BAB II LANDASAN TEORI

2.1 Virtualization

Menurut Amit Singh [SIN04], definisi lepas Virtualization adalah: "virtualization is a framework or methodology of dividing the resources of a computer into multiple execution environments, by applying one or more concepts or technologies such as hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, quality of service, and many others". Kira-kira artinya adalah: virtualization adalah sebuah kerangka kerja atau metodologi membagi sumber daya dalam satu komputer ke dalam berbagai lingkungan eksekusi, dengan cara menerapkan satu atau lebih konsep atau teknologi seperti pembuatan partisi secara hardware dan software, pembagian-waktu, simulasi mesin sebagian atau keseluruhan, emulasi, kualitas layanan, dan lainnya. Sementara menurut IBM [IBM07], virtualization adalah teknik untuk menyembunyikan karakteristik fisik sumber daya komputasi dari cara yang dipakai oleh sistem lain, aplikasi, atau user untuk berinteraksi dengan sumber daya itu. Virtualization memungkinkan pemakaian bersama sumber daya seperti sistem operasi, software, layanan TI, dalam suatu cara sedemikian rupa sehingga menyembunyikan rincian teknis dari user dan mengurangi biaya layanan per unit (mesin).

Sistem virtualization berada antara hardware dengan guest, sehingga dapat mengendalikan penggunaan CPU, memori, media penyimpanan oleh guest, bahkan memungkinkan untuk guest OS berpindah dari satu mesin fisik ke mesin lainnya. Beberapa alasan umum penggunaan virtualization adalah:

- **Konsolidasi Server**. Perusahaan dapat meringkas kebutuhan fisik server yang perlu dimiliki. Beberapa server nyata dapat diganti dengan masing-masing satu virtual server yang dijalankan di atas satu server nyata, terisolasi satu sama lain.

- ***Disaster recovery***. Virtual server dapat dijadikan sebagai sistem "siap pakai" untuk server operasional. Jika terjadi kerusakan pada server produksi, image backup dari virtual mesin ini bisa di-boot untuk menjadi virtual mesin yang melayani tugas server produksi.

- **Pelatihan dan Pengujian**. Dengan virtualization, sebuah perusahaan atau lembaga pendidikan dapat memberikan hak akses administrator di dalam virtual mesin kepada peserta latihan atau user lain. User bisa melakukan banyak hal di dalam virtual mesin, mulai dari boot, restart, *crash*, merusak layanan, uji coba konfigurasi, dan lain-lain tanpa mengganggu aplikasi lain yang berjalan pada host.

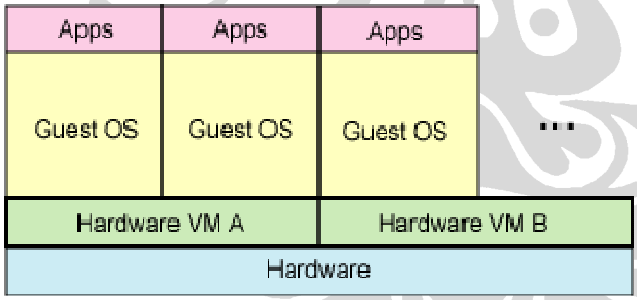
2.1.1 Tipe-tipe Virtualization

Ada beberapa tipe Virtualization menurut Tim Jones [TIM06], yaitu Hardware Emulation, Full Virtualization, Paravirtualization, dan Operating System-Level Virtualization. Masing-masing virtualization memiliki keuntungan dan kerugian sendiri-sendiri, yang terkait dengan masalah kompatibilitas sistem operasi dan kinerja mesin guest yang dijalankan di atas host OS. Kompatibilitas dan kinerja ini biasanya berbanding terbalik. Artinya, semakin baik kompatibilitas suatu teknik virtualization, semakin jelek kinerjanya (semakin lambat).

