BAB II

LANDASAN TEORI

2.1. Tinjauan Pustaka

Menurut penelitian yang dilakukan oleh Jun, Bae, dan Soh (2011), seiring dengan semakin populernya *web service* dan *smart phone*, jumlah konstruksi *web server* semakin bertambah. Sebelumnya, jumlah *web server* ditentukan berdasarkan kapasitas akses dari sebuah *server* fisik dan jumlah maksimum pengguna yang mengakses secara bersamaan. Pada penelitiannya, Jun, Bae, dan Soh (2011) membandingkan performa antara x86_64 *based load balancing virtual server* dengan VirtualBox, *Kernel Virtual Machine* (KVM), dan *server* fisik yang berjalan secara mandiri (tanpa *load balancing*).

Dalam perancangannya, Jun, Bae, dan Soh (2011) menggunakan beberapa *tools* untuk melakukan *benchmark* atau penilaian terhadap Kinerja sistem diatas, yaitu:

a. ab (*Apache HTTP Server Benchmarking Tools*)
b. httperf (*http performance measurement tools*)
c. *siege*

Ketiga *tools* diatas digunakan untuk mengambil parameter hasil pengujian berupa *time per request, reply time, dan concurrency*. *Hardware* yang digunakan adalah sebuah *server* dengan spesifikasi :

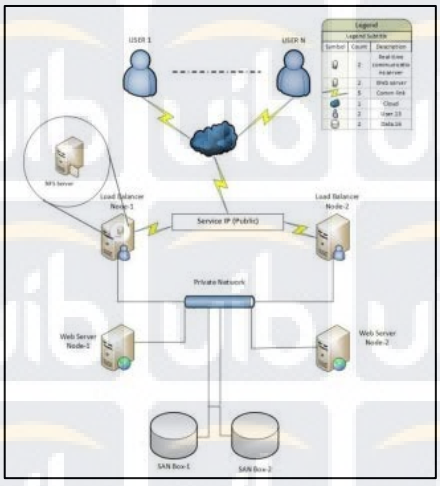
a. CPU : Intel Xeon CPU 2.40 GHz

b. *Memory* : 2x4 GB DDR3 PC-10600 ECC/REG

Sedangkan untuk *Virtual Machine* dan KVM, *server* yang digunakan menggunakan spesifikasi yang sama dengan *server* mandiri, namun dipecah menjadi sistem *virtual* dengan masing – masing *virtual machine* menggunakan 1GB RAM.

Dari hasil pengujian yang dilakukan Jun, Bae, dan Soh (2011), dapat diambil kesimpulan bahwa *server* yang berjalan secara mandiri memiliki time per *request, reply time*, dan *concurrency* yang lebih baik dibandingkan *multiple server* dengan *VirtualBox* ataupun KVM. Namun, pengujian dengan KVM memiliki hasil yang lebih baik daripada *VirtualBox*, dan tidak terpaut jauh dengan *server* mandiri.

Dalam penelitiannya, Moniruzzaman dam Hossain (2014), melakukan kombinasi antara *load balancing server* dan *high availability* menggunakan *Red Hat*, dengan topologi seperti gambar berikut.



Gambar 2.1 *Two-tier high availability and load balancing*

Dalam penelitian tersebut, Moniruzzaman dam Hossain (2014) memfokuskan pada topologi yang lebih hemat biaya dibandingkan menggunakan topologi *three-tier*. Topologi yang diteliti ini disebut *low-cost two-tier high availability and load balancing*. Fokus dari penelitian ini adalah membangun topologi *server cluster* yang lebih hemat biaya, namun tidak menghilangkan fungsi dan efektifitas dari *load balancing* dan *high availability* itu sendiri. Penelitian ini berhasil membuktikan bahwa topologi ini dapat berjalan secara maksimal, dan tidak ada *downtime* yang dirasakan oleh *user*, dan halaman web dapat dikirimkan oleh *server* secara *round robin*.

