BAB III
METODE PENELITIAN

Penelitian ini bertujuan untuk menganalisis penerapan *load* balancing pada perangkat *FortiGate* dengan algoritma *weighted round robin*. Penelitian ini menerapkan pendekatan kuantitatif dengan algoritma eksperimen melalui pengujian dan analisis data.

3.1 Perangkat Yang Digunakan

Perangkat pada penelitian ini dapat dibagi menjadi dua kategori yaitu, perangkat keras (*hardware*) dan perangkat lunak (*software*)

3.1.1 Perangkat keras (*Hardware*)

Perangkat keras merupakan perangkat yang digunakan untuk melakukan simulasi, dan menjalankan perangkat lunak. Pada penelitian ini menggunakan 1 perangkat keras yaitu laptop, spesifikasi perangkat tercantum dalam Tabel 3.1.

Tabel 3. 1 Spesifikasi perangkat keras

Sistem Operasi	Windows 10
Prosesor	Intel Core I7-4702MQ 2,20 Ghz
RAM	10 Gb
HDD	500 Gb

3.1.2 Perangkat Lunak (*Software*)

Pada penelitian ini perangkat lunak dibedakan menjadi dua yaitu, perangkat virtual dan perangkat lunak *tools*.

3.1.2.1 Perangkat Virtual

Pada penelitian ini menggunakan 3 perangkat lunak virtual yaitu 1 perangkat *FortiGate* sebagai *load balance* dan 2 server *web*. Spesifikasi setiap perangkat lunak virtual yang digunakan dalam penelitian dicantumkan dalam Tabel 3.2

Tabel 3. 2 Spesifikasi Perangkat Virtual

FortiGate	Sistem operasi	FortiOS
	RAM	2 GB
	CPU	1 Core
Web server 1	Sistem operasi	Ubuntu Live Server 18.04
	RAM	1 GB
	CPU	1 Core
Web server 2	Sistem operasi	Ubuntu Live Server 18.04
	RAM	2 GB
	CPU	2 Core
Client	Sistem Operasi	Ubuntu Desktop 18.04
	RAM	2 GB
	CPU	2 Core

3.1.2.2 Software Tools

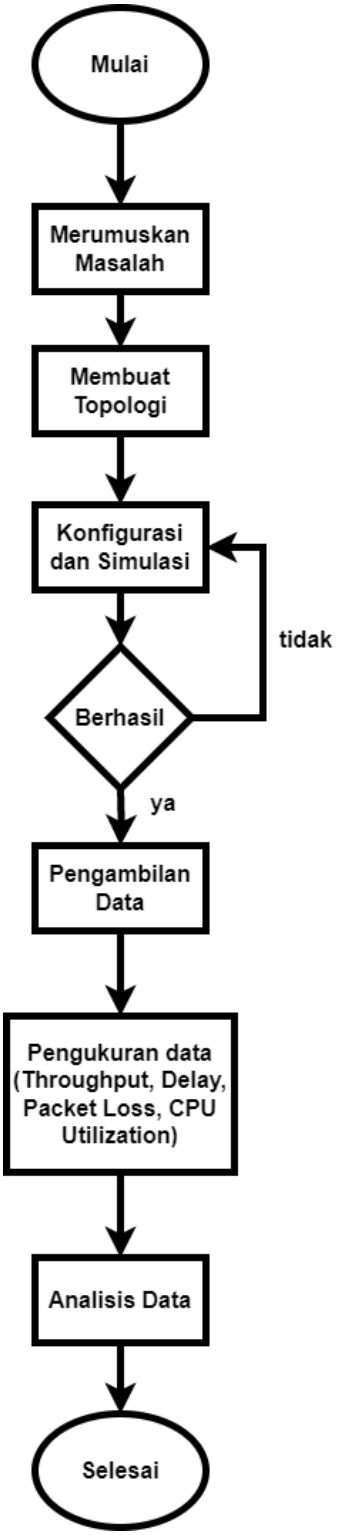
Perangkat lunak sebagai *tool* yang digunakan pada penelitian ini dilampirkan dalam Tabel 3.3.

