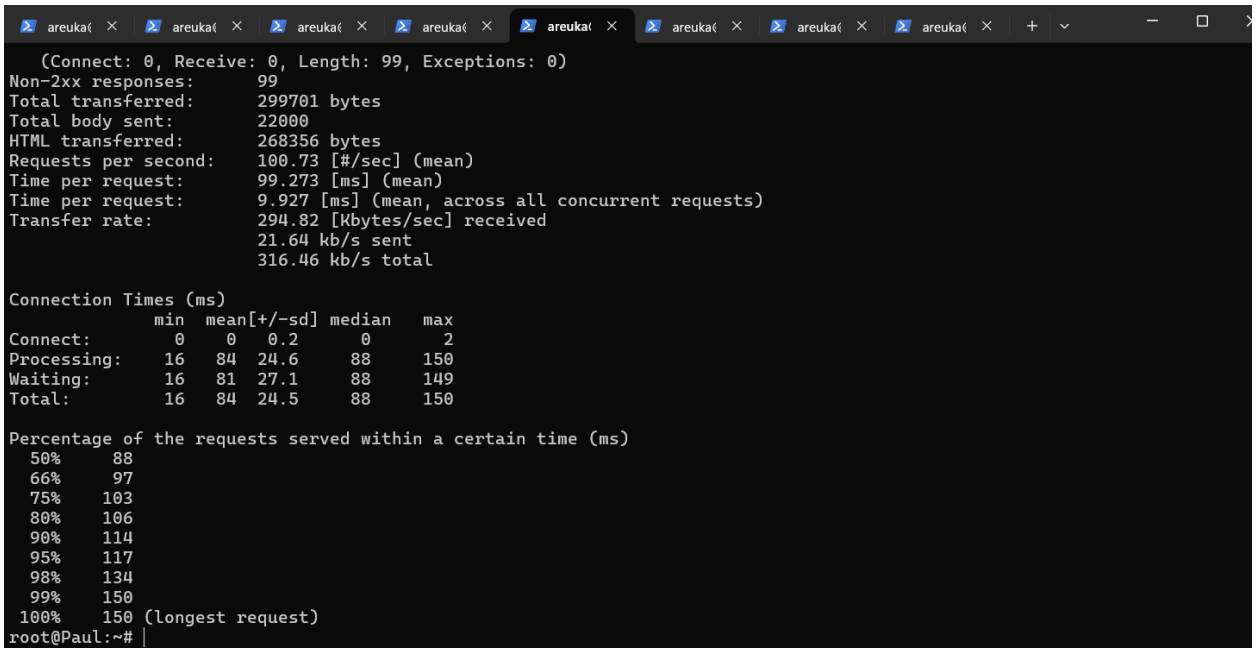
Dengan menggunakan algoritma Least-Connection, lakukan testing dengan menggunakan 3 worker, 2 worker, dan 1 worker sebanyak 1000 request dengan 10 request/second, kemudian tambahkan grafiknya pada peta.