Asyanto (2011) melakukan penelitian dari penerapan *load balancing* server pada web *server* Lembaga Minyak dan Gas Bumi (Lemigas), dimana pada penelitian tersebut dilakukan uji coba pengukuran *troughput* perbandingan antara server tunggal dan *load balanced server*. Dalam penelitian tersebut, Asyanto (2011) mendapatkan hasil bahwa *server* dengan *load balancer* memiliki *troughput* dan waktu respon yang lebih baik dibandingkan dengan *server* tunggal.

2.2 Landasan Teori

2.2.1 TCP/IP (*Transfer Control Protocol / Internet Protocol*)

TCP/IP adalah dari lapisan-lapisan protokol. Untuk memudahkan dalam memahaminya maka akan diambil contoh pengiriman email (Alan Nur Aditya, 2011). Dalam pengiriman email yang diperlukan adalah protokol untuk email. Protokol ini mendefnisikan perintah-perintah yang diperlukan dalam pengiriman

email, dan protokol ini juga mengasumsikan bahwa ada hubungan antara terminal yang mengirim dengan terminal yang dituju. Dalam hal ini perintah-perintah tersebut diatur oleh TCP dan IP. TCP mengatur masalah perintah-perintah pengiriman data, mengawasi jalannya data dan memastikan data tersebut sampai ke tujuannya, apabila ada bagian dari data yang tidak mencapai tujuan maka TCP akan mengirimkan ulang. Proses tersebut terus berlangsung sampai data yang dikirimkan sampai ke tujuannya. Apabila ada data yang sangat besar untuk dimuat dalam satu datagram maka TCP akan memecahnya menjadi beberapa data dan kemudian mengirimkan ke tujuan dan memastikan sampai dengan benar. TCP dapat dianggap sebagai suatu pembentuk kumpulan - kumpulan routine (perintah) yang dibutuhkan oleh aplikasi untuk dapat berhubungan dengan terminal lain dalam jaringan. Pada protokol TCP/IP dibagi menjadi 4 lapisan (*layer*), yaitu :

1. *Aplication Layer*

Aplication Layer digunakan pada program untuk berkomunikasi menggunakan TCP/IP. Contoh aplikasi antara lain Telnet dan *File Transfer Protocol* (FTP). Interface yang digunakan untuk saling berkomunikasi adalah nomer port dan socket.

2. *Transport Layer*

Transport Layer memberikan fungsi pengiriman data secara *end to end* ke sisi remote. Aplikasi yang beragam dapat melakukan komunikasi secara serentak. Protokol pada lapisan transport yang paling sering digunakan adalah *Transmission Control Protocol* (TCP), dimana memberikan fungsi pengiriman data secara

connection oriented, pencegahan duplikasi data, *congestion control* dan *flow control*. Protokol lainnya adalah *User Datagram Protocol* (UDP), dimana memberikan fungsi pengiriman *connectionless*, jalur yang tidak reliabel. UDP banyak digunakan pada aplikasi yang membutuhkan kecepatan tinggi dan dapat metoleransi terhadap kerusakan data.

3. *Internetwork Layer*

Internetwork Layer biasa disebut juga *layer* internet atau *layer network*, dimana memberikan "*vitual network*" pada internet. Internet Protocol (IP) adalah protokol yang paling penting. IP memberikan fungsi routing pada jaringan dalam pengiriman data. Protokol lainnya antara lain : IP, ICMP, IGMP, ARP, RARP.

4. Network Interface Layer

Network interface layer disebut juga *layer link* atau *layer datalink*, yang merupakan perangkat keras pada jaringan. Contoh : IEEE802.2, X.25, ATM, FDDI, dan SNA.

2.2.1.1 IP Address

IP address merupakan singkatan dari Internet Protokol (IP) Address atau dalam Bahasa Indonesia berarti alamat internet protokol. Seperti halnya suatu alamat rumah, IP address merupakan suatu cara untuk mengetahui asal atau alamat suatu komputer berupa sistem penomoran masing-masing komputer yang

bersifat unik atau tidak sama. Sistem penomoran itu sendiri terdiri dari empat bagian yang dipisahkan oleh titik contoh : 202.155.245.2 (Feilner, 2006).

