BAB 3
METODOLOGI PENELITIAN

3.1 Perangkat Yang Digunakan

Perangkat yang digunakan dalam penelitian ini meliputi perangkat keras *(hardware)* dan perangkat lunak *(Sofware)*.

3.1.1 Perangkat Keras (*Hardware)*

Perangkat keras yang digunakan pada penelitian ini menggunakan 1 laptop sebagai *remote server* pada *cloud* AWS dengan spesifikasi terdapat pada Tabel 3. 1.

Tabel 3. 1 Spesifikasi Perangkat Keras

Laptop	OS	Windows 10
	Processor	AMD A9-9420 RADEON R5
	RAM	4 GB

3.1.2 Perangkat Lunak (*Software)*

Perangkat lunak dalam penelitian ini menggunakan perangkat *virtual* dan beberapa aplikasi.

3.1.2.1 Perangkat Virtual

Pada penelitian ini terdapat beberapa perangkat virtual yang dibangun pada *cloud Amazon Web Service* (AWS) yaitu satu sebagai *load balancer* dan tiga *server* LMS. Sepesifikasi perangkat lunak tercantum pada Tabel 3.2

Tabel 3. 2 Spesifikasi Perangkat Virtual

Nama Perangkat Virtual	Spesifikasi	
Load Balancer	OS	Ubuntu Server 18.04
	RAM	2 GB
	Harddisk	20 GB
	Alamat IP	172.31.6.228
LMS	OS	Ubuntu Server 22.04
	RAM	16 GB
	Harddisk	50 GB
	Alamat IP	172.31.100.123

Nama Perangkat Virtual	Spesifikasi	
LMS	OS	Ubuntu Server 22.04
	RAM	16 GB
	Harddisk	50 GB
	Alamat IP	171.31.100.208
LMS	OS	Ubuntu Server 22.04
	RAM	16 GB
	Harddisk	50 GB
	Alamat IP	172.31.100.251

3.1.2.2 *Software Tools*

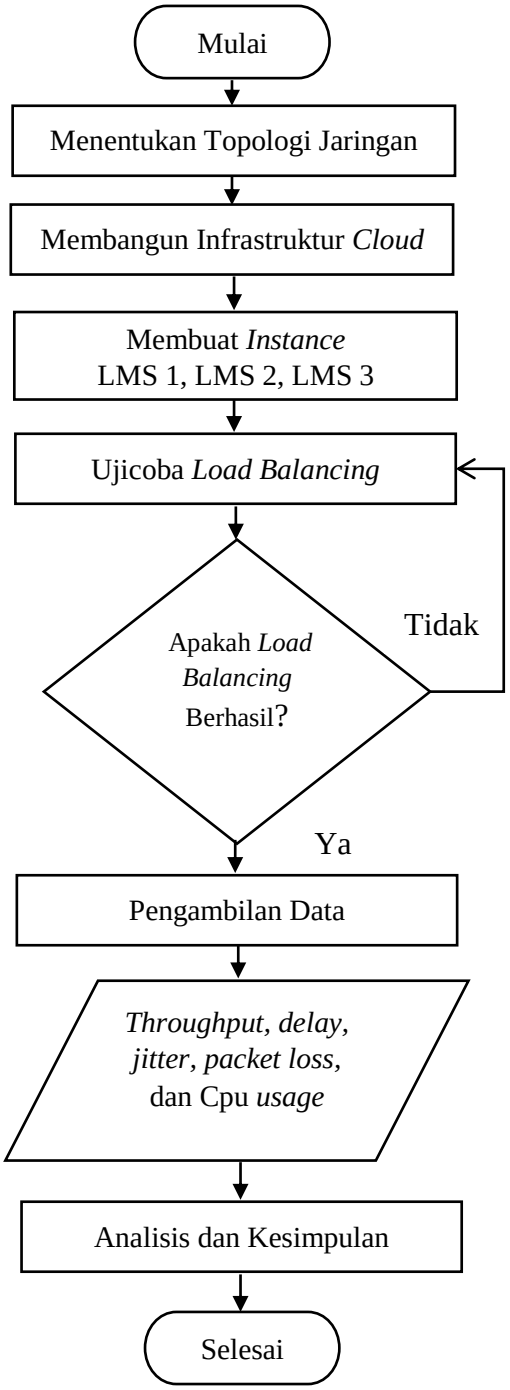
Perangkat lunak sebagai *tools* dan aplikasi yang digunakan pada penelitian ini ditunjukkan pada Tabel 3.3

Tabel 3. 3 *Software* dan *Tools*

No	Software	Versi	Fungsi
1	*Cloud* AWS		*Resource*
2	*Docker*	19.03	*Tool container*
3	*Moodle*	3.11	LMS
4	*Mariadb*	10.9.1	*Database Server*
5	*Fortigate Firewall*	7.2.3	*Load Balancing*
6	*Apache benchmark*	2.6	*Tools* Pengujian *Load balancing* LMS
7	*Wireshark*	3.2.4	Pengukuran QoS

3.2 Alur Penelitian

Alur penelitian yang dilakukan pada penelitian ini akan digambarkan pada Gambar 3.1. Dalam alur penelitian ini, akan tergambar langkah-langkah sistematis yang diikuti selama proses penelitian. Dimulai dengan menentukan topologi jaringan, selanjutnya membangun infrastruktur *cloud computing* pada *provider Amazon Web Service* (AWS). Tahap berikutnya membuat instance untuk ketiga LMS. Berikutnya dengan membuat system *load balancing* yang berfungsi membagi beban kepada *server* LMS dan setela berhasil, maka dilakukan pengambilan data menggunakan parameter seperti *throughput, delay, jitter, packet loss,* dan Cpu *usage*. Setelah selesai, data diberikan analisis dan kesimpulan.