2.1.1.1 Hardware emulation

Hardware emulation menggunakan metode membuat VM hardware pada sistem host untuk mengemulasi hardware yang diinginkan, seperti pada Gambar 2. Masalah utama hardware emulation adalah hasilnya bisa jadi sangat lambat. Karena setiap instruksi harus disimulasikan pada hardware di bawahnya, maka perlambatan 100 kali menjadi biasa. Untuk emulasi *high-fidelity* yang menyertakan akurasi siklus, simulasi pipeline CPU, dan perilaku *caching*, perbedaan kecepatan yang sebenarnya bisa menjadi 1000 kali lebih lambat.

Keuntungan hardware emulation adalah, dengan menggunakan hardware emulation, orang bisa menggunakan sistem operasi tanpa modifikasi yang dibuat khusus untuk host PowerPC® pada prosesor ARM. Orang bisa menggunakan banyak VM, masing-masing menggunakan prosesor yang berbeda.

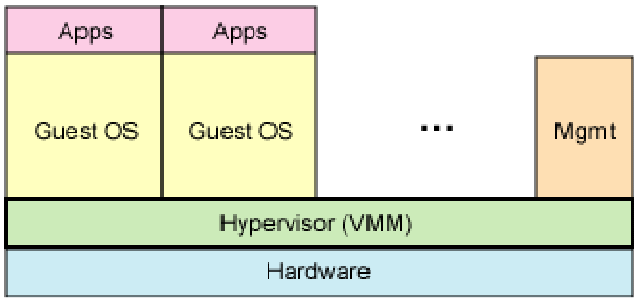


Gambar 2 Hardware emulation menggunakan sebuah VM untuk mensimulasikan hardware yang dibutuhkan [TIM06]

2.1.1.2 Full virtualization

Full virtualization, atau yang juga dikenal sebagai native virtualization, menggunakan sebuah virtual machine yang menghubungkan sistem operasi tamu dengan hardware native, seperti pada Gambar 3. Beberapa instruksi terproteksi harus

ditangkap dan ditangani dalam hypervisor karena hardware di bawahnya tidak dimiliki oleh sistem operasi tamu, melainkan dipakai bersama melalui hypervisor.

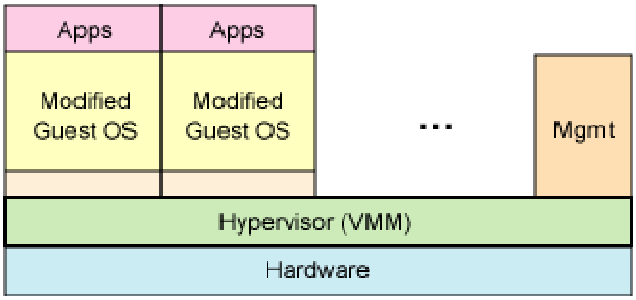


Gambar 3 Full virtualization menggunakan hypervisor untuk berbagi penggunaan hardware di bawahnya [TIM06].

Full virtualization lebih cepat daripada hardware emulation, tapi kinerjanya masih kalah bila dibandingkan dengan akses langsung ke hardware karena adanya mediasi hypervisor. Keuntungan terbesar dari full virtualization adalah sebuah sistem operasi dapat dijalankan tanpa modifikasi. Satu-satunya batasan adalah si sistem operasi harus mendukung hardware yang dipakai (misalnya, PowerPC).

2.1.1.3 Paravirtualization

Paravirtualization mirip dengan full virtualization. Metode ini menggunakan sebuah hypervisor untuk akses bersama kepada hardware di bawahnya tapi mengintegrasikan kode yang siap terhadap virtualization ke dalam sistem operasi, seperti pada Gambar 4. Pendekatan ini jelas membutuhkan kompilasi ulang atau trapping karena masing-masing sistem operasi bekerja sama dalam proses virtualization.