Setiap komputer yang terhubung dengan internet memiliki IP address ini. Fungsinya adalah untuk melacak asal komputer tersebut dengan mengetahu asal negara dan kota asal komputer tersebut. Dengan kata lain ketika anda berseluncur di dunia maya misalnya ketika mengirim email, mengklik iklan adsense, atau menghacking komputer orang lain dan sebagainya, maka seseorang diluar sana bisa mengetahui lokasi anda yang sebenarnya dari IP address tersebut (Soemarwanto, 2008).

Ada beberapa cara untuk menyembunyikan IP address atau mengacak IP address, namun penulis tidak menyarankan untuk melakukan hal-hal yang melanggar hukum atau berbuat kecurangan. Cara yang paling mudah adalah dengan menggunakan program yang dapat menyembunyikan IP address anda atau mengacak/merubah IP address anda setiap beberapa menit sekali, seolah-olah anda berasal dari lokasi yang berbeda dari lokasi anda yang sebenarnya. Program tersebut dapat anda dapatkan di internet secara gratisan atau berbayar dengan mencarinya di search engine. Catatan tambahan, gunakanlah program tersebut untuk menjaga privacy anda, dan bukan untuk berbuat hal-hal negatif yang dapat merugikan diri anda sendiri (Soemarwanto, 2008).

Jumlah IP address yang tersedia secara teoritis adalah 255x255x255x255 atau sekitar 4 milyar lebih yang harus dibagikan ke seluruh pengguna jaringan internet di seluruh dunia. Pembagian kelas-kelas ini ditujukan untuk

mempermudah alokasi IP Address, baik untuk *host*/jaringan tertentu atau untuk keperluan tertentu (Feilner, 2006).

IP Address dapat dipisahkan menjadi 2 bagian, yakni bagian *network* (net ID) dan bagian *host* (*host* ID). Net ID berperan dalam identifikasi suatu *network* dari *network* yang lain, sedangkan *host* ID berperan untuk identifikasi host dalam suatu *network*. Jadi, seluruh host yang tersambung dalam jaringan yang sama memiliki net ID yang sama. Sebagian dari bit-bit bagian awal dari IP *Address* merupakan network bit/*network number*, sedangkan sisanya untuk *host*. Garis pemisah antara bagian *network* dan *host* tidak tetap, bergantung kepada kelas *network* (Soemarwanto, 2008).

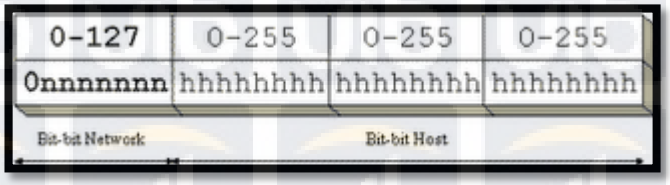
2.2.1.2 Pembagian Kelas IP Address

IP address dibagi ke dalam lima kelas, yaitu kelas A, kelas B, kelas C, kelas D dan kelas E. Perbedaan tiap kelas adalah pada ukuran dan jumlahnya. Contohnya IP kelas A dipakai oleh sedikit jaringan namun jumlah host yang dapat ditampung oleh tiap jaringan sangat besar. Kelas D dan E tidak digunakan secara umum, kelas D digunakan bagi jaringan multicast dan kelas E untuk keprluan eksperimental. Perangkat lunak Internet Protocol menentukan pembagian jenis kelas ini dengan menguji beberapa bit pertama dari IP Address (Soemarwanto, 2008).

