BAB II
DASAR TEORI

2.1 KAJIAN PUSTAKA

Penelitian Albert Yakobus Chandra pada tahun 2019 yang bejudul tentang "Analisis Performansi Antara Apache & Nginx *Web server* dalam Menangani *Client Request*" membahas tentang perbandingan performa antara *web server* yang berbasis Apache dengan *web server* yang berbasis Nginx dalam menangani *client request*. Kedua *web server* berjalan di atas sistem operasi CentOS dengan menggunakan virtualisasi yang berbasis *container*. Pada penelitian ini difokuskan untuk menguji dan membandingkan total waktu yang dibutuhkan oleh kedua *web server* dalam menyelesaikan seluruh *request*. Dari hasil uji yang dilakukan pada *kedua web server* tersebut menujukan bahwa Nginx memiliki rata-rata waktu penyelesaian *request* yang lebih cepat dibandingkan dengan Apache. Hasil ini didapatkan setelah proses pengujian dengan jumlah *request* mulai dari 100 sampai 1.000.000 dengan menggunakan *tool* Apache Bench [9].

Penelitian Ardian Dwi Praba dan Hariyanto pada tahun 2020, yang berjudul "Performansi *Web server* Apache dan Nginx Pada Aplikasi penjualan *Online*", melakukan pengujian terhadap *web server* berbasis Apache dengan *web server* berbasis Nginx. Penlitian ini bertujuan untuk dapat mengetahui *web server* mana yang lebih baik untuk digunakan pada aplikasi penjualan *Online*. Pengujian ini berfokus pada kecepatan waktu yang digunakan user secara bersamaan untuk mengakses sebuah *website*. Berdasarkan hasil pengujian, didapatkan bahwa *web server* Apache memiliki hasil waktu 9,608 detik untuk 500 *request*, 84,387 detik untuk 5.000 *request*, 159,994 detik untuk 10.000 *request*, dan 234,430 detik untuk 15.000 *request*. *Web server* Nginx memiliki hasil waktu 3,211 detik untuk 500 *request*, 31,253 detik untuk 5000 *request*, 59,703 detik untuk 10.000 *request*, dan 96,704 detik untuk 15.000 *request*. Berdasarkan hasil dari penelitian, *web server* Nginx lebih unggul dalam menangani *request* dari *client* dibandingkan dengan Apache [10].

Penelitian I Kadek Susila Satwika dan Ketut Ngurah Semadi pada tahun 2020 yang berjudul "Perbandingan Performansi *Web server* Apache Dan Nginx

Dengan Menggunakan Ipv6", bertujuan untuk menguji *web server* berbasis Nginx dan Apache dengan jenis pengalamatan IPv6. Masing-masing *server* akan dikonfigurasi dengan menggunakan IPv6 sebagai alamat identik dari *server* virtual. Parameter yang dianalisis yaitu *Time Taken for Tests, Request Per Second, Transfer Rate (Kb/s), Time per Request (ms), Memory Usage , dan Load Maximum*. Dari hasil pengujian, *web server* dengan menggunakan Nginx memiliki performansi yang lebih baik dibandingkan dengan Apache. Hal ini didapatkan berdasarkan parameter *Time Taken for Tests, Request Per Second, Transfer Rate* pada *server* Nginx menghasilkan hasil yang lebih baik dibandingkan Apache. Namun, *server* Nginx lebih efisien dibandingkan Apache dari sisi penggunaan *memory* dan *load* [4].

Penelitian Abdul Aziz dan Topan Tampati pada tahun 2015, yang berjudul "Analisis *Web server* untuk Pengembangan Hosting *Server* Institusi: Pembandingan Kinerja *Web server* Apache dengan Nginx", menguji performa kedua *web server* untuk mengetahui kinerja yang dihasilkan oleh masing-masing *web server*. Pengujian ini berfokus pada waktu yang digunakan untuk mengakses sebuah *web* secara bersamaan oleh beberapa pengguna. Pengujian dilakukan pada *web server* berbasis *Ubuntu server*. Parameter yang dianalisis mulai dari waktu *request*, waktu transfer, dan waktu koneksi. 5 halaman *website* akan diuji pada kedua *web server*. Berdasarkan hasil pengujian, *web server* Apache memiliki kecepatan transfer rate, time per *request* dan connection time lebih cepat dibandingkan dengan Nginx, di mana dengan nilai transfer rata-rata dari Apache 701 Kbytes/sec sedangkan Nginx 522 Kbytes/sec [2].