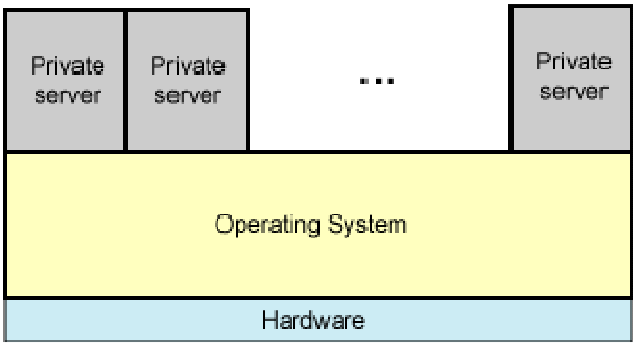


Gambar 4 Paravirtualization berbagi prosesor dengan sistem operasi tamu [TIM06]

Paravirtualization membutuhkan sistem operasi tamu dimodifikasi untuk hypervisor, yang merupakan kerugian. Tapi paravirtualization menawarkan kinerja mendekati sistem yang sebenarnya. Seperti full virtualization, berbagai macam sistem operasi bisa berjalan bersamaan.

2.1.1.4 Operating system-level virtualization

Operating system-level virtualization menggunakan teknik berbeda. Teknik ini memvirtualisasikan server di atas sistem operasi itu sendiri. Metode ini mendukung sebuah sistem operasi dan mengisolasi server-server independen satu sama lain, seperti pada Gambar 5.



Gambar 5 Operating system-level virtualization mengisolasi server-server [TIM06]

Operating system-level virtualization membutuhkan perubahan pada kernel sistem operasi, tapi keuntungannya adalah kinerja native.

2.1.2 Keuntungan Virtualization

Beberapa keuntungan Virtualization secara umum [SIN04], menurut Amit Singh adalah:

- Mesin virtual dapat digunakan untuk konsolidasi beban kerja server-server yang bebannya sangat sedikit ke dalam jumlah mesin yang lebih sedikit, bahkan mungkin ke dalam satu mesin saja. Keuntungan lain yang didapat secara otomatis adalah penghematan biaya hardware, biaya lingkungan, manajemen, dan administrasi infrastruktur server.
- Kebutuhan menjalankan aplikasi yang sudah tua bisa dilayani dengan baik oleh mesin virtual. Aplikasi lama peninggalan operasional sebelumnya bisa saja tidak dapat dijalankan pada hardware dan/atau sistem operasi baru. Bahkan jika bisa, si aplikasi bisa saja membuat server terbebani sangat sedikit, karena itu masuk akal untuk mengkonsolidasikan beberapa aplikasi sekaligus. Hal ini bisa menjadi sulit tanpa virtualisasi karena aplikasi yang dimaksud biasanya tidak dibuat untk berjalan berdampingan dalam lingkungan eksekusi tunggal.
- Mesin virtual bisa digunakan untuk menyediakan host yang terisolasi dan aman untuk menjalankan aplikasi yang belum terpercaya. Orang bahkan bisa membuat sebuah lingkungan eksekusi secara dinamik saat

dia mengunduh sesuatu dari Internet dan menjalankannya. Virtualization merupakan sebuah konsep penting dalam membangun platform komputasi yang aman.

- Mesin virtual dapat digunakan untuk membuat sistem operasi, atau lingkungan eksekusi dengan pembatasan sumber daya, dan jaminan sumber daya jika disandingkan dengan *scheduler* yang benar. *Partitioning* biasanya berjalan bersamaan dengan *quality of service* dalam pembuatan sistem operasi yang *QoS-enabled*.

- Mesin virtual bisa memberikan gambaran hardware, atau konfigurasi hardware yang tidak dimiliki (seperti alat-alat SCSI, multiprosesor, dll). Virtualization juga bisa digunakan untuk mensimulasikan jaringan sekumpulan komputer yang independen.

- Mesin virtual dapat digunakan untuk menjalankan beberapa sistem operasi sekaligus: versi yng berbeda, atau sistem yang sama sekali berbeda. Beberapa sistem bisa saja sulit atau bahkan tidak mungkin dijalankan pada hardware yang dipakai.