1. IP address kelas A

Bit pertama IP address kelas A adalah 0, dengan panjang net ID 8 bit dan panjang host ID 24 bit. Jadi byte pertama IP address kelas A mempunyai range dari 0-127. Jadi pada kelas A terdapat 127 network dengan tiap network dapat

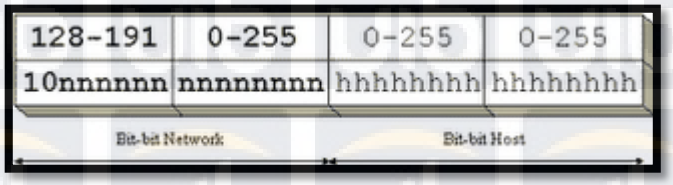
menampung sekitar 16 juta host (255x255x255). IP address kelas A diberikan untuk jaringan dengan jumlah host yang sangat besar, IP kelas ini dapat dilukiskan pada gambar berikut ini:



Gambar 2.2 IP Address Kelas A

2. IP address kelas B

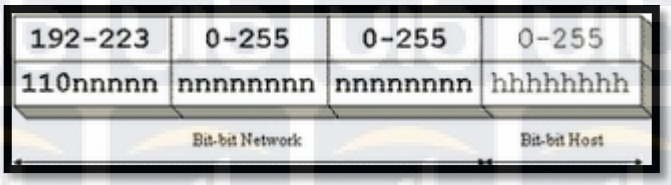
Dua bit IP address kelas B selalu diset 10 sehingga byte pertamanya selalu bernilai antara 128-191. Network ID adalah 16 bit pertama dan 16 bit sisanya adalah host ID sehingga kalau ada komputer mempunyai IP address 167.205.26.161, network ID = 167.205 dan host ID = 26.161. Pada. IP address kelas B ini mempunyai range IP dari 128.0.xxx.xxx sampai 191.155.xxx.xxx, yakni berjumlah 65.255 network dengan jumlah host tiap network 255 x 255 host atau sekitar 65 ribu *host* (Feilner, 2006).



Gambar 2.3 IP Address Kelas B

3. IP address kelas C

IP address kelas C mulanya digunakan untuk jaringan berukuran kecil seperti LAN. Tiga bit pertama IP address kelas C selalu diset 111. Network ID terdiri dari 24 bit dan host ID 8 bit sisanya sehingga dapat terbentuk sekitar 2 juta network dengan masing-masing network memiliki 256 host (Soemarwanto, 2008).



Gambar 2.4 IP Address Kelas C

2.2.1.3 Jenis – jenis IP Address

IP Address dibagi dalam dua jenis yaitu *IP Address Private* dan *IP Address Public*. Berikut penjelasan tentang kedua jenis IP tersebut:

1. *IP address private*

IP address private merupakan alamat-alamat IP yang disediakan untuk digunakan untuk melakukan komunikasi pada jaringan yang tidak terhubung langsung dengan internet. IP address private hanya dapat dipakai untuk komunikasi pada jaringan intranet dan tidak dapat digunakan pada jaringan internet (Wahyudi, 2011)

2. *IP address public*

IP *address public* merupakan alamat-alamat IP yang disediakan untuk digunakan pada jaringan internet. IP *public* dapat diperoleh melalui ISP (*Internet Service provider*) atau penyedia layanan internet , alamat IP ini telah ditetapkan oleh *Inter*NIC dan berisi beberapa buah *network* ID yang tidak mungkin ada yang sama (Astawa & Atmaja, 2012)

2.2.2 *Web Server*

Web server adalah program yang menerjemahkan alamat URL baik dalam bentuk nama *file*, yang kemudian mengirimkan kembali *file* tersebut maupun dalam bentuk nama program dan menjalankan program itu lalu mengirimkan kembali output dari program tersebut melalui internet (Ben & Peter 1999).