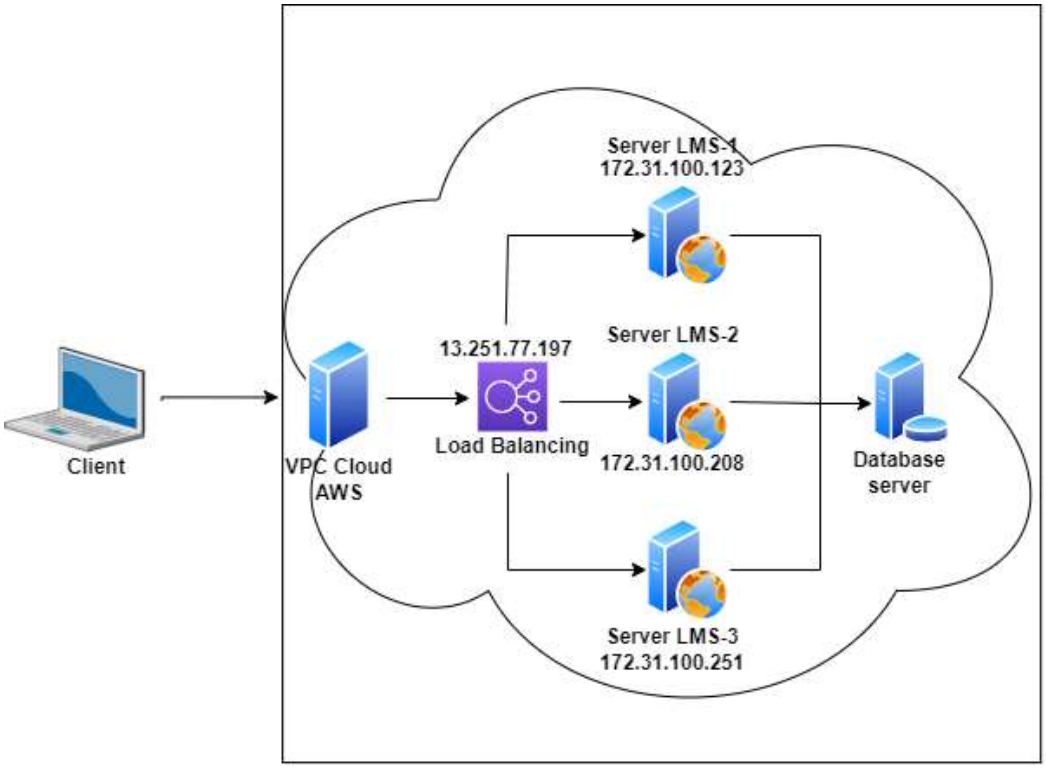
Gambar 3. 1 Diagram alur penelitian

Alur penelitian yang akan dilakukan digambarkan pada Gambar 3. 1. Hal pertama yang dilakukan membuat topologi jaringan sebelum membuat simulasi, yang bertujuan sebagai gambaran dalam melakukan implementasi *docker container* pada LMS berbasis *cloud computing*. Selanjutnya menerapkan

infrastruktur *cloud computing* dengan menggunakan *Virtual Private Cloud* (VPC) yang bertujuan untuk membuat dua jaringan secara terpisah. Setelah infrastruktur *cloud* selesai dikerjakan, hal selanjutnya yang perlu dilakukan yaitu membuat *instance* sebanyak 2 *instance* dengan satu sebagai *load balancer* dan *instance* kedua untuk meng*install* tiga 3 *server* LMS yang dibedakan *port*nya. Selanjutnya konfigurasi *load balancing*, apabila berhasil maka selanjutnya melakukan pengujian berdasarkan parameter berupa *throughput, packet loss, delay, jitter* dan CPU *usage*. Setelah data performansi dikumpulkam, maka selanjutnya menganalisisis hasil pengujian yang dilakukan untuk mendapatkan hasil performansi pada LMS menggunakan *docker container* dengan *load balancing* sebagai pembagi *traffic*.

3.3 Topologi Jaringan

Sebelum melakukan penelitian ini, diperlukan topologi jaringan sebagai gambaran implementasi *docker container* pada LMS dan *load balancing* sebagai pembagi *traffic* digambarkan pada Gambar 3.2. langkah awal yang diperlukan adalah menyusun topologi jaringan sebagai landasan implementasi *docker container* pada *Learning Management System* (LMS) serta konsep *load balancing* yang bertindak sebagai alat pembagi lalu lintas. Dalam konteks topologi ini, terdapat beberapa komponen yang saling berinteraksi. Pertama, terdapat satu entitas sebagai *client* yang akan digunakan untuk melakukan uji performansi. Selanjutnya, ada *Virtual Private Cloud* (VPC) yang memiliki dua jaringan terpisah dan memungkinkan akses internet ke kedua *instance* yang ada dalam jaringan tersebut. Komponen pertama dari *instance* adalah *fortigate firewall* yang berperan ganda sebagai *load balancing* dan *router*. Sebagai *load balancer*, fortigate firewall bertanggung jawab untuk meratakan beban trafik agar distribusi lalu lintas ke *instance* LMS berjalan seimbang. Selain itu, *fortigate* juga berfungsi sebagai router yang menghubungkan semua jaringan pada *instance*, memastikan koneksi yang tepat dan aman.Komponen kedua adalah *instance* LMS yang berperan sebagai pusat *Learning Management System*. Dengan begitu, topologi ini menggambarkan infrastruktur yang perlu diimplementasikan untuk menjalankan penelitian.



Gambar 3. 2 Topologi jaringan

3.4 Skenario Pengujian

3.4.1 Implementasi jaringan

Implementasi jaringan dalam penelitian ini menggunakan tiga *server* LMS sebagai layanan *e-learning*, satu *client* sebagai simulasi melakukan pengujian jaringan. *Load balancing* pada *server* akan membagikan beban kerja pada setiap *server* LMS. VPC berfungsi untuk membuat dua jaringan secara terpisah sekaligus memberikan akses internet untuk kedua *instance* yaitu *fortigate firewall* dan *server* LMS. *Client* akan digunakan untuk melakukan pengujian performansi dengan parameter berupa *throughput, packet loss, delay, jitter*, dan CPU *Usage* menggunakan *apache benchmark* dan *wireshark* untuk menguji *server* LMS.