Penelitian oleh Lukman dan Mayang Yudhiastari pada tahun 2021 yang berjudul "Analisis Kinerja *Web server* Apache Dan Litespeed Menggunakan Httperf Pada Virtual Private *Server* (VPS)", membahas mengenai perbandingan kinerja *web server* Apache dan Litespeed menggunakan Httperf pada *Virtual Private Server* (VPS) dengan sistem operasi CentOS. Penelitian dilakukan dengan cara memberikan beban yang sama kepada kedua *web server* dan membandingkan hasil yang diperoleh untuk mengetahui *web server* mana yang memiliki kinerja lebih baik. Parameter pengujian meliputi *throughput, Connection, dan reply*. Dalam pengujian secara bersamaan ataupun secara individu, *web server* Apache

menghasilkan nilai *throughput, connection, dan reply* yang lebih besar dibandingkan *web server* Litespeed pada pengujian menggunakan subjek halaman *web* 1 serta subjek halaman *web* 2 dengan 100, 500, 1000 *connection* dan 10, 50, 100 rate/detik. Hasil pengujian menunjukkan bahwa kinerja *web server* Apache lebih baik dilihat dari hasil keseluruhan nilai *throughput* yang lebih besar dari *web server* Litespeed. Sedangkan *web server* Litespeed lebih baik dilihat dari hasil keseluruhan nilai *connection,* dan *reply* yang lebih kecil dari *web server* Apache [11].

Penelitian Intan Ferina Irza , Zulhendra, dan Efrizon pada tahun 2017 yang berjudul "Analisis Perbandingan Kinerja *Web Server* Apache dan Nginx Menggunakan Httperf Pada Portal Berita (Studi Kasus beritalinux.com)", membahas mengenai perbandingan kinerja Apache dan Nginx berdasarkan parameter *throughput, connection, request, reply*, dan *error* menggunakan Httperf. Pengujian dilakukan menggunakan beban *connection* sebesar 100, 500, dan 1000 dengan *rate/second* setiap beban sebesar 10, 50, dan 100. Dari hasil pengujian didapatkan bahwa Apache lebih baik berdasarkan parameter *throughput* sedangkan Nginx lebih baik berdasarkan parameter *connection time*, dan *reply time*. Pada parameter *request* dan *error,* kedua *web server* memiliki kinerja yang sama baiknya [12].

Penelitian Fariq Adnan dan Kusnawi pada tahun 2017 yang berjudul "Analisis Perbandingan Performa *Web Server* Apache Dan Nginx Menggunakan Httperf Pada Vps Dengan Sistem Operasi CentOS", membahas mengenai perbandingan kinerja Apache dan Nginx pada VPS berbasis sistem operasi CentOS menggunakan Httperf. Parameter yang dianalisis adalah *reply time* dan *throughput*. Pengujian dilakukan dengan variasi beban pengujian jumlah koneksi (*number connection*) sebesar 100, 500, dan 1000 sedangkan jumlah permintaan per detik (*rate/second*) sebesar 10, 50, dan 100. Subjek penelitian meliputi *web* statis, Gambar, PHP, Wordpress, dan Toko *Online*. Berdasarkan hasil pengujian, Nginx unggul pada pengujian *web* statis, dan *throughput* pada pengujian Wordpress. Sedangkan Apache unggul pada pengujian PHP dan pada *bandwidth* subjek *web* statis, Gambar, PHP, dan Toko *Online* [13].

Penelitian Hyundam Yoo, Yonghoon Kim, Choong-Geon Song, Hyung-Eun Kim, dan Byeong-Jun Choi pada tahun 2018 yang berjudul "*A Study on The Comparative Performance Analysis of Open Source Web Server Using JMeter*", membahas mengenai perbandingan *web server open source* yaitu Apache, Nginx, Cherokee, Monkey HTTP, dan Sand Storm. Kemudian hasil kinerja dibandingkan dan dianalisis menggunakan program uji kinerja *web server* yaitu JMeter. Pengujian dilakukan dengan mengakes beberapa *file* yaitu index.html dengan ukuran *file* 104 Byte, cat.jpg dengan ukuran *file* 6.8KB dan 4k.jpg dengan ukuran *file* 4.9MB. Selain itu pengujian juga dilakukan dengan meningkatkan jumlah klien. Berdasarkan hasil pengujian, Lighttpd menunjukkan kinerja yang baik dalam hal ukuran *file* kecil, Cherokee dalam hal ukuran *file* sedang, dan Nginx dalam hal ukuran *file* besar. Ketika jumlah klien ditingkatkan, Cherokee menunjukkan penurunan kinerja terkecil dan Lighttpd menunjukkan penurunan kinerja terbesar [14].