- Mesin virtual bisa digunakan untuk *debugging* dan pemantauan kinerja. Contohnya orang bisa menggunakan alat pantau dalam monitor virtual mesin. Sistem operasi bisa di-debug tanpa kehilangan produktivitas, atau membuat berbagai skenario *debugging* yang rumit.

- Mesin virtual dapat mengisolasi apa yang dia jalankan, agar mereka menyediakan penangkaran *fault* dan *error*. Orang dapat memasukan bermacam *fault* ke dalam software untuk mempelajari akibat yang ditimbulkan.

- Mesin virtual membuat migrasi software lebih mudah, sehingga memanti mobilitas aplikasi dan sistem.
- Orang dapat memperlakukan kumpulan aplikasi sebagai *appliance* dan memaketkan serta menjalankan masing-masing *appliance* ke dalam sebuah mesin virtual.
- Mesin virtual adalah alat bantu yang sangat bagus untuk percobaan riset dan akademik. Karena Virtualization menyediakan isolasi, bekerja dengan mesin virtual menjadi aman. Mesin virtual membungkus seluruh *state* sistem yang sedang berjalan: orang bisa menyimpan *state*, mempelajarinya, memodifikasinya, memanggilnya kembali, dll. *State* ini juga menyediakan abstraksi beban kerja yang sedang dijalankan.
- Virtualization dapat memungkinkan sebuah sistem operasi yang ada untuk berjalan pada multi prosesor dengan *shared memory*.
- Mesin virtual dapat digunakan untuk membuat berbagai skenario uji coba acak, dan bisa mengarahkan ke *quality assurance* yang efektif dan sangat imajinatif.
- Virtualization bisa digunakan untuk melengkapi fitur-fitur baru dalam sistem operasi yang ada tanpa "terlalu banyak" usaha.
- Virtualization dapat membuat tugas-tugas seperti migrasi, *backup*, dan *recovery* menjadi lebih mudah dan lebih dapat dikelola.
- Virtualization bisa menjadi alat yang efektif untuk menyediakan kompatibilitas biner.

- Virtualization pada komoditas hardware telah menjadi populer dalam *co-located hosting*. Keuntungan-keuntungan yang sudah disebutkan di atas membuat *hosting* menjadi aman, tepat-biaya, dan secara umum menjadi lebih menarik.

Sementara itu Info-Tech [INF06] mendefinisikan keberhasilan implementasi Virtualization ditentukan jika suatu perusahaan berhasil mendapatkan tiga jenis keuntungan:

- **Tangible benefits**: pencapaian tujuan keuangan bisnis yang sudah ditentukan, termasuk penghematan biaya pada satu waktu dan satu periode tertentu dan penyelesaian implementasi.
- **Intangible benefits**: mengenali keuntungan fitur yang didapat dari implementasi virtualization dan menunjukkan peningkatan efisiensi. Termasuk juga terjaganya kelangsungan bisnis (server downtime lebih sedikit) dan disaster recovery (sistem backup yang disediakan virtualization). Selain itu juga sistem yang lebih mudah dikelola, pemisahan antara layar hardware dan software, dan kendali atas penambahan server.
- **Strategic benefits**: Mengenali nilai virtualization dan memiliki rencana untuk menggunakan teknologi ini demi perkembangan perusahaan di masa depan. Poin penting di sini adalah, perusahaan bisa menggunakan Virtualization untuk menolong mereka menjalankan infrastruktur internal sebagai layanan alih-alih sebagai tumpukan hardware.

