# Bab 8. *Virtual Machine*(VM)

## 8.1. Pendahuluan

*Virtual Machine*(VM) adalah sebuah mesin yang mempunyai dasar logika yang menggunakan pendekatan lapisan-lapisan ( *layers*)dari sistem komputer.Sehingga sistem komputer dengan tersendiri dibangun atas lapisan-lapisan tersebut, dengan urutan lapisannya mulai dari lapisan terendah sampai lapisan teratas adalah sebagai berikut:

- Perangkat keras (semua bagian fisik komputer)
- Kernel (program untuk mengontrol disk dan sistem file, *multi-tasking*, *load-balancing*, *networking* dan *security*)
- Sistem program (program yang membantu *general user*)

Kernel yang berada pada lapisan kedua ini, menggunakan instruksi perangkat keras untuk menciptakan seperangkat *system call* yang dapat digunakan oleh komponen-komponen pada level sistem program. Sistem program kemudian dapat menggunakan *system call* dan perangkat keras lainnya seolah-olah pada level yang sama. Meskipun sistem program berada di level tertinggi , namun program aplikasi bisa melihat segala sesuatu pada tingkatan dibawahnya seakan-akan mereka adalah bagian dari mesin. Pendekatan dengan lapisan-lapisan inilah yang kemudian menjadi kesimpulan logis pada konsep *Virtual Machine*(VM) atau *virtual machine*(VM).

## Kekurangan *Virtual Machine*(VM)

Ada beberapa kesulitan utama dari konsep VM, diantaranya adalah:

- **Dalam sistem penyimpanan.** Sebagai contoh kesulitan dalam sistem penyimpanan adalah sebagai berikut: Andaikan kita mempunyai suatu mesin yang memiliki 3 disk drive namun ingin mendukung 7 VM. Keadaan ini jelas tidak memungkinkan bagi kita untuk dapat mengalokasikan setiap disk drive untuk tiap VM, karena perangkat lunak untuk mesin virtual sendiri akan membutuhkan ruang disk secara substansi untuk menyediakan memori virtual dan *spooling*.Solusinya adalah dengan menyediakan disk virtual atau yang dikenal pula dengan *minidisk*, dimana ukuran daya penyimpanannya identik dengan ukuran sebenarnya. Dengan demikian, pendekatan VM juga menyediakan sebuah antarmuka yang identik dengan *underlying bare hardware*.
- **Dalam hal pengimplementasian.** Meski konsep VM cukup baik, namun VM sulit diimplementasikan.

## Kelebihan *Virtual Machine*(VM)

Terlepas dari segala kekurangannya, VM memiliki beberapa keunggulan, antara lain:

- **Dalam hal keamanan.** VM memiliki perlindungan yang lengkap pada berbagai sistem sumber daya, yaitu dengan meniadakan pembagian *resources* secara langsung, sehingga tidak ada masalah proteksi dalam VM. Sistem VM adalah kendaraan yang sempurna untuk penelitian dan pengembangan sistem operasi. Dengan VM, jika terdapat suatu perubahan pada satu bagian dari mesin, maka dijamin tidak akan mengubah komponen lainnya.
- **Memungkinkan untuk mendefinisikan suatu jaringan dari *Virtual Machine*(VM).** Tiap-tiap bagian mengirim informasi melalui jaringan komunikasi virtual. Sekali lagi, jaringan dimodelkan setelah komunikasi fisik jaringan diimplementasikan pada perangkat lunak.

## 8.2. Virtualisasi Penuh

Virtualisasi penuh dalam ilmu komputer ialah teknik virtualisasi yang digunakan untuk implementasi pada berbagai macam lingkungan *virtual machine*: Salah satunya menyediakan simulasi lengkap

yang mendasari suatu hardware. Hasilnya adalah sebuah system yang mampu mengeksekusi semua perangkat lunak pada perangkat keras yang bias dijalankan pada *Virtual Machine*(VM), termasuk semua sistem operasi.