Sebagai upaya untuk memperkuat dasar teori yang hendak digunakan pada penelitian ini, peneliti telah merangkum penelitian penulis dan beberapa referensi penelitian terdahulu terkait analisis kinerja *web server* pada Tabel 2.1.

Tabel 2. 1 Perbandingan dengan Penelitian Sebelumnya.

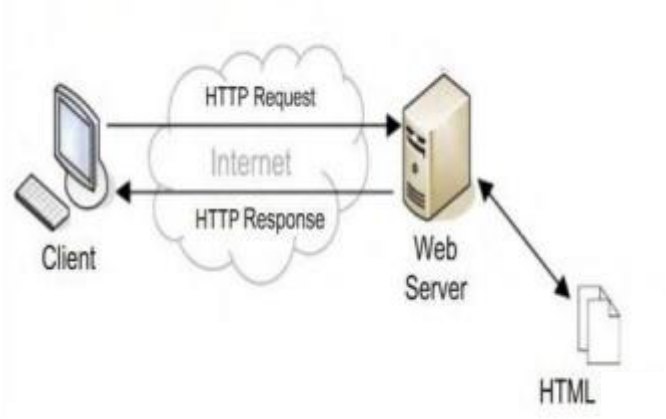
No	Peneliti	Metode	Parameter Kinerja	Variabel Pengujian	Sistem Operasi	Load Testing Tool
1	Albert Yakobus Chandra	*Load Testing Single URL*	*Elapsed Time (s)*	*Concurrency, Repetition*	CentOS	Apache Bench
2	Ardian Dwi Praba and Hariyanto	*Load Testing Single URL*	*Elapsed Time (s)*	*Concurrency, Repetition*	Ubuntu	Apache Bench
3	I Kadek Susila Satwika and Ketut Ngurah Semadi	*Load Testing Single URL*	*Elapsed Time (s), request per second, Transfer Rate (Kb/s) , Time per Request (ms)*	*Concurrency, Repetition*	Ubuntu	Apache Bench

No	Peneliti	Metode	Parameter Kinerja	Variabel Pengujian	Sistem Operasi	Load Testing Tool
4	Abdul Aziz dan Topan Tampati	*Load Testing Single URL*	*Time per Request (ms), Transfer Rate (Kb/s) , Connection Times (s)*	*Concurrency, Repetition, URL*	Ubuntu	Apache Bench
5	Lukman and Mayang Yudhiastari	*Load Testing Single URL*	*Throughput (KB/s), Connection (ms), Reply (ms)*	*Number Connection, Rate Per Second, URL*	CentOS	Httperf
6	Intan Ferina Irza , Zulhendra, and Efrizon	*Load Testing Single URL*	*Throughput, Connection, Request, Reply, Error.*	*Number Connection, Rate Per Second, URL*	-	Httperf
7	Fariq Adnan dan Kusnawi	*Load Testing Single URL*	*Throughput (KB/s), Reply (ms)*	*Number Connection, Rate Per Second, URL*	CentOS	Httperf
8	Hyundam Yoo dkk.	*Load Testing Single URL*	*Deviasi, Throughput (B/s)*	*Concurrency, Repetition, URL*	-	Jmeter
9	Rendy Patra Julriansyah	*Load Testing Single URL dan Load Testing Multiple URL*	*Response Time (s), Transaction Rate (transaction/s), Throughput (MB/s), Elapsed Time (s)*	*Concurrency, Repetition, Delay, Time dan URL*	Ubuntu	Siege

2.2 DASAR TEORI

2.2.1 *Web Server*

Secara perannya sebagai perangkat lunak, *web server* adalah *software* yang melayani permintaan pengguna/*client* yang terhubung dalam jaringan dengan memberikan informasi yang diminta sehingga dapat ditampilkan oleh *browser* pengguna dengan bentuk halaman *web* [2]. Sebagai perangkat keras, *web server* merupakan tempat atau perangkat untuk menyimpan konten *website* [4]. Cara kerja *web server* secara sederhana dapat dilihat pada Gambar 2.1.