Tabel 3. 3 *Software* Tools

No	*Software*	Fungsi
1	VMware	Virtualisasi
2	Apache	*web* server
3	EVE-NG	Emulator Jaringan
4	Apache benchmark	Tools pengujian web server
5	Wireshrak	*Network* Analyzer

3.2 Alur Penelitian

Penelitian ini dilakukan dalam beberapa tahap dimulai dari merumuskan perumusan masalah, membuat topologi jaringan, konfigurasi dan simulasi, mengambil data, serta menganalisis data dapat dilihat pada Gambar 3.1. Proses dalam konfigurasi dan simulasi antara lain: konfigurasi perangkat jaringan, install web server, membuat *load balance*, pengujian akses.

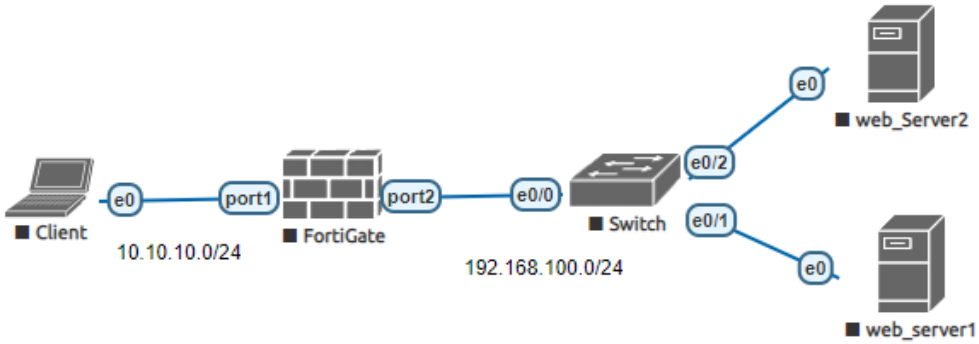


Gambar 3. 1 Diagram alur penelitian

Diagram alur perancangan sistem diperlukan untuk mencapai hasil penelitian yang diinginkan. Dimulai dari rumusan masalah yang menjadi dasar pembuatan *load balance*. Kemudian diperlukan perancangan topologi jaringan untuk

menjalankan *load balance* server. Setelah topologi berhasil dirancang di lanjutkan dengan melakukan konfigurasi dan simulasi. Konfigurasi yang dilakukan pada proses ini antara lain, melakukan pengalamatan IP pada setiap perangkat, melakukan intsalasi web server pada setiap server, dan melakukan konfigurasi *load balance* di *FortiGate*. Untuk mengetahui apakah konfigurasi telah berhasil perlu dilakukan simulasi percobaan, di mana *client* akan mencoba mengakses web server. Setelah semua berjalan dengan baik, dilanjutkan dengan pengambilan data. Pengambilan data dilakukan dengan *client* yang mengirim *request* menggunakan *software* apache benchmark kemudian traffic pada masing-masing server akan di *capture* menggunakan wireshark. Data yang telah di *capture* kemudian di analisis untuk melihat dampak dari penerapan *load balance*.

3.3 Topologi Jaringan



Gambar 3. 2 Topologi Jaringan

Pada Gambar 3.2 topologi jaringan menggunakan 2 buah *web* server, di mana server 1 dan 2 bersifat aktif. *FortiGate* berguna sebagai perangkat untuk proses *load balance*. Algoritma *weighted Round Robin* pada *FortiGate* mampu membagi beban pada server dengan spesifikasi berbeda sesuai bobot yang telah ditentukan. *FortiGate* akan membuat HTTP virtual server yang akan diakses oleh *client* dan meneruskan traffic ke server 1 dan server 2. Penjelasan fungsi setiap perangkat dapat dilihat pada Tabel 3.4

Pemberian alamat IP dapat dilihat pada Tabel 3.5. Semua server berada pada satu *network* yang sama, di mana server 1 memiliki IP 192.168.100.11, server 2 memiliki IP 192.168.100.12. Pada *FortiGate* port 1 memiliki IP 10.10.10.1, pada

port 2 memiliki IP 192.168.100.1, dan virtual IP 10.10.10.100. Virtual IP (VIP) merupakan IP virtual yang digunakan *client* untuk mengakses ke web server.