3.4.2 Implementasi *Load Balancing* Pada AWS

Implementasi *load balancing* pada penelitian ini menggunakan *software fortigate firewall* dengan membagikan beban pada tiga *server* LMS. Sebelum mengimplementasikan *load balancing*, diperlukan konfigurasi *Virtual Private*

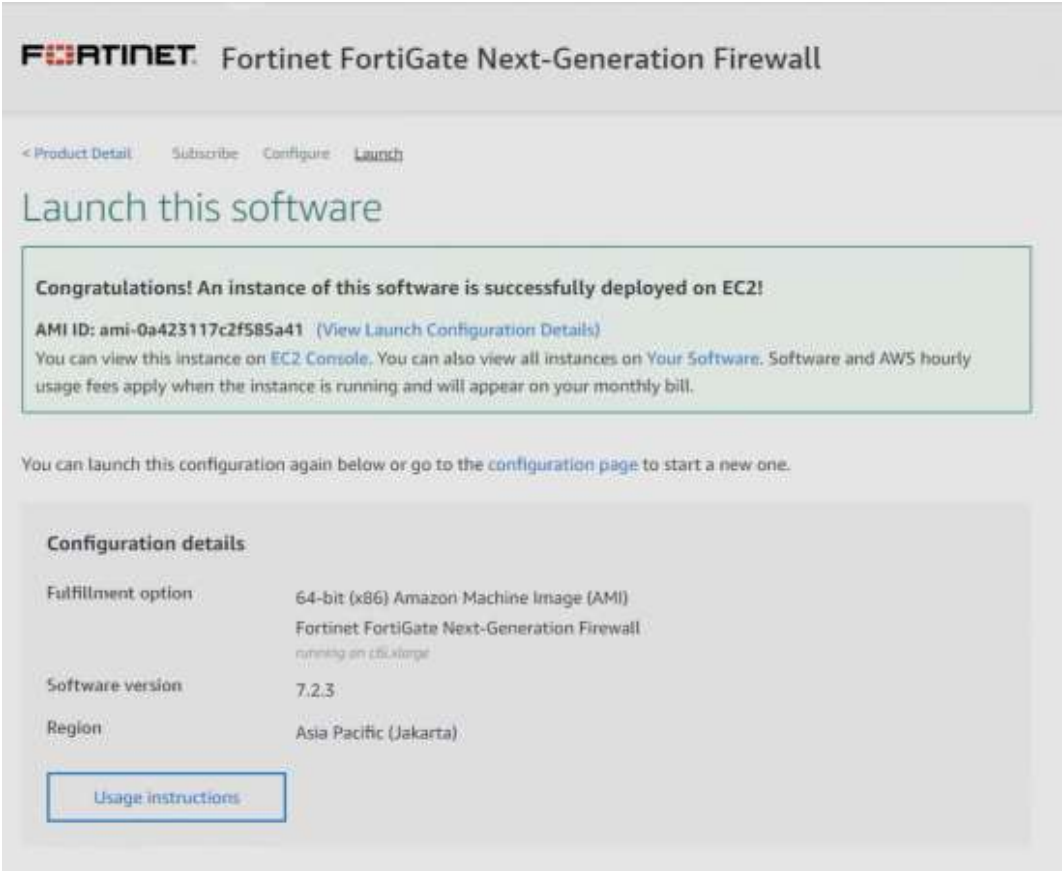
Cloud (VPC) sebagai infrastruktur jaringan yang akan memberikan akses internet pada *fortigate firewall* dan *server* LMS.

3.4.2.1. Konfigurasi *Virtual Private Cloud* (VPC)

Virtual Private Cloud (VPC) merupakan salah satu jenis layanan *Infrastructure as a Service* (IaaS) melalui *platform cloud* AWS sebagai *infrastructure network*. VPC memiliki fungsi untuk membuat dua jaringan secara terpisah. Jaringan tersebut berupa *Wide Area Network* (WAN) dan *Local Area Network* (LAN). Jaringan WAN dihubungkan ke *fortigate firewall* yang berfungsi untuk menghubungkan jaringan lokal dengan internet. Dengan menggunakan jaringan WAN, *fortigate firewall* dapat memonitor dan mengelola lalu lintas data yang masuk dan keluar dari jaringan lokal. Sedangkan untuk jaringan LAN hanya dihubungkan ke *server*. Selain itu, VPC juga menggunakan *internet gateway* agar semua *instance* yang dibuat memiliki akses internet. Pada VPC juga menggunakan dua *route* berupa *public route* dan *private route*. *Public route* dihubungkan ke *internet gateway* agar semua *routing* diarahkan ke-*default route* (*internet*), sedangkan *private route* hanya digunakan untuk *server* yang berada di-*subnet* LAN. Tujuan dibuatkan VPC adalah memberikan akses internet untuk kedua *instance*. *Instance* pertama ialah *fortigate firewall* berfungsi sebagai *load balancing* sekaligus sebagai *router* yang akan menghubungkan pada semua jaringan. *Intsance* kedua berfungsi sebagai LMS yang di dalamnya sudah ter-*install moodle* dan *database server*.

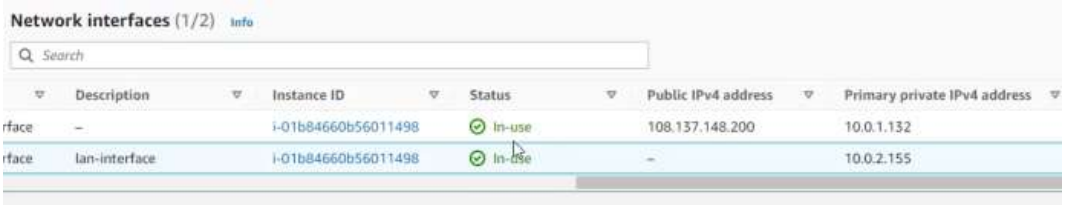
3.4.2.2. Konfigurasi *Fortigate Firewall*

Konfigurasi *fortigate firewall* dilakukan pada satu *instance* EC2 AWS yang telah dipersiapkan. *Fortigate firewall* merupakan salah satu jenis layanan *Software as a Service* (SaaS) yang ditawarkan oleh *platform cloud* AWS. Konfigurasi awal yaitu mencari *fortigate firewall* pada *menu marketplace* AWS. Setelah itu memilih opsi dan versi untuk kebutuhan *fortigate firewall* dengan membuat *instance* sebagai wadah instalasi dan dihubungkan dengan VPC serta menggunakan jaringan WAN agar memiliki akses internet. Berikut Gambar 3. 3 mengenai instalasi *fortigate firewall*.