Yang menarik dari berbagai keuntungan yang telah disebutkan adalah keuntungan dari penghematan biaya kepemilikan, atau penghematan Total Cost of Ownership (TCO). Menurut laporan analisis perusahaan Hewlett-Packard [HEW05], sebuah solusi yang mengimplementasikan sebuah sistem BEA dan Oracle yang dijalankan pada HP Integrity dan HP-UX 11i dengan Virtualization pada suatu periode selama tiga tahun bisa menghemat biaya sebesar USD 4,955,092 jika dibandingkan dengan solusi yang sama tanpa Virtualization. Perbedaan paling besar di antara kedua solusi itu adalah biaya kepemilikan server hardware dan server software, dengan nilai perbedaan masing-masing adalah USD 1,455,850 untuk server hardware dan USD 3,094,648 untuk server software. Ini adalah contoh nyata seberapa besar Virtualization dapat memberikan penghematan kepada suatu organisasi.

2.1.3 *Pitfalls*

Menurut James Magure [MAG06], ada beberapa *pitfall* yang harus diawasi dalam menerapkan virtualization. Beberapa yang paling penting adalah:

- **Ekspektasi pengurangan karyawan**. Jangan terlalu berharap bisa mengurangi pegawai dari penerapan virtualization. Walau virtualization bisa mengurangi beban kerja staf TI, mereka tetap masih punya banyak pekerjaan lain.
- **Ekspektasi konsolidasi besar-besaran**. Jangan terlalu berharap bahwa akan ada pengurangan server fisik dengan jumlah ekstrim. Berdasarkan klaim vendor virtualization, sistem yang mereka tawarkan bisa menggantikan 10 sampai 12 server menjadi satu server nyata. Kondisi yang realistik, dengan adanya tuntutan SLA dan lain-lain,

menunjukkan bahwa rata-rata konsolidasi adalah enam server menjadi satu.

- **I/O Bottleneck**. Ada kemungkinan bahwa server host akan mengalami bottleneck dalam kapasitas I/O. Virtualization dapat menjalankan satu fisik server dengan beberapa OS sekaligus beserta aplikasi-aplikasi lain. Perlu diperhatikan bahwa jika terlalu banyak data dengan cepat disalurkan melaluinya, server bisa mengalami bottleneck I/O.
- **Masalah lisensi**. Beberapa vendor software mengenakan biaya lisensi tergantung pada jumlah prosesor pada server. Contohnya Oracle. Jika suatu server dengan empat prosesor akan menjalankan suatu virtual server dengan satu virtual prosesor dan Oracle, Oracle bisa mengenakan biaya atas lisensi dengan empat prosesor walaupun yang digunakan hanya satu. Intinya, jika tidak hati-hati dalam implementasi virtualiation, biaya lisensi malah meningkat.

2.2 Software dan Hardware Virtualization

Paper karya Popek dan Goldber pada tahun 1974 [POP74] mendefinisikan tiga karakteristik utama untuk sistem software agar bisa dianggap sebagai Virtual Machine Monitor (VMM):

1. Fidelity. Software pada VMM mengeksekusi secara identik dengan eksekusi pada hardware, membawa akibat pada efek timing.
2. Performance. Sekumpulan besar instruksi guest dieksekusi oleh hardware tanpa intervensi VMM.

3. Safety. VMM mengelola seluruh sumber daya hardware.

Pada saat itu, gaya implementasi VMM spesifik adalah trap-and-emulate, dan sangat populer sehingga dianggap satu-satunya metode praktikal untuk virtualization. Walau Popek dan Goldberg tidak mengeliminasi penggunaan teknik lain, selama bertahun-tahun orang mencoba menyamakan konsep "virtualizability" dengan kemampuan menggunakan trap-and-emulate. Arsitektur yang bisa di-virtualisasikan murni dengan trap-and-emulate disebut dengan "classical virtualization" [ADA06]. "Classical virtualization" sendiri memiliki beberapa masalah. Arsitektur x86 tidak termasuk ke dalam "classical virtualization", tapi bisa melakukan virtualization melalui software virtualization atau/dan hardware virtualization.