Tabel 3. 4 Fungsi perangkat

Perangkat	Fungsi
Server1	Web server dengan status aktif
Server2	Web server dengan status aktif
Switch	Menghubungkan semua server dalam satu *network*
FortiGate	Perangkat *load balance* dan *gateway*

Tabel 3. 5 Daftar alamat IP

Perangkat	*Interface*	Alamat IP
Client	e0	10.10.10.10/24
Server 1	e0	192.168.100.11/24
Server 2	e0	192.168.100.12/24
FortiGate	Port 2 (*gateway*)	192.168.100.1/24
	Port 1	10.10.10.1/24
	VIP (virtual IP)	10.10.10.100/24

3.4 Skenario Pengujian

3.4.1 Konfigurasi jaringan

Gambar 3.3 menunjukan konfigurasi IP pada perangkat Fortigate.Terdapat dua port pada *FortiGate* yang perlu dilakukan konfigurasi IP yaitu pada port1 dan port2. Port1 merupakan port mengarah ke *client*, sementara port2 merupakan port sebagai *gateway* dari server. *Command allowaccess* digunakan untuk mengizinkan *protocol* melewati port tersebut, hal ini diperlukan karena secara *default* port akan memblokir semua protokol. Pada server, konfigurasi dilakukan di directory /etc/netplan/00-installer-config.yml. Gambar 3.4 menunjukan *gateway* dari server di arahkan ke IP 192.168.100.1 yang merupakan IP port2 dari *FortiGate*.

```
config system interface
    edit "port1"
        set vdom "root"
        set ip 10.10.10.1 255.255.255.0
        set allowaccess ping https ssh http
        set type physical
        set snmp-index 1
    next
    edit "port2"
        set vdom "root"
        set ip 192.168.100.1 255.255.255.0
        set allowaccess ping https ssh http
        set type physical
        set snmp-index 2
    next
```

Gambar 3. 3 Konfigurasi IP pada *interface FortiGate*

```
 GNU nano 2.9.3                          /etc/netplan/00-installer-config.yaml

# This is the network config written by 'subiquity'
network:
  renderer: networkd
  ethernets:
    ens3:
      dhcp4: no
      addresses: [192.168.100.11/24]
      gateway4: 192.168.100.1
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
  version: 2
```

Gambar 3. 4 konfigurasi IP pada server 1

3.4.2 Instalasi Web Server

Sebelum melakukan instalasi web server pastikan setiap server dapat terhubung ke internet. Lakukan instalasi web server ke semua server yang telah disiapkan. Untuk melakukan perubahan pada tampilan server saat diakses perlu dilakukan perubahan pada *file* index.html dalam directory /var/www/html/index.html. Perintah intalasi web server apache2 pada ubuntu dapat dilihat pada Gambar 3.5.

```
sudo apt update
sudo apt install apache2
```