Pada dasarnya tugas *web server* hanya ada dua, yaitu:

1. Menerima permintaan (*request*) dari klien, dan
2. Mengirimkan apa yang diminta oleh klien (*response*)

Secara sederhana dapat digambarkan sebagai berikut:

Gambar 2.5 Cara Kerja *Web Server*

Ilustrasi sebelumnya dapat dijelaskan sebagai berikut :

1. Klien disini dapat berupa komputer *desktop* dengan minimal memiliki *browser* dan terhubung ke *web server* melalui jaringan (intranet atau internet).
2. Komputer yang berfungsi sebagai *server*, di dalamnya terdapat perangkat lunak *web server*. Agar komputer ini dapat diakses oleh klien maka komputer harus terhubung ke jaringan (intranet atau internet).
3. Pertama-tama, klien (*user*) akan meminta suatu halaman ke (*we*b) *server* untuk ditampilkan di komputer klien. Misalnya, klien mengetikkan suatu alamat URL di *browser* http://www.google.com. Lalu dengan menggunakan protokol HTTP, *browser* melakukan permintaan dari *web server* Google. Proses permintaan ini disebut dengan *request*.
4. Dari sisi *server* mendapat permintaan halaman utama dari Google, *server* kemudian melakukan pencarian sesuai permintaan. Jika ditemukan maka halaman yang diminta akan dikirim ke klien, namun jika tidak ditemukan, maka *server* akan memberi pesan "404. *Page Not Found*", yang artinya halaman tidak ditemukan.

2.2.3 Apache *Web Server*

Apache merupakan *web server* yang paling banyak dipergunakan di Internet. Program ini pertama kali didesain untuk sistem operasi lingkungan UNIX. Namun demikian, ada beberapa versi berikutnya Apache mengeluarkan programnya yang dapat dijalankan di Windows NT.

Berdasarkan sejarahnya, Apache dimulai oleh veteran developer NCSA *httpd* (*National Center for Supercomputing Application*). Saat itu pengembangan NCSA *httpd* sebagai *web server* mengalami stagnasi. ROB MC COOL meninggalkan NCSA dan memulai sebuah proyek baru bersama para *webmaster* lainnya, menambal *bug*, dan menambahkan fitur pada NCSA *httpd*. Mereka mengembangkan program ini lewat *mailing list*. Dengan berpijak pada NCSA *httpd* versi 1.3, Team Apache mengeluarkan rilis pertama kali secara resmi Apache versi 0.6.2.*File* konfigurasi Apache terletak di direktori /var/apache/conf. Nama *file* konfigurasi tersebut adalah httpd.conf, srm.conf dan access.conf.

httpd.conf merupakan *file* yang dieksekusi pertama kali saat Apache dijalankan. Didalamnya berisi konfigurasi secara umum. srm.conf adalah *file* konfigurasi yang dieksekusi setelah httpd.conf. Disarankan untuk membiarkan *file* konfigurasi ini tetap kosong. Dan access.conf merupakan konfigurasi untuk memfilter *host-host* yang boleh mengakses layanan Apache. Secara *default*, Apache memisahkan *file* konfigurasinya menjadi 3 bagian, yakni httpd.conf, srm.conf dan access.conf.

Kecanggihan model arsitektur Apache adalah kunci keberhasilan permulaan. Pada saat itu, banyak layanan jaringan yang dipicu dari layanan utama yang disebut inetd; ketika jaringan baru (TCP) koneksi diterima, inetd *preforking* () dan exec () proses unix dari jenis yang tepat untuk menangani koneksi. Proses membaca permintaan pada sambungan, dihitung respon dan menulis kembali kepada sambungan, dan kemudian keluar.

Kerugian terbesar adalah penggunaan *forking* sebuah 'httpd' yang memproses pekerja baru untuk setiap koneksi baru, dan pengembang Apache cepat mengadopsi model *prefork* di mana proses pekerja diciptakan dimuka, dan bersedia menerima satu koneksi HTTP baru .

Ketika *prefork* web server Apache menerima koneksi HTTP, salah satu proses pekerja httpd meraih dan menanganinya. Setiap proses ditangani satu koneksi pada satu waktu, dan jika semua proses sedang sibuk, Apache menciptakan proses pekerja lebih siap untuk lonjakan lebih lanjut dalam lalu lintas.