Software virtualization adalah implementasi virtualization yang menghubungkan hardware, guest OS, dan manusia secara software. Sedangkan hardware virtualization sebenarnya mengacu kepada "hardware-assisted virtualization system", di mana sebagian tugas Virtual Machine Monitor (VMM) dikerjakan oleh hardware yang memiliki instruction-set khusus untuk menangani tugas-tugas ini [ADA06]. Jadi keberadaan hardware virtualization tidak mutlak harus ada dalam implementasi virtualization, sementara software virtualization adalah komponen utama dalam virtualization.

2.3 Uji Kinerja

Meier dkk. menulis panduan yang bagus [MEI07] untuk melakukan berbagai macam uji kinerja. Dalam panduan itu disebutkan bahwa uji kinerja adalah sebuah pengujian yang bertujuan untuk menentukan *responsiveness*, *throughput*, *reliability*,

dan/atau *scalability* dari sebuah sistem terhadap sebuah beban kerja yang diberikan. Biasanya uji kinerja dilakukan untuk menjalankan:

- Pemeriksaan kesiapan produksi.
- Evaluasi terhadap kriteria kinerja.
- Membandingkan karakteristik kinerja dari berbagai sistem atau konfigurasi sistem.
- Mencari sumber permasalahan kinerja.
- Mendukung *tuning* sistem.
- Mencari tingkatan *throughput*.

Untuk melakukan uji kinerja, aktivitas-aktivitas utama yang dilakukan adalah:

1. Identifikasi lingkungan ujian.
2. Identifikasi kriteria penerimaan kinerja.
3. Merencanakan dan merancang ujian.
4. Konfigurasi lingkungan ujian.
5. Implementasi rancangan ujian.
6. Eksekusi ujian.
7. Analisis, laporkan, dan uji ulang.

Uji kinerja sendiri biasanya termasuk ke dalam salah satu dari tiga kategori ini:

- **Uji kinerja**. Jenis ujian ini menentukan atau memvalidasi kecepatan, skalabilitas, dan/atau stabilitas karakteristik dari sistem atau aplikasi yang diuji. Kinerja diukur dengan pencapaian waktu respon,

throughput, dan tingkatan utilisasi sumber daya yang memenuhi tujuan kinerja produk atau proyek. Uji kinerja merupakan superset dari seluruh subkategori ujian yang berhubungan dengan kinerja.

- **Uji beban**. Subkategori ini berfokus pada menentukan atau memvalidasi karakteristik kinerja sistem atau aplikasi yang diuji bila diujikan dengan berbagai beban kerja atau beban volume yang diantisipasi selama masa produksi.
- **Uji stress (stress testing)**. Subkategori ini berfokus pada menentukan atau memvalidasi karakteristik kinerja sistem atau aplikasi yang diuji pada kondisi di luar yang diantisipasi pada periode produksi. Uji stress bisa menyertakan ujian terhadap subyek yang sama dengan kondisi *stressful*, seperti memory terbatas, ruang disk yang tidak cukup, atau *server failure*.

Dari panduan ini diambil langkah-langkah yang releven dengan aktivitas-aktivitas yang disebutkan di atas untuk melakukan uji kinerja sesuai dengan tujuan penelitian. Tujuan penelitian ini sendiri sesuai dengan butir " Membandingkan karakteristik kinerja dari berbagai sistem atau konfigurasi sistem". Uji kinerja yang dilakukan termasuk ke dalam subkategori uji beban, untuk mengetahui kinerja virtualization pada berbagai kondisi beban.

2.4 Layanan Jaringan Yang Dipilih

Dalam penelitian ini, beberapa layanan dipilih sebagai representasi layanan yang umum digunakan dalam dunia TI untuk diujikan dalam uji kinerja. Layanan-

layanan yang dipilih adalah layanan HTTP (Hypertext Transfer Protocol), SMTP (Simple Mail Transfer Protocol), dan CIFS (Common Internet File System) yang merupakan layanan file-sharing yang dikembangkan Microsoft dan populer seiring dengan kepopuleran Microsoft Windows.