Gambar 3. 5 baris perintah instalasi web server

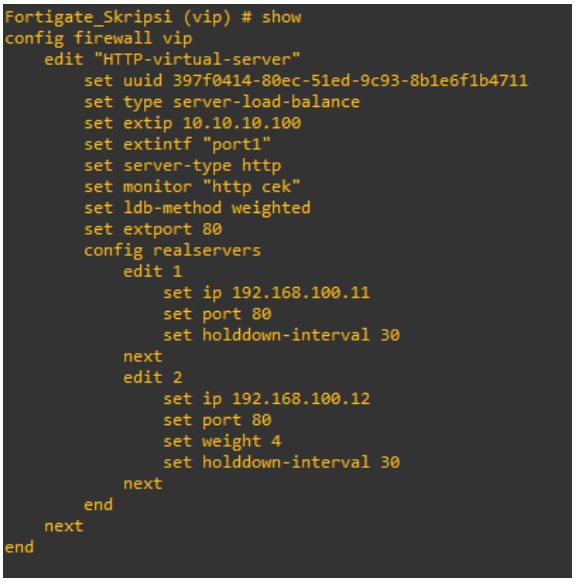
3.4.3 Konfigurasi *Load Balance*

Sebelum melakukan konfigurasi *load balance*, perlu untuk melakukan konfigurasi *health check* terlebih dahulu. *Health check* berguna supaya perangkat *FortiGate* mengetahui apakah server dalam keadaan mati/hidup. Pada Gambar 3.6 *health check* diberi nama http cek dengan type http, di mana *FortiGate* akan mengirimkan http get dan server akan menanggapi dengan http ok. Interval dibuat dalam waktu 10 detik yang mana *FortiGate* akan mengirimkan http get setiap 10 detik, jika dalam waktu 2 detik server tidak memberikan *response* maka server akan dianggap mati.

```
Fortigate_Skripsi (http cek) # show full-configuration
config firewall ldb-monitor
    edit "http cek"
        set type http
        set interval 10
        set timeout 2
        set retry 3
        set port 0
        set src-ip 0.0.0.0
        set http-get ''
        set http-match ''
        set http-max-redirects 0
    next
end
```

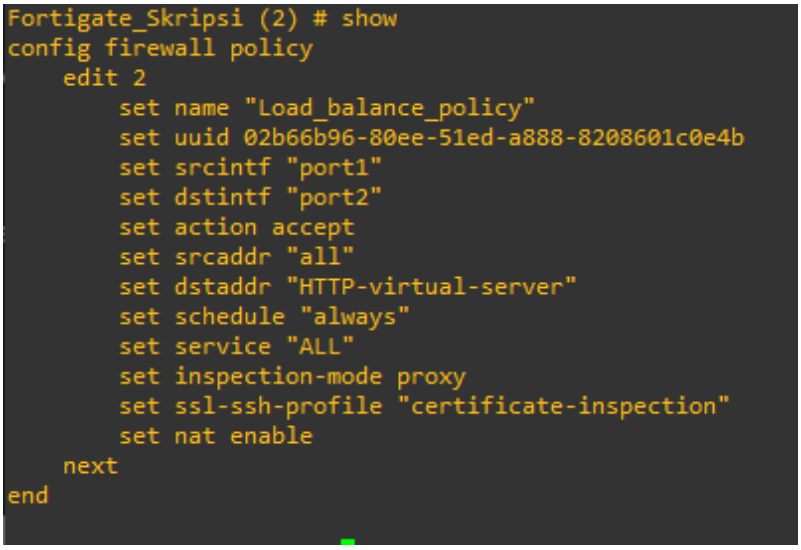
Gambar 3. 6 konfigurasi *health* check

Konfigurasi *load balance* dapat dilihat pada Gambar 3.7, dimana IP dari setiap server didaftarkan di dalam perintah realserver. Untuk monitoring server terdapat perintah set monitor yang diarahkan ke “http cek” yang merupakan nama dari *health check* yang telah dibuat. Pemiliham algortima dapat dilakukan pada perintah set ldb-method. Konfigurasi virtual IP dilakukan dengan perintah set extip. Dengan server-type http maka *FortiGate* hanya akan melakukan *load balance* permintaan http yang ditunjukan ke alamat virtual IP. konfigurasi *weight* dapat dilakukan dengan perintah set *weight* pada setiap realserver yang telah ditambahkan. Jika *weight* tidak diatur maka secara *default* akan memiliki *weight* 1.