2.2.4 Nginx *Web Server*

Nginx adalah server HTTP dan *reverse proxy* berbasis *open source* berkemampuan tinggi, yang dapat juga digunakan sebagai proxy IMAP/POP3. *Source code* nginx ditulis oleh seorang warga negara Rusia yang bernama Igor Sysoev pada tahun 2002 dan dirilis ke publik pada tahun 2004. Nginx terkenal karena performanya yang tinggi, stabil, memiliki banyak fitur, mudah dikonfigurasi, dan bisa menggunakan spesifikasi hardware yang tidak terlalu tinggi.

Tidak seperti *software server* yang lainnya, nginx tidak bergantung kepada *thread* untuk melayani *client.* Sebaliknya, nginx menggunakan arsitektur *asynchronus* yang lebih stabil. Arsitektur ini membutuhkan lebih sedikit memori serta dapat mengatasi ribuan koneksi pada saat yang bersamaan (Nginx, 2006).

Nginx mampu memberdayakan konten dynamic HTTP disebuah jaringan menggunakan FastCGI, SCGI handlers untuk scripts, aplikasi server uWSGI atau pada modul *Phusion Passenger*, dan dapat bekerja sebagai sebuah piranti lunak penyeimbang beban (*load balancer).* Banyak situs terkenal yang menggunakan Nginx termasuk Wikipedia, WordPress, Fastmail, Ohloh, Sourceforge dan Github. Berdasarkan data yang dikumpulkan Netcraft sekitar 70 juta server terhitung yang ditenagai menggunakan Nginx dan menguasai sekitar 12.49% server web dunia yang paling ramai dikunjungi atau nomor 3 setelah Microsoft IIS (14%) dan penguasa server web Apache yang menguasai 65% kebutuhan dunia.

Dari penciptaan awal, nginx dimaksudkan untuk menjadi alat khusus untuk mencapai kinerja yang lebih, kepadatan permintaan tetapi penggunaan yang ekonomis sumber daya *server* yang dapat melayani pertumbuhan yang dinamis dari sebuah situs web. Karena dalam nginx yang menyampaikan *output file* bukan langsung *memory* sehingga penumpukan data di memory akan berkurang dan server pun akan bekerja lebih cepat dan stabil.

2.2.5 HTTP

Hypertext Transfer Protocol (HTTP) adalah sebuah protokol jaringan lapisan aplikasi yang digunakan untuk sistem informasi terdistribusi, kolaboratif, dan menggunakan hipermedia. Penggunaannya banyak pada pengambilan sumber daya yang saling terhubung dengan tautan, yang disebut dengan dokumen hiperteks, yang kemudian membentuk *World Wide Web* pada tahun 1990 oleh fisikawan Inggris, Tim Berners-Lee. Hingga kini, ada dua versi

mayor dari protokol HTTP, yakni HTTP/1.0 yang menggunakan koneksi terpisah untuk setiap dokumen, dan HTTP/1.1 yang dapat menggunakan koneksi yang sama untuk melakukan transaksi. Dengan demikian, HTTP/1.1 bisa lebih cepat karena memang tidak perlu membuang waktu untuk pembuatan koneksi berulang-ulang.

Pengembangan standar HTTP telah dilaksanakan oleh Konsorsium *World Wide Web* (*World Wide Web Consortium*/W3C) dan juga *Internet Engineering Task Force* (IETF), yang berujung pada publikasi beberapa dokumen *Request for Comments* (RFC), dan yang paling banyak dirujuk adalah RFC 2616 (yang dipublikasikan pada bulan Juni 1999), yang mendefinisikan HTTP/1.1.

Dukungan untuk HTTP/1.1 yang belum disahkan, yang pada waktu itu RFC 2068, secara cepat diadopsi oleh banyak pengembang penjelajah situa pada tahun 1996 awal. Hingga Maret 1996, HTTP/1.1 yang belum disahkan itu didukung oleh Netscape 2.0, Netscape Navigator Gold 2.01, Mosaic 2.7, Lynx 2.5, dan dalam Microsoft Internet Explorer 3.0. Adopsi yang dilakukan oleh pengguna akhir penjelajah situs pun juga cepat. Pada bulan Maret 2006, salah satu perusahaan *Web Hosting* melaporkan bahwa lebih dari 40% dari penjelajah situs yang digunakan di Internet adalah penjelajah situs yang mendukung HTTP/1.1. Perusahaan yang sama juga melaporkan bahwa hingga Juni 1996, 65% dari semua penjelejah yang mengakses *server-server* mereka merupakan penjelajah situs yang mendukung HTTP/1.1. Standar HTTP/1.1 yang didefinisikan dalam RFC 2068 secara resmi dirilis pada bulan Januari 1997.