Gambar 2.1 Ilustrasi Cara Kerja *Web server* [14]

Client dengan *web server* berkomunikasi menggunakan protokol bernama *Hyper Text Transfer Protocol* (HTTP). Pada sisi *client* terdapat *web browser atau browser* yang digunakan untuk berkomunikasi dengan *web server* di mana *web browser* yang berfungsi sebagai antarmuka untuk meminta informasi yang diinginkan kepada *web server* dan juga menampilkan informasi yang dikirim oleh *web server*. Untuk melakukan *request*, *client* memasukkan alamat situs (alamat *web server* yang menyimpan informasi) pada *browser* yang akan mencoba membentuk koneksi ke *web server* sesuai alamat situs dengan menggunakan protokol HTTP *request*. *Browser* meminta (*request*). *Web server* melayani *request* dengan mengirimkan informasi yang umumnya berbentuk *Hypertext Markup Language* (HTML), proses ini disebut HTTP *response*. *Browser* yang telah menerima informasi kemudian menampilkan hasilnya dalam bentuk halaman *web* [15].

2.2.2 Linux

Linux adalah sistem operasi yang diciptakan oleh Linus Trovalds dibawah lisensi General Public License (GPL). Distro linux merupakan penyebaran paket-paket Linux diantaranya Red Hat, Fedora, Debian, SUSE, Mandriva, Ubuntu, dan lain sebagainya. Salah satu kelebihan Linux yaitu tahan terhadap virus menjadi salah satu penyebab Linux digunakan menjadi sistem operasi pada *server*, selain itu Linux mudah didapat, terjangkau, *open source*, dapat berjalan pada sistem 32 bit dan 64 bit, ringan dengan hanya menggunakan CLI (Command Line Interface) bukan GUI (Graphical User Interface). Linux memiliki struktur direktori yang membentuk pohon dalam organisasi file dan direktori sebagimana yang ditunjukkan

pada Gambar 2.2. Seluruh direktori berada dibawah satu direktori paling tinggi yang disebut root (/) [16].

Gambar 2.2 Struktur Direktori pada Sistem Operasi Linux [16]

Berikut merupakan penjelasan dari setiap direktori Linux pada Gambar 2.2.

a) /bin dan /sbin, tempat menyimpan program file binary.
b) /boot, tempat menyimpan file booting Linux.
c) /root, direktori home yang dikhususkan untuk user root.
d) /etc, tempat menyimpan file-file konfigurasi sistem.
e) /home, direktori tempat menyimpan direktori home user biasa.
f) /dev, berisi file yang merupakan refleksi hardware yang dikenali sistem.
g) /usr, direktori tempat menyimpan library, binary, dokumentasi, dan file hasil instalasi user.
h) /mnt, direktori tempat menyimpan direktori-direktori mount point.
i) /tmp, direktori tempat menyimpan aplikasi yang sedang berjalan.
j) /var, direktori tempat menyimpan log, mailbox, dan data aplikasi.
k) /lib, direktori tempat menyimpan library yang mendukung kerja kernel.
l) /proc, direktori tempat untuk menyimpan file yang menunjukan data-data kernel.

2.2.3 *Website*

Website adalah kumpulan dari halaman *web* yang berupa teks, gambar, atau data yang dipublikasikan dijaringan internet dan memiliki domain yang dapat di akses semua pengguna internet dengan cara mengetikan alamatnya. Secara singkat

website adalah kumpulan halaman yang menyediakan informasi (Irwansya, 2014) [10]. Penemu *website* adalah Sir Timothy John, sedangkan *website* yang tersambung dengan jaringan, pertama kali muncul pada tahun 1991. Maksud dari pembuatan *website* adalah untuk mempermudah tukar-menukar dan memperbarui informasi kepada sesama peneliti ke tempat dia bekerja. *Website* atau situs *web* dapat diartikan sebagai kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya yang saling terkait dan dihubungan dengan jaringan halaman. Hubungan antara satu halaman *website* dengan halaman *website* yang lainnya disebut *Hyperlink*, sedangkan teks yang dijadikan media penghubung disebut *Hypertext*. *Hypermedia* adalah media yang tidak hanya memuat teks saja, namun juga foto, audio dan video dan grafis komputer yang saling terhubung. Istilah *Hypermedia* merupakan istilah yang diciptakan oleh Ted Nelson. Contoh klasik *Hypermedia* adalah *World Wide Web*, karena adanya *hyperlink* [17].