Gambar 3. 7 konfigurasi *load balance*

Setelah *load balance* selesai dikonfigurasi, perlu dilakukan konfigurasi tambahan pada firewall *policy*. Konfigurasi ini diperlukan agar *client* dapat mengakses virtual IP melalui port1. Pada Gambar 3.8 source *interface* diarahkan pada port1 dan destination *interface* ke port2 Selain itu policy ini juga berguna untuk mengizinkan virtual IP mengakses ke server.

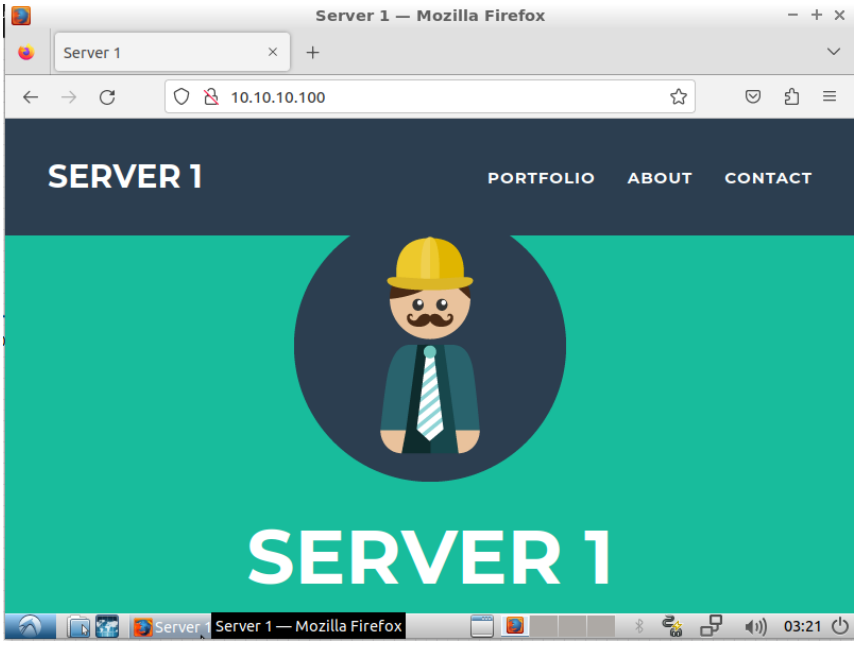


Gambar 3. 8 Konfigurasi firewall policy

3.4.4 Pengujian Akses

Pengujian akses digunakan untuk mengetahui apakah *load balance* sudah berfungsi dengan baik. Selain itu, pada dasarnya *load balance* server pada

FortiGate hanya memiliki algoritma dengan label *weight*ed. Untuk itu pengujian ini diperlukan untuk mengetahui apakah algortima *weight*ed pada *FortiGate* merupakan algoritma *weight*ed *round robin*, dengan cara melihat pola bagaimana server melayani permintaan. Percobaan akses dilakukan dengan mengakses alamat virtual IP pada browser *client*. Dengan mengakses IP tersebut pada browser dapat dilihat web server mana yang sedang diakses. Pada Gambar 3.9 merupakan tampilan dari server1.



Gambar 3. 9 Tampilan akses alamat virtual IP

Tabel 3. 6 Hasil uji akses pada *weight* 1:2

Pengujian	Server 1	Server 2
Akses ke 1		Melayani
Akses ke 2		Melayani
Akses ke 3	Melayani	
Akses ke 4		Melayani
Akses ke 5		Melayani
Akses ke 6	Melayani	
Akses ke 7		Melayani
Akses ke 8		Melayani
Akses ke 9	Melayani	
Akses ke 10		Melayani

Untuk melihat bagaimana *request* berganti antar web server dilakukan pengujian dengan perintah curl 10.10.10.100 pada terminal linux sebanyak 10 kali secara berurutan. Pada Tabel 3.6 merupakan tabel hasil dari bagaimana pola server melayani *request* pada *weight* 1:2. Berdasarkan urutan *request* yang diterima server, algortima *weight*ed untuk *load balance* server pada *FortiGate* dapat dikatakan sebagai algoritma *Weighted Round Robin*.