Peningkatan dan pembaruan terhadap standar HTTP/1.1 dirilis dengan dokumen RFC 2616 pada bulan Juni 1999.

HTTP adalah sebuah protokol meminta/menjawab antara klien dan server. Sebuah klien HTTP (seperti web browser atau robot dan lain sebagainya), biasanya memulai permintaan dengan membuat hubungan ke port tertentu di sebuah server Webhosting tertentu (biasanya port 80). Klien yang mengirimkan permintaan HTTP juga dikenal dengan user agent. Server yang meresponsnya, yang menyimpan sumber daya seperti berkas HTML dan gambar, dikenal juga sebagai origin server. Di antara user agent dan juga origin server, bisa saja ada penghubung, seperti halnya proxy, gateway, dan juga tunnel.

HTTP tidaklah terbatas untuk penggunaan dengan TCP/IP, meskipun HTTP merupakan salah satu protokol aplikasi TCP/IP paling populer melalui Internet. Memang HTTP dapat diimplementasikan di atas protokol yang lain di atas Internet atau di atas jaringan lainnya. seperti disebutkan dalam "implemented on top of any other protocol on the Internet, or on other networks.", tapi HTTP membutuhkan sebuah protokol lapisan transport yang dapat diandalkan. Protokol lainnya yang menyediakan layanan dan jaminan seperti itu juga dapat digunakan.

Sumber daya yang hendak diakses dengan menggunakan HTTP diidentifikasi dengan menggunakan *Uniform Resource Identifier* (URI), atau lebih khusus melalui *Uniform Resource Locator* (URL), menggunakan skema URI http atau https

2.2.6 HTML

HyperText Markup Language (HTML) adalah sebuah bahasa *markup* yang digunakan untuk membuat sebuah halaman situs, menampilkan berbagai informasi di dalam sebuah penjelah situs *internet* dan *formating hypertext* sederhana yang ditulis kedalam berkas *format* ASCII agar dapat menghasilkan tampilan wujud yang terintegerasi. Dengan kata lain, berkas yang dibuat dalam perangkat lunak pengolah kata dan disimpan kedalam format ASCII normal sehingga menjadi *home page* dengan perintah-perintah HTML. Bermula dari sebuah bahasa yang sebelumnya banyak digunakan di dunia penerbitan dan percetakan yang disebut dengan SGML *(Standard Generalized Markup Language).* HTML adalah sebuah standar yang digunakan secara luas untuk menampilkan halaman situs. HTML saat ini merupakan standar internet yang didefinisikan dan dikendalikan penggunaannya oleh *world wide web consortium* (W3C). HTML dibuat oleh kolaborasi Caillau TIM dengan Berners-lee robert ketika mereka bekerja di CERN pada tahun 1989 (CERN adalah lembaga penelitian fisika energi tinggi di Jenewa).

1. Tahun 1980, IBM memikirkan pembuatan suatu dokumen yang akan mengenali setiap elemen dari dokumen dengan suatu tanda tertentu. IBM kemudian mengembangkan suatu jenis bahasa yang menggabungkan teks dengan perintah-perintah pemformatan dokumen. Bahasa ini dinamakan Markup Langiage, sebuah bahasa yang menggunakan tanda-tanda sebagai basisnya. IBM menamakan sistemnya ini sebagai Generalized Markup Language atau GML.