2.4.1 HTTP

HTTP adalah protokol komunikasi untuk bertukar informasi di intranet dan Internet. HTTP merupakan protokol yang paling populer di Internet, bahkan pernah mengisi 75% trafik backbone Internet [THO97]. Dengan kepopulerannya, banyak vendor software membuat aplikasi yang memanfaatkan protokol ini.

HTTP dikembangkan di bawah koordinasi W3C (Word Wide Web Consortium) yang bekerja sama dengan IETF (Internet Engineering Task Force). Hasil pengembangan ini adalah publikasi sekumpulan Request For Comments (RFC), terutama RFC 2616 yang mendefinisikan HTTP/1.1 versi yang paling banyak digunakan saat ini.

Implementasi HTTP adalah berupa protokol klien-server. Klien adalah end-user, dan server adalah situs web. Klien mengirimkan sebuah permintaan HTTP kepada server, biasanya menggunakan web browser. Server yang bersangkutan menerima permintaan itu dan memberikan jawaban yang sesuai dengan konfigurasinya, apakah itu hanya status permintaan atau informasi yang diminta klien.

Dari segi utilisasi CPU, HTTP akan memakan sumber daya CPU seiring dengan bertambahnya jumlah HTTP-request yang harus dilayani. Setiap HTTP-request yang datang akan diproses server HTTP terhadap sekumpulan peraturan yang

sudah didefinisikan sebelumnya, dan akan dibalas dengan sebuh HTTP-reply berdasarkan aturan tadi. Utilisasi CPU ini akan bertambah jika server HTTP dikombinasikan dengan pemrosesan output secara dinamis, seperti HTML preprocessor, komputasi database, otentikasi user, dll. Sementara dari segi Input Output, layanan HTTP akan memakan sumber daya I/O seiring dengan besarnya file yang harus dikirim sebagai jawaban atas HTTP-request. Besarnya file ini juga akan mempengaruhi bandwidth jaringan yang terpakai sehingga jumlah HTTP-request dan HTTP-reply akan terbentur pada kapasitas bandwidth.

Dalam penelitian ini, fokus ditekankan kepada pemantauan kinerja terhadap utilisasi CPU yang terpakai oleh murni proses server HTTP, karena itu server HTTP di sini akan menjawab HTTP-request dengan balasan statis. Mengingat bandwidth yang dipakai dalam eksperimen ini terbatas (100Mbps), maka besar file yang dipakai untuk proses HTTP-reply dibatasi agar kombinasinya dengan HTTP-request tidak terbentur kapasitas bandwidth jaringan.

2.4.2 SMTP

SMTP merupakan standar de facto untuk transmisi e-mail melalui Internet. Definisi SMTP secara formal terdokumentasikan dalam RFC 821, dan diamandemen oleh RFC 1123 bab 5. Protokol SMTP yang digunakan saat ini juga dikenal sebagai ESMTP dan didefinisikan dalam RFC 2821.

SMTP adalah protokol berbasis teks yang relatif sederhana, yang didalamnya disebutkan satu atau lebih penerima pesan, bersama dengan teks pesan dan obyek lain dalam bentuk terurai (*encoded*). Pesan ini kemudian dipindahkan ke server lain menggunakan sebuah prosedur permintaan dan jawaban antara klien dan server. Yang

bertindak sebagai SMTP klien bisa jadi sebuah klien e-mail milik user, alias MUA (Mail User Agent), atau sebuah server MTA (Mail Transport Agent) yang me-relay e-mail. MTA juga bisa bertindak sebagai SMTP server, bila fungsinya sebagai penerima relay dari MTA lain. SMTP klien menjembatani hubungan antara user dengan SMTP server. Sementara SMTP server berfungsi sebagai penerima pesan yang dikirim oleh SMTP klien, dan mendistribusikannya kepada penerima yang tercantum di dalam pesan.