3.4.5 Pengujian *Load Balance*

Pengujian ini bertujuan untuk menentukan bagaimana pengaruh pembagian beban terhadap kinerja masing-masing server. Selain itu, untuk menentukan pembagian beban yang lebih optimal digunakan 3 rasio *weight* yang berbeda, dari *weight* 1:2, *weight* 1:3, *weight* 1:4. Pengujian akan dilakukan menggunakan apache benchmark dan wireshark untuk menentukan *throughput*, *delay*, CPU *utilization*, *packet loss*. Apache benchmark akan mengirim 1000, 3000, 6000 permintaan ke server dengan *concurrency* 50. Pengujian akan dilakukan pada sisi server. Pembagian beban dapat dilihat pada Tabel 3.7 di mana server dengan nilai bobot yang lebih tinggi menerima persentase koneksi yang lebih besar. Pada *Weight* 1:2 berarti server 2 akan menerima persentase koneksi 2x lebih besar dari server1. *Weight* 1:3 berarti server 2 akan menerima persentase koneksi 3x lebih besar dari server1 ataupun. *Weight* 1:4 berarti server 2 akan menerima persentase koneksi 4x lebih besar dari server1.

Tabel 3. 7 *Weight* rasio dan parameter

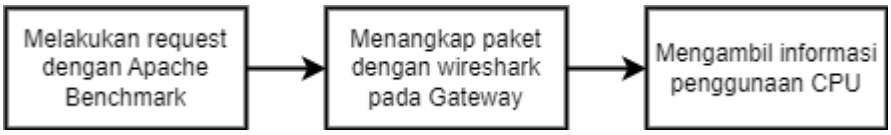
Pembagian Beban	Server 1	Server 2	Parameter
Weight 1:2	1 *request*	2 *request*	*Throughput, delay* CPU *Utilization, packet loss*.
Weight 1:3	1 *request*	3 *request*	
Weight 1:4	1 *request*	4 *request*	
Status server	Aktif	Aktif	

3.4.6 Proses Pengujian

Real Server	Status	Mode	Monitor Events	Active Sessions	RTT
10.10.10.100:80 2					
192.168.100.11:80	Up	Active	1	0	<1 second(s)
192.168.100.12:80	Up	Active	1	0	<1 second(s)

Gambar 3. 10 Status pada setiap server

Pada Gambar 3.10 server 1 dan server 2 berada mode active dan status up. Dalam mode ini, semua server dalam grup yang sama menerima dan memproses lalu lintas secara aktif. Dalam penelitian ini, lalu lintas dapat didistri*bus*ikan berdasarkan bobot/*weight* yang telah ditentukan. Dalam distri*bus*i berdasarkan *weight*, setiap server memiliki *weight* tertentu yang menentukan seberapa banyak permintaan yang akan diterima oleh setiap server.



Gambar 3. 11 Diagram blok proses pengujian

Gambar 3.11 menjelaskan urutan dalam pengambilan data yang mana dimulai dari *request* dari apache benchmark, dilanjutkan dengan menangkap paket *gateway*, dan mengambil informasi penggunaan CPU. Dalam pengujian ini apache benchmark hanya berfungsi untuk memberikan *request*, dan semua data yang dianalisis berasal dari paket yang ditangkap oleh wireshark.