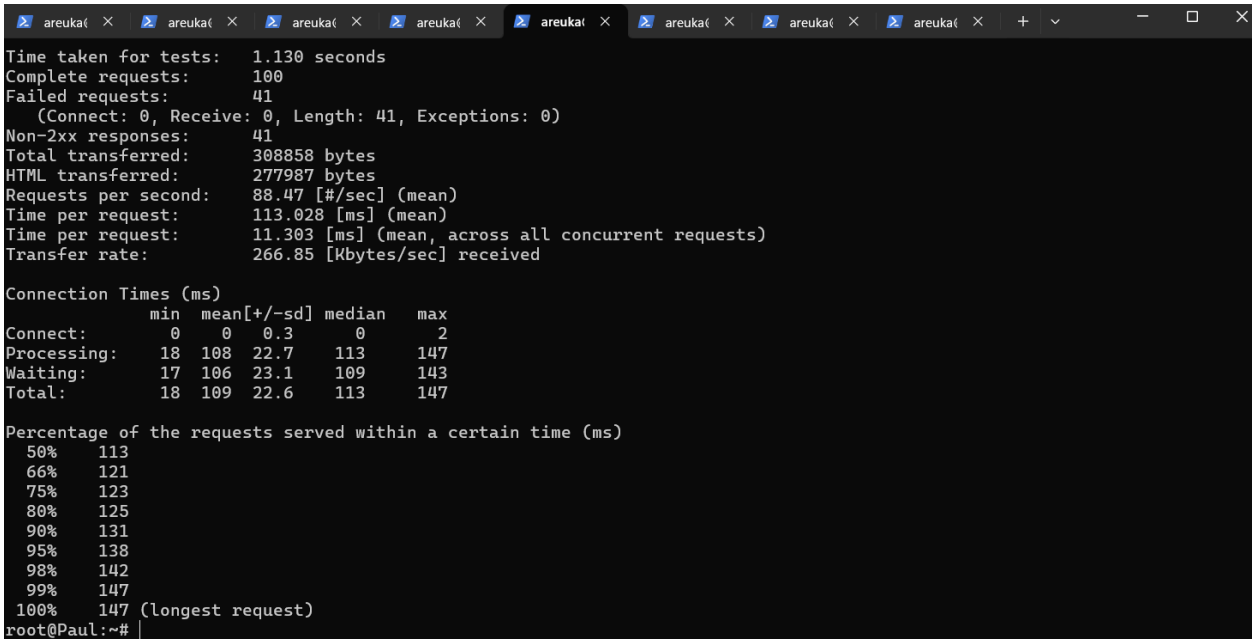
Nomor 15



Nomor 16



Nomor 17


areuka(
areuka(
areuka(
areuka(
areuka(
areuka(
areuka(
areuka(
Time taken for tests: 1.130 seconds
Complete requests: 100
Failed requests: 41
(Connect: 0, Receive: 0, Length: 41, Exceptions: 0)
Non-2xx responses: 41
Total transferred: 308858 bytes
HTML transferred: 277987 bytes
Requests per second: 88.47 [#/sec] (mean)
Time per request: 113.028 [ms] (mean)
Time per request: 11.303 [ms] (mean, across all concurrent requests)
Transfer rate: 266.85 [Kbytes/sec] received
Connection Times (ms)
min mean[+/-sd] median max
Connect: 0 0 0.3 0 2
Processing: 18 108 22.7 113 147
Waiting: 17 106 23.1 109 143
Total: 18 109 22.6 113 147
Percentage of the requests served within a certain time (ms)
50% 113
66% 121
75% 123
80% 125
90% 131
95% 138
98% 142
99% 147
100% 147 (longest request)
root@Paul:~#

Nomor 19

Soal memerintahkan untuk menaikkan

- pm.max_children
- pm.start_servers
- pm.min_spare_servers
- pm.max_spare_servers

sebanyak tiga percobaan dan lakukan testing sebanyak 100 request dengan 10 request/second

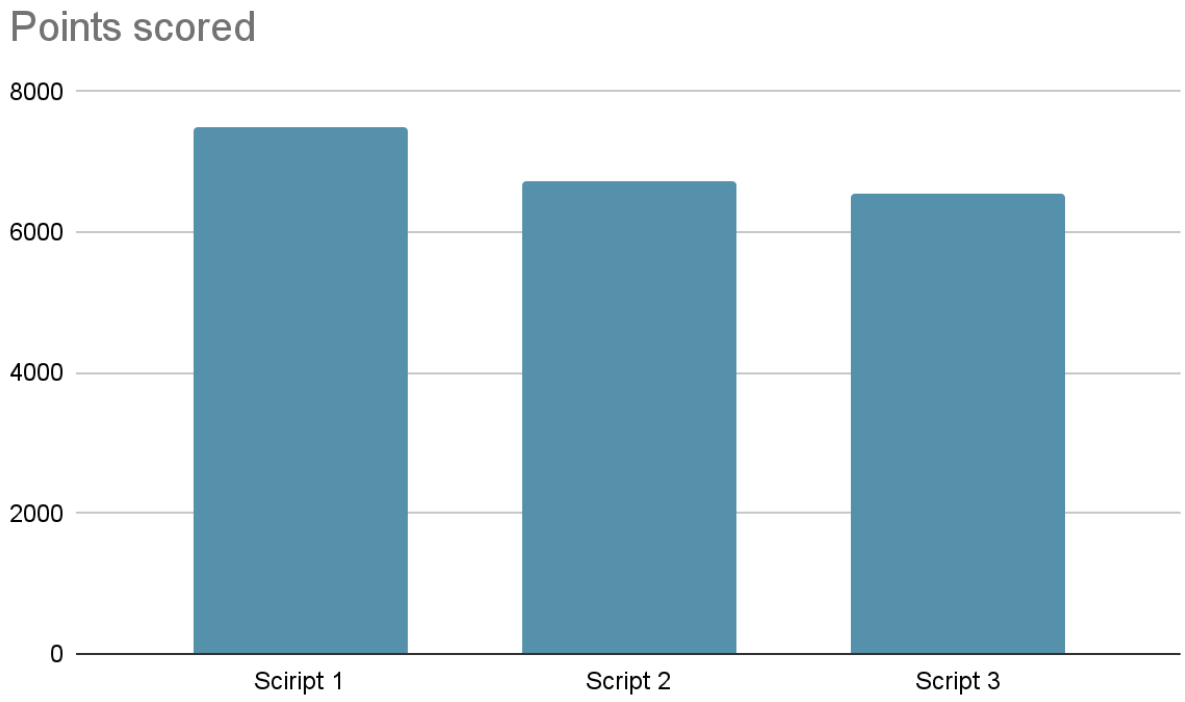
Saya membuat 3 script yang berisi angka angka yang berbeda tiap barisnya

Script 1 berisi:
pm.max_children = 3
pm.start_servers = 1
pm.min_spare_servers = 1
pm.max_spare_servers = 3

Script 2 berisi:
pm.max_children = 15
pm.start_servers = 5
pm.min_spare_servers = 5
pm.max_spare_servers = 7

Script 3 berisi:
pm.max_children = 25
pm.start_servers = 5
pm.min_spare_servers = 5
pm.max_spare_servers = 15

Berikut hasil testingnya



Data 1 menunjukkan performa terbaik di antara ketiga set data dalam hal:

Requests per second
Time per request
Transfer rate
Data 3 menunjukkan performa terendah di antara ketiga set data. Data 2 berada di tengah dalam hal kinerja tetapi lebih dekat ke Data 3 dibandingkan dengan Data 1.

Kesimpulan ini menunjukkan bahwa sistem atau server yang diuji bekerja paling efisien dan cepat pada pengujian yang menghasilkan Data 1.