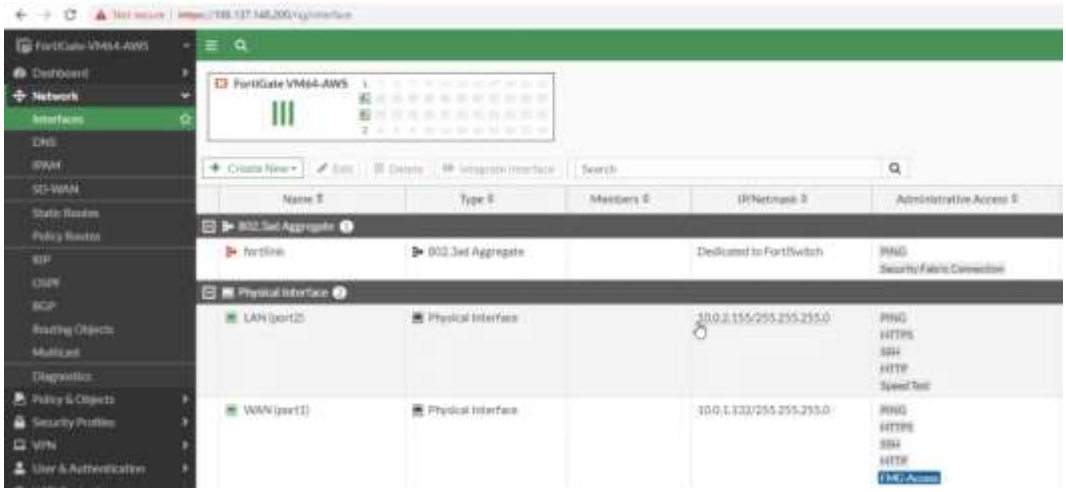
Gambar 3. 3 Instalasi *Fortigate Firewall*

Selanjutnya, pada instance *fortigate firewall* membuat *allocate elastic* IPs yang bertujuan agar *fortigate firewall* dapat IP *public* secara *dhcp*. Kemudian dibagian *network interface* di-*attach* ke *fortigate firewall* agar dapat *mapping* antara *port fortigate firewall* dengan *subnet* yang digunakan. Berikut ditunjukkan pada Gambar 3.4 mengenai *network interface* pada *instance fortigate firewall*



Gambar 3. 4 *Network interface* pada *instance Fortigate Firewall*

Selanjutnya *login* pada *fortigate firewall* menggunakan IP *public* yang didapatkan dari IP NATnya dari IP WAN.



Gambar 3. 5 *Interface Fortigate Firewall*

Pada Gambar 3. 5 akan menampilkan dari *interface fortigate firewall* yang sudah login menggunakan IP *public.*

3.4.2.3. Konfigurasi *Moodle* LMS

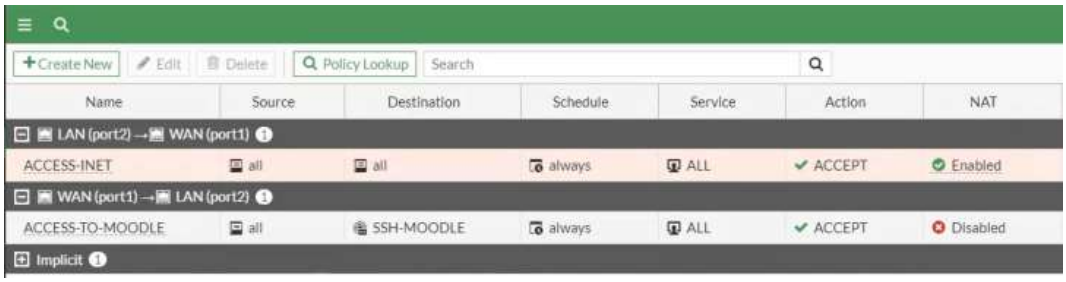
Konfigurasi *Moodle* dilakukan pada satu *instance* EC2 AWS yang telah dipersiapkan sebagai *server* LMS. Konfigurasi awal yaitu melakukan instalasi *docker* sebagai *tool container* untuk wadah *moodle* dan *database server*. Selanjutnya, *pull image moodle* dan *pull image mariadb* pada *container* yang dihubungkan oleh *docker network* . Setelah berhasil melakukan instalasi *Moodle*, langkah selanjutnya yaitu melakukan *setting* pada *instance moodle* dengan *instance fortigate firewall* agar saling terhubung. Berikut dibawah Gambar 3.5. instalasi tiga *moodle* dan satu *database server*.

Gambar 3. 6 Instalasi tiga *Moodle*

3.4.2.4. Konfigurasi dan Ujicoba Akses *Load Balancing*

Load balancing dikonfigurasi pada satu *instance* EC2 AWS yang dipersiapkan sebagai *load balancer*. *Load balancing* memakai *fortigate firewall*

yang telah diinstalasi pada *instance* EC2 AWS. Konfigurasi awal yang dilakukan dengan membuat ssh-moodle pada *virtual* IP yang bertujuan agar *server moodle* (yang tidak punya IP *public*) dapat diakses dari *public* melalui *fortigate firewall*. Selanjutnya membuat acces-to-moodle pada *firewall policy* yang betujuan untuk mengatur lalu lintas dari jaringan WAN (internet) diperbolehkan mecapai jaringan LAN yang sudah terhubung dengan *server moodle*. Langkat berikutnya dengan membuat acces-inet *pada firewall policy* yang bertujuan supaya *server moodle* mempunyai akses internet. Berikut dibawah Gambar 3. 7 konfigurasi *fortigate firewall* untuk mengakses server LMS