Dari segi utilisasi CPU, SMTP akan memakan sumber daya CPU seiring dengan bertambahnya jumlah SMTP-request yang harus dilayani. Setiap SMTP-request terdiri dari dua bagian, yaitu header dan isi. Header berisi informasi-informasi tentang alamat asal pengirim, alamat tujuan penerima, dan informasi lain yang berhubungan dengan pesan. Isi berisi teks yang merupakan komunikasi antara pengirim pesan dengan penerima pesan. Selanjutnya seluruh pesan yang masuk akan diproses ke alamat penerima yang dimaksud. Untuk pesan-pesan yang sudah diterima tapi belum sempat diproses, server SMTP akan menampungnya ke dalam antrian, menunggu untuk diproses. Sementara dari segi Input Output, layanan SMTP akan memakan sumber daya I/O seiring dengan besarnya total pesan yang diterima. Besarnya file ini juga akan mempengaruhi bandwidth jaringan yang terpakai sehingga jumlah SMTP-request dan SMTP-reply akan terbentur pada kapasitas bandwidth.

Dalam penelitian ini, fokus ditekankan kepada pemantauan kinerja terhadap utilisasi CPU yang terpakai oleh murni proses server SMTP. Mengingat bandwidth yang dipakai dalam eksperimen ini terbatas (100Mbps), maka besar data yang dipakai pada setiap transaksi SMTP dibatasi agar total transaksi SMTP dalam satu waktu tidak terbentur kapasitas bandwidth jaringan.

2.4.3 CIFS

CIFS merupakan teknologi yang dibuat untuk menggantikan SMB (Server Message Block). SMB adalah sebuah protokol yang aslinya dibuat oleh IBM, Microsoft, Intel, dan 3Com untuk MS-DOS dan PC-DOS [HER03] pada era 1980-an. SMB kemudian dikembangkan untuk memungkinkan user melihat layanan file dan printer yang tersedia di jaringan melalui layanan Browse dari sistem operasi Microsoft Windows. Seiring dengan kepopuleran sistem operasi Microsoft Windows, protokol SMB juga mendapatkan kepopulerannya sehingga memicu kemunculan implementasi SMB agar dapat diakses dari sistem operasi selain Microsoft Windows.

Pada tahun 1996 Microsoft menyerahkan draft spesifikasi CIFS kepada IETF. Walau draft-nya saat ini sudah kadaluarsa, tidak banyak dokumen lain yang memuat spesifikasi rinci tentang CIFS ini. Karena itu spesifikasi implementasi yang diakui de facto adalah implementasi yang tersedia built-in pada keluarga sistem operasi Microsoft Windows. Implementasi dari vendor lain merupakan hasil studi dari implementasi Microsoft Windows, kebanyakan melibatkan proses reverse-engineering.

Dari segi utilisasi CPU, server CIFS akan memakan sumber daya CPU seiring dengan bertambahnya jumlah transaksi file yang harus dilayani. Semakin banyak klien yang mengakses dan memanfaatkan server CIFS, maka semakin banyak pula transaksi file yang terjadi. Setiap transaksi file, baik itu baca, tulis, atau hapus akan memakan utilisasi I/O dan otomatis utilisasi CPU.

Dalam dunia nyata, ukuran file yang ditransfer melalui layanan CIFS hampir tidak ada yang hanya berukuran di bawah 1KB. Sebuah file bertipe Microsoft Word 2003 Document yang masih kosong saja mempunyai ukuran sebesar 11KB. Jika orang mentransfer file lagu dengan format MP3, kemungkinan besar file itu berukuran

lebih dari 1MB, dan aktivitas ini sering terjadi di dunia nyata. Karena itu, besar file yang terlibat dalam pengujian layanan CIFS ini sangat bervariasi.

Dalam penelitian ini, fokus ditekankan kepada pemantauan kinerja terhadap utilisasi CPU yang terpakai oleh murni proses server CIFS. Mengingat bandwidth yang dipakai dalam eksperimen ini terbatas (100Mbps), sedangkan ukuran file sangat bervariasi, maka parameter lain yang dijadikan ukuran adalah throughput data, yaitu besarnya data yang dapat ditransfer dalam satu waktu.