2. Tahun 1986, ISO menyatakan bahwa IBM memiliki suatu konsep tentang dokumen yang sangat baik, dan kemudian mengeluarkan suatu publikasi (ISO 8879) yang menyatakan markup language sebagai standar untuk pembuatan dokumen-dokumen. ISO membuat bahasa ini dari GML milik IBM, tetapi memberinya nama lain, yaitu SGML (*Standard Generalized Markup Language*).

ISO dalam publikasinya meyakini bahwa SGML akan sangat berguna untuk pemrosesan informasi teks dan sistem-sistem perkantoran. Tetapi diluar perkiraan ISO, SGML dan terutama subset dari SGML, yaitu HTML juga berguna untuk menjelajahi internet. Khususnya bagi mereka yang menggunakan *World Wide Web.* Versi terakhir dari HTML adalah HTML 4.01, meskipun saat ini telah berkembang XHTML yang merupakan pengembangan dari HTML.

2.2.7 Siege

Siege adalah sebuah *tools benchmarking open source* untuk menguji performa sebuah web server . Siege dapat melakukan *stress test* dan *load test* kepada sebuah url dengan jumlah koneksi dan waktu yang dapat kita sesuaikan. Siege dirancang untuk memungkinkan pengembang web melihat kinerja sebuah *web server*. Ini memungkinkan penggunanya untuk melakukan pengujian *web server* dengan sejumlah *web browser* simulasi.

Data yang didapat dari pengujian sebuah *web server* menggunakan siege antara lain :

Transactions :	Throughput :
Availability :	Concurrency :
Elapsed time :	Successful transactions :
Data transferred :	Failed transactions :
Response time :	Longest transaction :
Transaction rate :	Shortest transaction :

2.2.8 Stress Testing

Testing atau ujicoba, dalam hal ini adalah proses menjalankan sebuah aplikasi dengan tujuan menemukan permasalahan. (Sharmila & Ramadevi, 2013) Performance testing ini dilakukan dengan melakukan permintaan dalam jumlah besar, seperti mengakses sistem dengan banyak user dalam waktu bersamaan. (Kundu, 2012)

Stress testing adalah bagian dari *Performance* testing. *Performace testing* sendiri memiliki tujuan untuk mengevaluasi kemampuan dari suatu sistem dalam menangani sebuah permintaan atau *request*. Adapun tujuan dari stress testing dan *performance* testing pada umumnya adalah *response time* atau *latency*, *troughput*, utilisasi sumberdaya, dan *workload*.

1. *Throughput* (MB/s)

   Parameter ini mengukur rata – rata keluaran bandwidth dari setiap pengujian, semakin besar nilainya semakin baik.

2. *Response Time* (second)

   Mengukur rata – rata waktu respon dari setiap permintaan pengujian. Semakin kecil semakin baik.

3. *Longest Transactions* (second)

   Mengukur waktu transaksi terlama dari setiap pengujian. Semakin kecil semakin baik.

4. *Transaction Rate* (trans/s)

   Mengukur kemampuan rata – rata server dalam menangani permintaan user per detiknya. Semakin besar semakin baik

2.2.9 *Tolerable Waiting Time*

Seseorang yang mengakses halaman web seringkali menghadapi waktu yang cukup lama dalam membuka sebuah halaman. Meskipun sebuah sistem seringkali terus diperbaiki, dan teknologi terus berubah ke arah lebih baik, namun sedikit sekali riset yang meneliti berapa lama seorang user mau menunggu untuk mengakses sebuah halaman web. Oleh karena itu, Nah, F. (2004) Dalam penelitiannya yang berjudul *A study on tolerable waiting time: How long are web users willing to wait?* melakukan penelitian untuk mengetahui berapa detik waktu respon sebuah halaman web yang dapat diterima oleh pengguna.

Dalam penelitian tersebut, Nah. F (2004) menyebutkan bahwa seorang pengguna mengharapkan sebuah halaman web yang berisi informasi tanpa adanya gambar dapat di buka dalam waktu 2 – 5 detik. Jika waktu respon diatas 10 detik, Ia menyarankan agar pengguna dapat melihat *progress* dar i *loading* halaman.