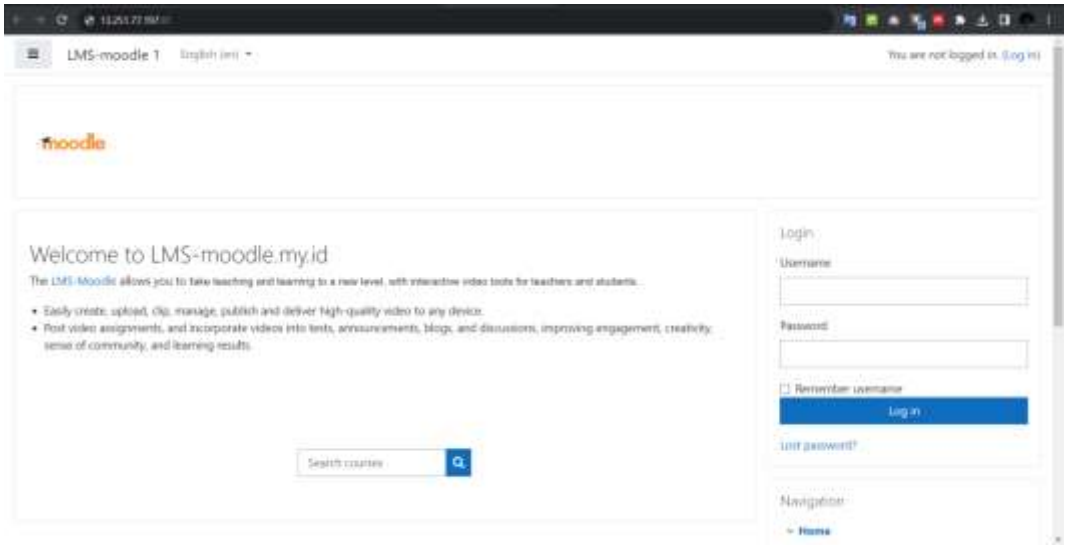


Gambar 3. 7 Konfigurasi *Fortigate Firewall* untuk akses *server* LMS

Kemudian sebelum ke konfigurasi *load balancing* dilanjutkan dengan membuat tiga *virtual* IP uktuk mengakses *server* LMS yang dibedakan *port*-nya. Berikut dibawah Gambar 3. 8 Konfigurasi tiga *virtual* IP untuk mengakses server LMS.

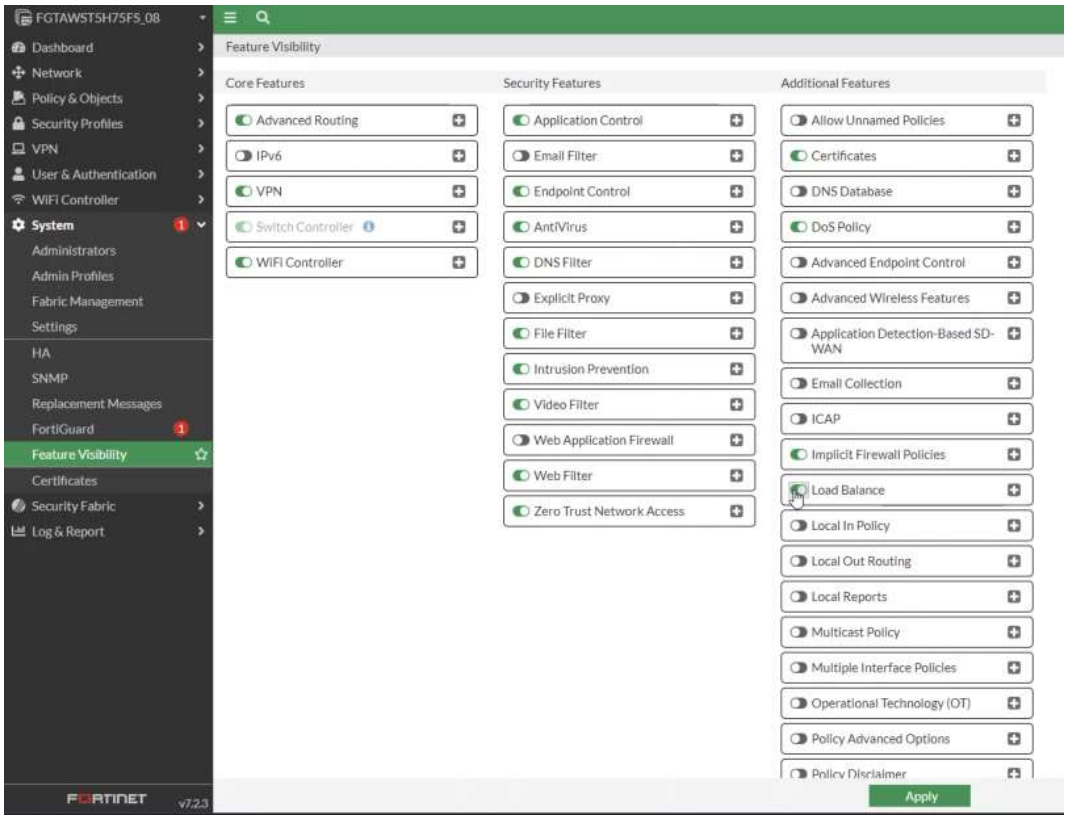
Gambar 3. 8 Konfigurasi tiga *virtual* IP untuk *server* LMS

Setelah itu, uji coba akses *server* LMS menggunakan IP *public* dengan menambahkan port untuk membedakan *server* mana yang sedang diakses. Berikut dibawah Gambar 3. 9 uji coba akses *server* LMS 1 menggunakan IP *public.*



Gambar 3. 9 Uji coba akses *server* LMS 1 menggunakan IP *public*

Langkah berikutnya untuk mengaktifkan *load balancing* yaitu dengan cara pilih menu *feature visibility* yang ada di *system fortigate firewall* dan aktifkan *load balance*. Berikut dibawah Gambar 3. 10 untuk mengaktifkan *fitur load balance.*

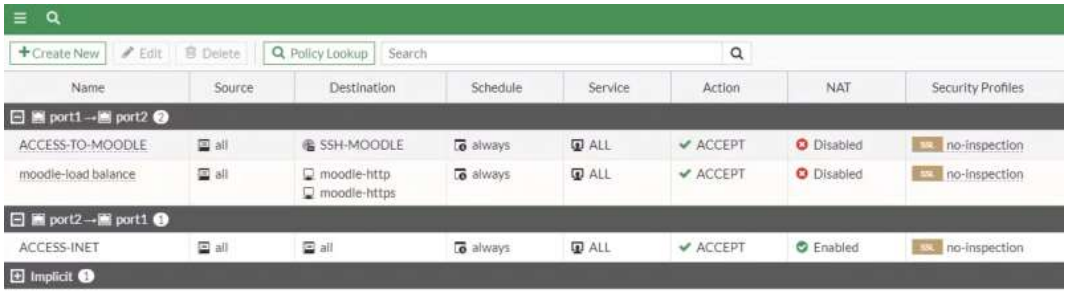


Gambar 3. 10 Mengaktifkan fitur *load balance*

Setelah mengaktifkan *load balance*, selanjutnya membuat *virtual server* untuk moodle-http dan moodle-https. Pada fitur *virtual server* berfungsi untuk mengatur load balancing dengan *service* berupa http dan https pada *server* LMS dan juga menentukan algoritma yang akan digunakan yaitu algoritma *least connection*. Berikut dibawah Gambar 3. 11 membuat *virtual server* untuk *moodle-http* dan *moodle-https.*