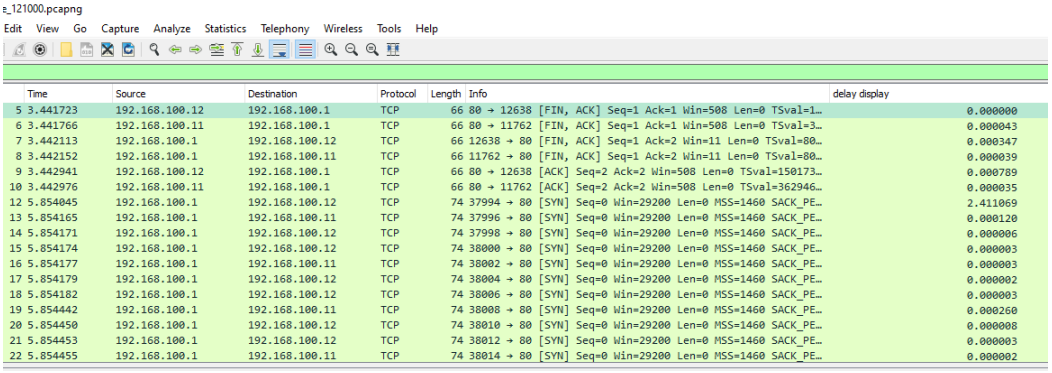
```
 GNU nano 2.9.3                    script.sh

#!/bin/bash
n=1
while [ $n -le 15 ]
do
 echo "--------------------Pengujian $n ---------------------- "
 ab -n 6000 -c 50 -r -l http://10.10.10.100/
 sleep 1
 n=$(($n+1))
done
```

Gambar 3. 12 Script pengujian apache benchmark

Pada Gambar 3.12 terdapat command yang digunakan untuk menjalankan apache benchmark. Pada -n digunakan untuk menentukan jumlah *request* yang dilakukan.

Perintah -c (*concurrency*) digunakan untuk menentukan jumlah klien atau pengguna yang akan melakukan permintaan secara bersamaan ke server web yang diuji. Perintah -r berguna agar program dapat terus berjalan dan menampilkan output meskipun terjadi kesalahan pada saat menerima data melalui soket. Perintah -l digunakan untuk tidak menampilkan *response* error jika Panjang *response* tidak konstan.



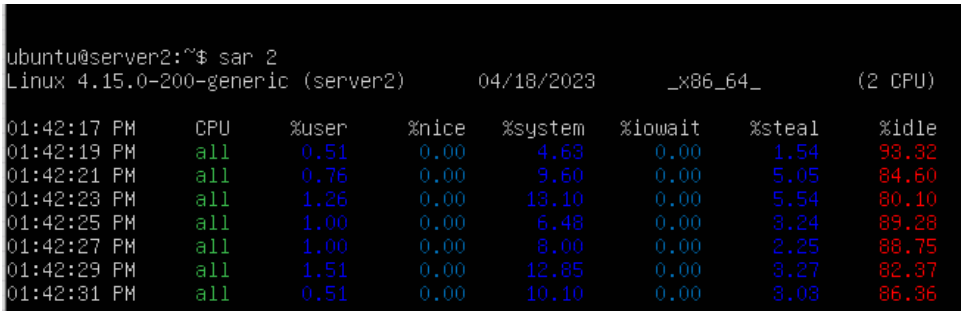
Gambar 3. 13 *Capture* lalu lintas jaringan

Statistics

Measurement	Captured	Displayed	Marked
Packets	308195	308161 (100.0%)	—
Time span, s	64.084	60.219	—
Average pps	4809.2	5117.3	—
Average packet size, B	908	908	—
Bytes	279703549	279701020 (100.0%)	0
Average bytes/s	4364 k	4644 k	—
Average bits/s	34 M	37 M	—

Gambar 3. 14 Statistik pada wireshark

Saat apache benchmark berjalan, lalu lintas kemudian ditangkap menggunakan wireshark. Proses penangkapan lalu lintas dapat dilihat pada Gambar 3.13, di mana pada kolom paling kanan terdapat *delay display* yang digunakan untuk menghitung *delay* yang terjadi. Setelah lalu lintas selesai ditangkap, wireshark akan secara otomatis memberikan hasil pengukuran beberapa parameter pada menu statistic seperti Gambar 3.14.



Gambar 3. 15 pemantauan cpu pada server

Pemantauan CPU Usage dilakukan pada server dan secara berkala setiap 2 detik menggunakan perintah “sar”, dapat dilihat pada Gambar 3.15. Melalui perintah “sar” penulis mendapatkan informasi tentang rata-rata penggunaan CPU pada server proses pengujian apache bencmark.