2.2.4 *Hypertext Transfer Protocol* (HTTP)

Hypertext Transfer Protocol (HTTP) merupakan protokol komunikasi yang digunakan untuk layanan world wide *web* (www). Pengembangan HTTP dikoordinasi oleh konsorsium www dan *Internet Engineering Task Force* (IETF). Dipublikasikan melalui kumpulan *Request* For Comments (RFC). RCF 2616 mendefinisikan HTTP/1.1 yang merupakan versi HTTP yang saat ini umum digunakan. HTTP *client* mengirimkan permintaan atau *request* dengan membuat koneksi *Transmission Control Protocol* (TCP) kepada *server* (umumnya bekerja pada port 80). Disamping itu, HTTP *server* menunggu permintaan atau pesan *request* pada port yang telah ditentukan. Setelah menerima *request* dari *client*, *server* akan mengirimkan status line yaitu "HTTP/1.1 200 OK". Kemudian dilanjutkan dengan mengirimkan dokumen atau *file* yang diminta *client* beserta pesan kesalahan dan informasi lainnya. HTTP diidentifikasi menggunakan *Uniform Resource Identifier* (URI) dengan format penulisan yang telah ditentukan. Protokol HTTP bersifat *request*-respon (permintaan-balasan), artinya dalam protokol ini *client* mengirimkan pesan *request* ke *web server*, dan *web server* akan memberikan balasan atau respon yang sesuai dengan permintaan atau *request* tersebut. *Request*

dan respon dalam protokol HTTP disebut sebagai *request* chain dan respon chain. Hubungan HTTP paling sederhana terdiri atas hubungan langsung antara *client* dengan *server* [18].

2.2.5 Apache

Apache merupakan aplikasi *web server open source* berbasis Unix dari Apache Software Foundation (www.apache.org). Apache tersedia dalam berbagai versi sistem operasi termasuk Unix dan Windows, dan sampai saat ini Apache merupakan HTTP *server* yang paling banyak digunakan di internet. Apache merupakan perangkat lunak *open source* yang dikembangkan oleh komunitas terbuka yang terdiri dari pengembang-pengembang di bawah naungan Apache Software Foundation. Halaman resmi Apache menerangkan, sejarah pengembangan Apache dimulai oleh veteran *developer* NCSA httpd (National Center for Supercomputing Application). Pengerjaan *web server* NCSA dibiayai oleh pemerintah Amerika. maka *web server* Apache dikembangkan oleh sekelompok programmer yang bekerja secara sukarela [19].

Secara umum arsitektur apache bekerja dengan model *thread-based* atau berbasis proses. *Web server* berbasis proses menggunakan proses (*thread*) untuk menerima dan merespon permintaan. Setiap permintaan akan diciptakan sebuah *thread* yang disimpan dalam sebuah *pool* pada alokasi memori tertentu dan dilakukan proses untuk menjawab, setelah proses dieksekusi kemudian hasil proses dikirimkan kembali ke klien serta dicatat dalam sebuah *log* [20].

2.2.6 Nginx

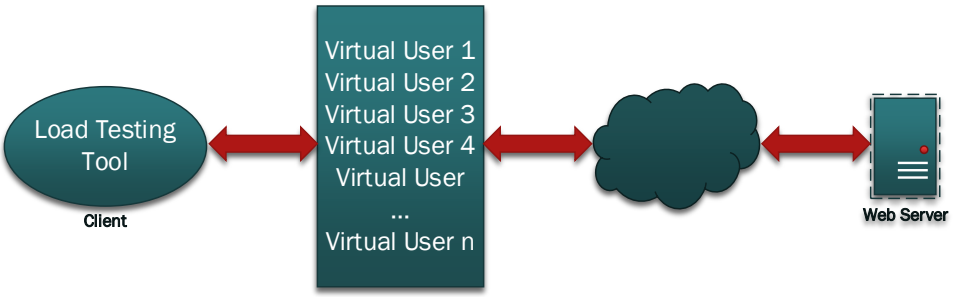
Secara umum *web server* nginx menerapkan model *event-based*. Model *event-based* hanya menggunakan sebuah *thread* untuk memproses permintaan dan mengelola *event* dengan menggunakan *multiplexing* dan banyak notifikasi *event*. Setiap koneksi diproses secara sangat efisien dalam sejumlah proses *single threaded* yang disebut *workers*. Dalam setiap nginx, *workers* dapat menangani ribuan koneksi bersamaan dan permintaan per detik [20].