Gambar 3. 11 *Virtual server* untuk *moodle-http* dan *moodle-https*

Lalu, membuat moodle-load balance pada *firewall policy* yang bertujuan untuk mengelola server LMS agar dapat diakses melalui *http* atau *https* dan sekaligus mengaktifkan *load balancing*. Berikut dibawah Gambar 3. 12 untuk membuat *moodle-load balance.*

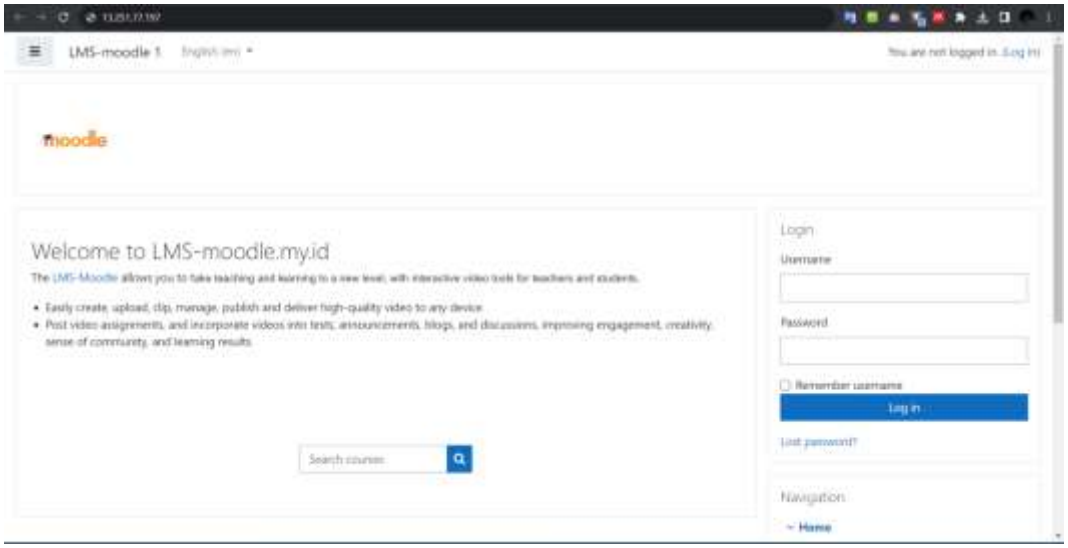


Gambar 3. 12 *Moodle-load balance*

Setelah itu, untuk mengetahui *server* LMS yang sedang aktif dapat diketahui dengan *klik* pada *menu dasboard fortigate firewall* dan klik *load balance monitor*. Berikut dibawah Gambar 3. 13 untuk konfigurasi *load balancing* dengan *fortigate firewall*.

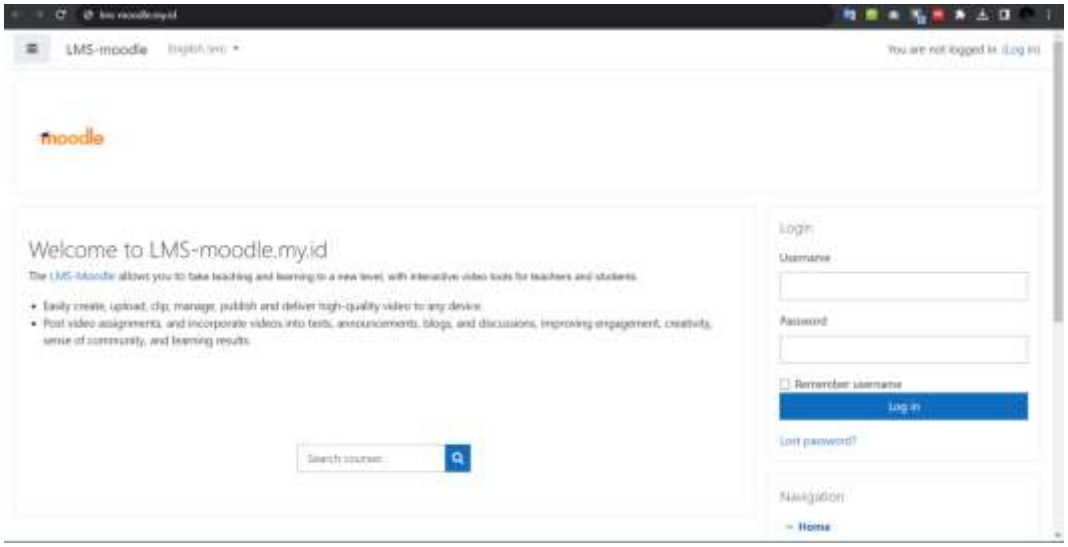
Gambar 3. 13 Konfigurasi *load balancing* dengan *Fortigate Firewall*

Sistem *load balancing* perlu diuji akses untuk melihat apakah sistem berhasil. Uji akses dilakukan dengan memakai alamat IP *public* dari *fortigate firewall* pada *browser*. Seperti pada Gambar 3.14, ketika mengakses LMS menggunakan alamat IP *public* pada *browser*.



Gambar 3. 14 Tampilan akses LMS

Selanjutnya agar dalam mengakses *server* LMS lebih mudah yang awalnya menggunakan IP *public* dapat dirubah *domain*nya menjadi lms-moodle.my.id. Berikut gambar 3.15 tampilan LMS yang sudah dirubah *domain*-nya.