NGINX merupakan sebuah *web server* yang *opensource* yang dapat digunakan juga sebagai *proxy* IMAP/POP3, Nginx banyak dipakai karena stabil,

konsumsi sumber daya rendah dan memiliki performa yang bagus. Nginx memiliki beberapa fitur unggulan yaitu *reverse proxy multiple protocols* seperti SCGI, UWSGI dan memiliki *cache* sedangkan untuk video bisa mendukung seperti FLV, HLS dan mp4 dan http/2 *gateway* serta bisa menjadi *load balancer*. *Opensource code* Nginx pertama ditulis oleh orang Rusia yang bernama Igor Sysoev pada tahun 2002 dan dirilis ke publik pada tahun 2004 untuk menjawab permasalahan C10K. Nginx menggunakan arsitektur *event-driven* dan asinkron, di mana *thread* yang sama akan di proses dalam sebuah bagian yaitu *worker process* dan *worker process* sendiri terdiri atas *worker connection*. Dan semuanya bertujuan untuk menangani *request* yang muncul dari *thread*. *Worker connection* akan mengirimkan permintaan kepada *worker process* dan kepada *master process*, *worker connection* bisa menampung sampai 1024 *request* yang sama. Sedangkan bagian yang untuk menghasilkan hasil akhir adalah *master process*, hasil yang keluar merupakan hasil dari permintaan atau *request* tersebut [21].

2.2.7 *Load Testing*

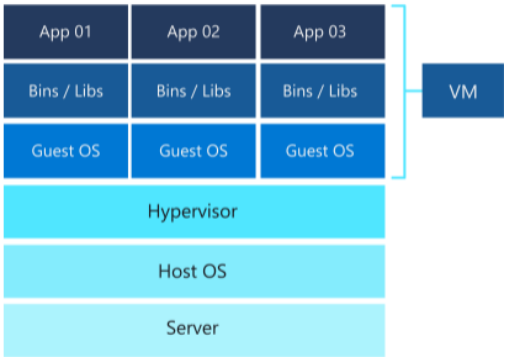
Load testing merupakan salah satu teknik pengujian kinerja yang mengukur respons sistem di bawah berbagai kondisi beban. Tes ini membantu menentukan bagaimana *server web* berperilaku ketika beberapa pengguna mengakses *server web* secara bersamaan. *Load testing* diperlukan untuk mensimulasikan akses simultan ke *web*/*website*. Cara ini lebih baik daripada mengundang puluhan atau puluhan orang untuk mengunjungi *website* Anda secara bersamaan. *Load testing* adalah proses pengujian perangkat lunak non-fungsional yang menguji kinerja aplikasi perangkat lunak di bawah beban tertentu yang diharapkan. Menentukan perilaku aplikasi perangkat lunak saat diakses oleh banyak pengguna secara bersamaan. Tujuannya adalah untuk mengetahui kinerja dan memastikan stabilitas aplikasi perangkat lunak [22]. Salah satu cara untuk menilai kinerja infrastruktur TI (dalam hal ini *web server*) adalah melalui *load testing* yang memungkinkan untuk menilai caranya situs *web* mendukung beban kerja yang diharapkan dengan menjalankan serangkaian skrip tertentu yang meniru perilaku pelanggan pada tingkat beban yang berbeda [6]. Ilustrasi sederhana proses *load testing* dapat dilihat pada Gambar 2.3.



Gambar 2.3 Ilustrasi Sederhana *Load Testing*

2.2.8 *Virtual machine*

Konsep dari virtualisasi adalah membagi sumber daya *native server* menjadi beberapa computer/*server* secara bersama-sama atau yang dinamakan *computer virtual* atau *virtual machine* (VM). Komputer virtual ini berdiri sendiri tanpa mempengaruhi komputer virtual lainnya walaupun berada dalam 1 *native server*. Walaupun *server* virtual ini berada dalam 1 *native server* namun penggunaan dayanya seperti CPU, RAM dan *Hardisk* dapat di konfigurasikan sehingga di dapat hasil yang lebih efisien dalam penggunaan sumber daya pada setiap layanan. Setiap komputer virtual ini dapat menjalankan sistem operasi yang berbeda-beda seperti Windows dan Linux secara bersamaan, sistem operasi yang berada pada komputer virtual dinamakan *Guest Operating System*. setiap intruksi yang di berikan oleh komputer virtual ini akan langsung di teruskan ke *native server* sehingga prosesnya sama seperti menjalankan sistem operasi pada *native server* secara langsung [23]. Salah satu contoh perangkat lunak *virtual machine* adalah Virtual Box. Gambar 2.4 menunjukkan arsitektur *virtual machine*.



Gambar 2.4 Arsitektur *Virtual machine* [24]

2.2.9 Siege

Siege merupakan sebuah HTTP *load testing* dan utilitas *benchmarking*. Siege dirancang unutuk memungkinkan pengembang *web* mengukur kode mereka di bawah suatu tekanan, untuk melihat bagaimana kode itu akan bertahan untuk dapat dimuat pada internet. Siege mendukung autentikasi dasar, *cookies,* HTTP, HTTPS, dan protocol FTP. *Load testing* menggunakan Siege dapat dilakukan dengan menjalankan berbagai *load condition*. *Load condition* merupakan kondisi pengujian yang akan dijalankan pada saat pengujian menggunakan Siege. Berikut merupakan parameter *load condition* yang digunakan [8].

a. -c (*Concurrent*) : Jumlah *virtual user* yang digunakan untuk mengakses suatu halaman *web* secara bersamaan.
b. -r (*Repetition*) : Jumlah pengulangan yang dilakukan *virtual user* dalam mengakses suatu halaman *web*.
c. -*t* (*Time*) : Total waktu yang diinginkan untuk menjalankan suatu *load testing*.
d. -d (*Delay*) : Interval waktu antara satu *virtual user* dengan *virtual user* lain dalam mengakses halaman *web* yang dituju.
e. -f (*File*) : *File* konfigurasi berfungsi untuk menjalankan *load testing* yang dapat mengakses beberapa URL dalam satu perintah. File konfigurasi berisi kumpulan URL yang dapat dikonfigurasi manual untuk mengakses halaman *web* yang dinginkan secara bersamaan dalam satu pengujian.
f. -I (*Internet*) : Parameter yang digunakan untuk mengakses beberapa halaman *web* secara acak. Parameter ini sebagai simulasi nyata dari *client* yang mengakses suatu halaman *web* yang dinginkan pada suatu *website.* Parameter ini digunakan dengan parameter -f untuk dapat mengakses lebih dari satu URL secara acak dalam satu pengujian [8].

2.2.10 Parameter Kinerja *Load Testing*

Pada penellitian ini, parameter kinerja yang digunakan adalah *response time, transaction rate, throughput,* dan *elapsed time*. *Response time* merupakan jumlah waktu yang dibutuhkan aplikasi *server* untuk mengembalikan hasil

permintaan kepada pengguna [25]. Menurut Jakob Nielsen, ada 3 *response time limit* atau batas waktu respon yaitu 0,1 detik, 1 detik dan 10 detik. *Response time* dapat dikatakan sangat baik jika bernilai kurang lebih 0,1 detik. *Response time* dapat dikatakan baik jika bernilai kurang lebih 1 detik. Jika nilai *response time* berada diantara 1 sampai 10 detik, dapat dikatakan nilai tersebut cukup namun pengguna akan berharap waktu respon yang lebih cepat. Setelah 10 detik, maka nilai *response time* dapat dikatakan buruk [26].

Menurut LoadNinja by Smartbear, parameter kinerja pada *load testing* seperti *response time*, *transaction rate* (*request per second*), *throughput*, dan *elapsed time* (*duration*) merupakan parameter kinerja yang perlu diperhatikan. *Response time* mengukur jumlah rata-rata waktu yang berlalu antara permintaan awal klien dan byte terakhir dari respon *server*, termasuk pengiriman HTML, gambar, CSS, JavaScript, dan sumber daya lainnya. Ini adalah pengukuran standar paling akurat dari pengalaman pengguna yang sebenarnya. *Transaction rate mengukur jumlah permintaan yang dikirim ke server setiap detik, termasuk permintaan untuk halaman HTML, lembar gaya CSS, dokumen XML, file JavaScript, gambar, dan sumber daya lainnya. Throughput* mengukur jumlah *bandwidth*, dalam kilobyte per detik, yang dikonsumsi selama pengujian. *Throughput* yang rendah dapat menyarankan kebutuhan untuk mengompresi sumber daya. Parameter kinerja *load testing* paling dasar adalah *elapsed time* atau durasi, yang mengukur waktu untuk menyelesaikan suatu pengujian [27].