Gambar 3. 15 Tampilan akses LMS setelah berubah *domain*

3.4.3 Pengujian Performansi LMS

Pengujian performansi dengan parameter *Quality of Service* (QoS) *throughput*, *delay*, *packet loss*, *jitter* dan parameter luaran yaitu CPU *usage* bertujuan untuk melihat kualitas jaringan pada saat permintaan dikirimkan ke *server* LMS. Pengujian dilakukan sebanyak 30 kali, yang selanjutnya akan diambil hasil rata-rata pada setiap parameter. Pengujian pemberian beban dilakukan dengan bantuan *software apache benchmark* 1000, 3000, 5000, 7000 dan 10000 dengan 10 *request rate* per detik yang kemudian hasil datanya di*capture* pada *wireshark* yang ada pada pc *client*. Jumlah permintaan, *request* per detik dan banyaknya pengujian yang akan dilakukan dilampirkan pada Tabel 3.4.

Tabel 3. 4 Skenario jumlah beban *request* dan banyaknya pengujian

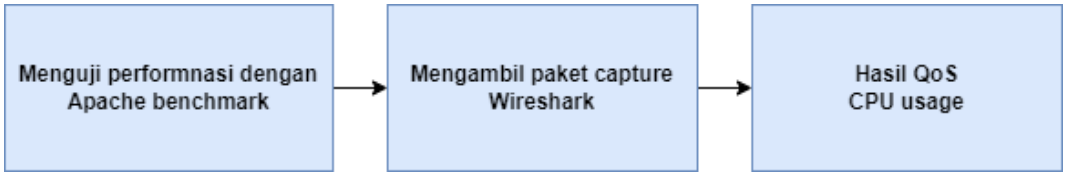
No	Jumlah permintaan	*Request* per detik	Banyaknya Pengujian
1	1000	10	30
2	3000	10	30
3	5000	10	30
4	7000	10	30
5	10000	10	30

3.4.4 Proses Pengujian

Pengujian LMS dilakukan dengan mengirimkan *request* menggunakan *apache benchmark.* Pengujian dilakukan dengan menngunakan *script* “bpy” yang bertujuan untuk menjalankan *apache bencmark* sebagai *traffic generator dan* untuk menjalankan *wireshark* sebagai *packet capture* secara bersamaan. Pada perintah “ab” digunakan untuk menjalankan uji kinerja atau pengujian beban pada *server* LMS, sedangkan perintah “-n” digunakan untuk mengatur jumlah total permintaan yang akan dikirimkan selama pengujian dan perintah “-c” untuk mengatur jumlah koneksi *concurent* yang akan dibuat secara bersamaan. Pada perintah “tshark” digunakan untuk menganalisis jaringan yang sedang berjalan pada *wireshrak*, " -i any" menginstruksikan *tshark* untuk menangkap lalu lintas dari semua antarmuka jaringan yang ada, dan "-w filename" mengarahkan *tshark* untuk menyimpan data lalu lintas yang ditangkap ke dalam *file* dengan nama yang diberikan (filename).

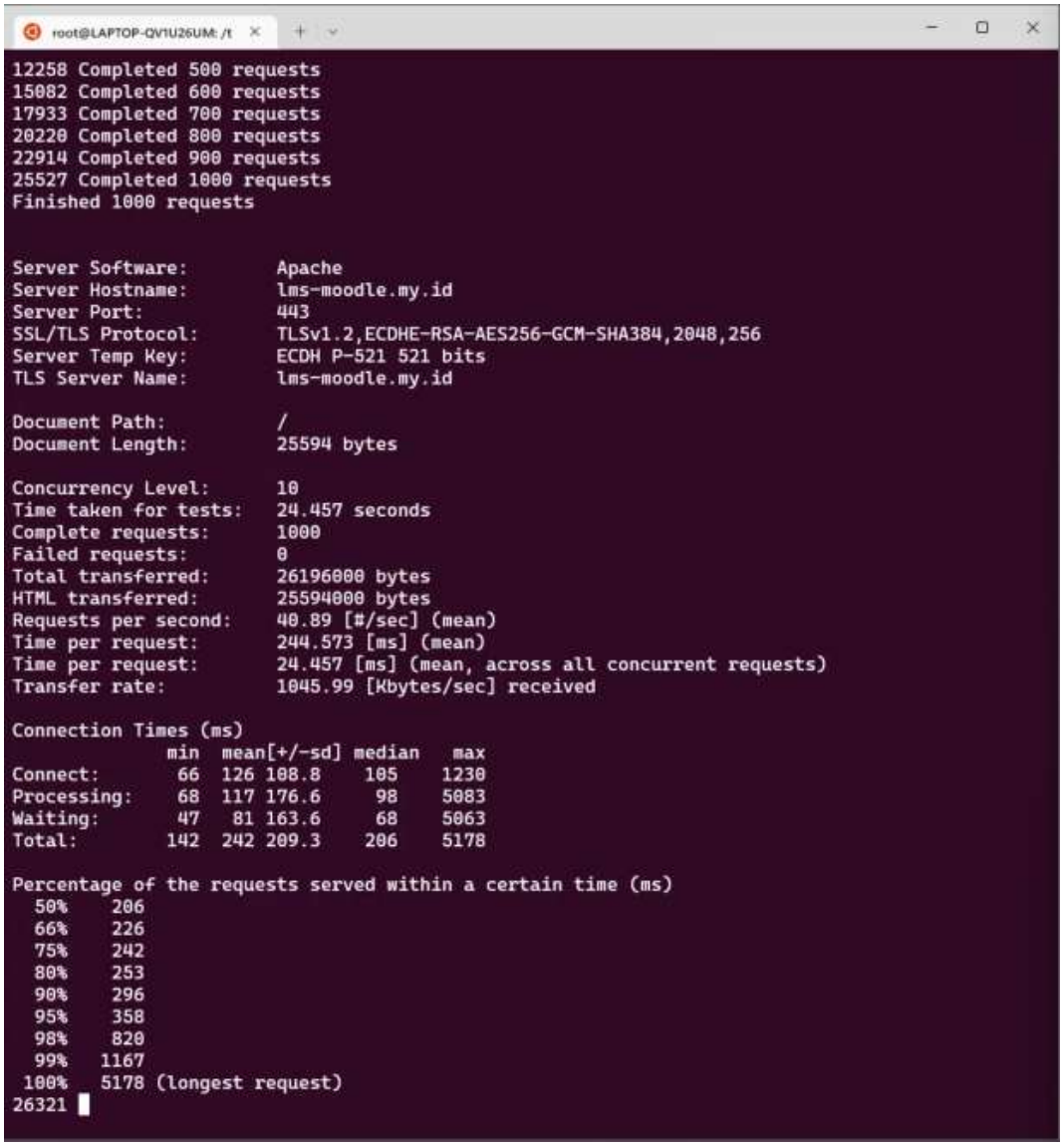
```
“p1 = subprocess.Popen("ab -n 1000 -c 10 https://lms-moodle.my.id/",
shell=True)”

“p2 = os.system("tshark -i any -w " + filename)”
```

Penggunaan *wireshark* untuk mendapatkan hasil pengujian berupa parameter *throughput*, *latency*, *jitter*, dan *packet loss*. Diagram blok alur pengujian akan digambarkan pada Gambar 3.16.



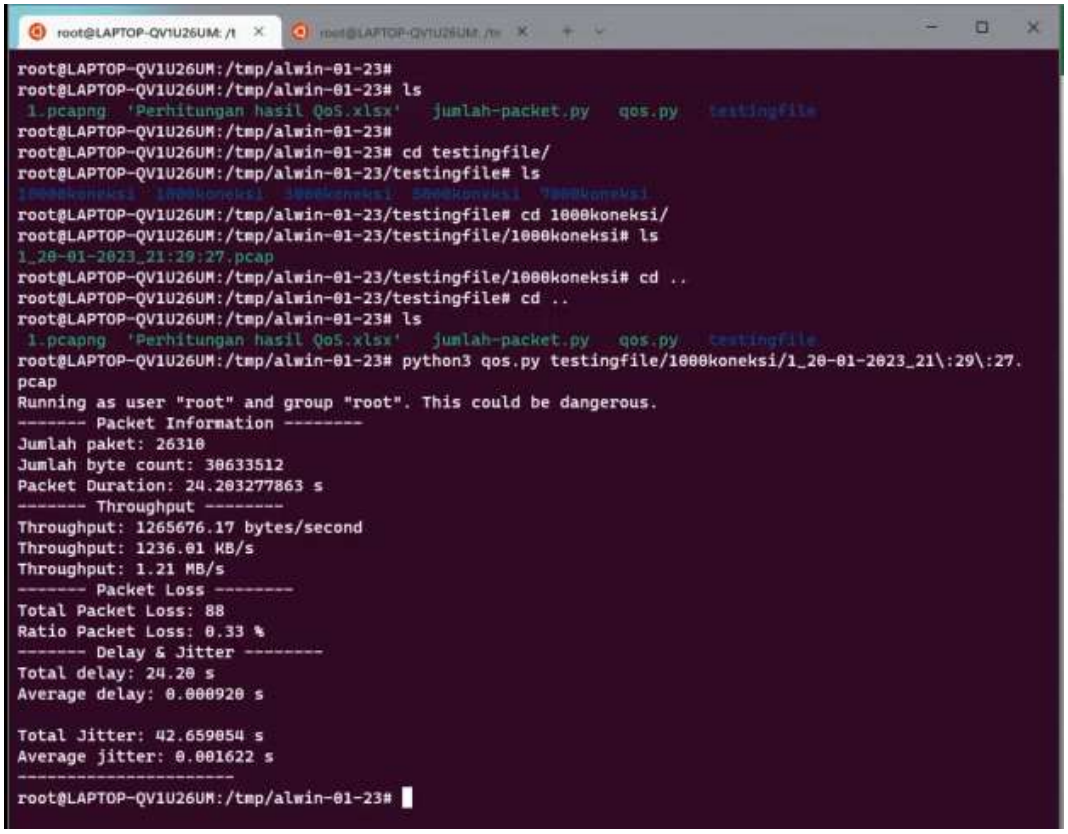
Gambar 3. 16 Diagram blok alur pengujian

Selanjutnya melakukan pengujian dengan *Apache Benchmark* yang akan mengirimkan 1000, 3000, 5000, 7000, dan 10000 permintaan dengan 10 *request rate* per detik. Sebagai contoh pengujian menggunakan *Apache Benchmark* dengan jumlah 1000 permintaan akan ditampilkan pada Gambar 3.17.



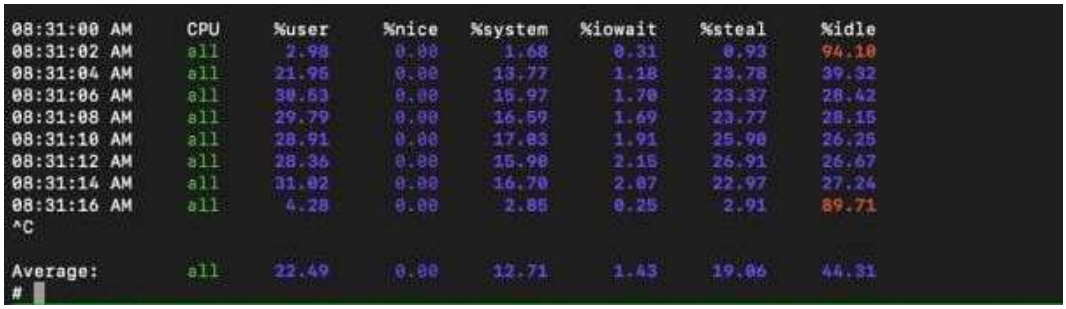
Gambar 3. 17 Contoh pengujian apache benchmark 1000 permintaan

Selanjutnya menggunakan script “qospy” yang di dalamnya memuat perintah untuk menghasilkan nilai QoS yang di-*capture* menggunakan *wireshark* pada *server* LMS. Proses ini dilakukan untuk melihat pembagian beban ke setiap *server* ketika *request* dikirimkan. Contoh hasil pengujian nilai QoS pada saat 1000 permintaan akan ditampilkan pada Gambar 3.18.



Gambar 3. 18 Contoh hasil QoS pada 1000 permintaan

Selanjutnya dengan melakukan pemantauan di setiap *server* dengan menggunakan perintah “sar –u 2”. Pemantauan ini dilakukan untuk melihat pembagian beban ke setiap *server* ketika permintaan dikirimkan. Contoh pemantauan CPU pada saat 1000 permintaan akan ditampilkan pada Gambar 3.19.



Gambar 3. 19 Contoh pemantauan CPU pada saat 1000 permintaan