Setiap user CP/CMS telah disediakan sebuah simulasi, komputer yang berdiri sendiri (stand-alone computer). Setiap mesin virtual serupa telah mempunyai kemampuan lengkap mesin yang mendasar, dan untuk *user Virtual Machine (VM)*telah tak dapat dibedakan dengan sistem privasi. Simulasi ini sangat luas, dan didasarkan pada prinsip operasi manual untuk perangkat keras. Jadi termasuk setiap elemen sebagai set instruksi, main memory, intrupsi, exceptions, dan akses peralatan. Hasilnya ialah sebuah mesin tunggal yang dapat menjadi multiplexed diantara banyak user. Virtualisasi penuh hanya mungkin diberikan pada kombinasi yang benar dari elemen hardware dan software. Sebagai contoh, tidak mungkin dengan kebanyakan system IBM pada seri 360, hanya dengan IBM sistem 360-67.

Tantangan utama pada virtualisasi penuh ada pada intersepsi dan simulasi dari operasi yang memiliki hak istimewa seperti instruksi I/O. Efek dari setiap operasi yang terbentuk dengan penggunaan *Virtual Machine*(VM) haruslah dirawat dalam *Virtual Machine*(VM) itu operasi virtual tidak diijinkan untuk diubah *state* dari virtual mesin lainnya, control program atau hardware. Beberapa instruksi mesin dapat di eksekusi secara langsung oleh hardware, semenjak itu efek sepenuhnya terkandung di dalam elemen yang dimanage oleh program kontrol, seperti lokasi memori dan register aritmatik. Tetapi instruksi lain yang dikenal dapat menembus mesin virtual tidak diijinkan untuk langsung di eksekusi, haruslah sebagai gantinya dikurung dan disimulasi.

Beberapa instruksi baik akses atau pengaruh *state* informasi berada di luar *Virtual Machine*(VM). Virtualisasi penuh telah terbukti sukses untuk sharing sistem diantara banyak user dan mengisolasi user dari user yang lainnya untuk reabilitas (kepercayaan) dan keamanan.

# 8.3. Virtualisasi Paruh

Virtualisasi paruh dalam ilmu komputer ialah teknik virtualisasi yang digunakan untuk pengimplementasian pada berbagai macam lingkungan *virtual machine*, salah satunya dengan menyediakan sebagian besar hal yang mendasari suatu *hardware*. Sebenarnya tidak semua fitur yang dimiliki *hardware* tersebut tersimulasi, tapi ada beberapa kemudahan *Virtual Machine*(VM) yang mana tidak semua *software* dapat berjalan tanpa modifikasi. Biasanya untuk mengartikan bahwa seluruh sistem operasi tidak dapat berjalan pada *Virtual Machine*(VM) akan mengisyaratkan virtualisasi penuh, namun banyak aplikasi dapat berjalan pada mesin tersebut.

Virtualisasi paruh memiliki pertanda lebih mudah diterapkan daripada virtualisasi penuh. Seiring dengan syarat kegunaan, *Virtual Machine*(VM) yang kuat mampu untuk mendukung aplikasi penting. Kekurangan virtualisasi paruh dibandingkan dengan virtualisasi penuh adalah keterbelakangan kesesuaian ( *compatibility*) atau mudah dibawa ( *portability*). Jika tampilan hardware tertentu tidak disimulasikan, suatu software yang menggunakan tampilan itu akan fail. Terlebih lagi, akan sulit untuk mengantisipasi nilai suatu tampilan yang telah akan digunakan oleh pemberian aplikasi. Virtualisasi paruh telah terbukti secara sukses untuk pertukaran sumberdaya antar banyak pengguna.

# 8.4. IBM VM

Istilah *Virtual Machine*(VM) sendiri mulai dikenalkan oleh IBM ketika meluncurkan sistem operasi mainframenya pada tahun 1965-an. Diperkenalkan untuk sistem S/370 dan S/390 dan disebut sebagai sistem operasi VM/ESA ( *Enterprise System Architecture*). Sehingga sering menimbulkan kebingungan antara penamaan produk atau penamaan mekanisme. Banyak orang yang menyebut, walau memiliki mekanisme *Virtual Machine*(VM) tetapi bila bukan dari sistem IBM tersebut, maka tidak disebut dengan *Virtual Machine*.

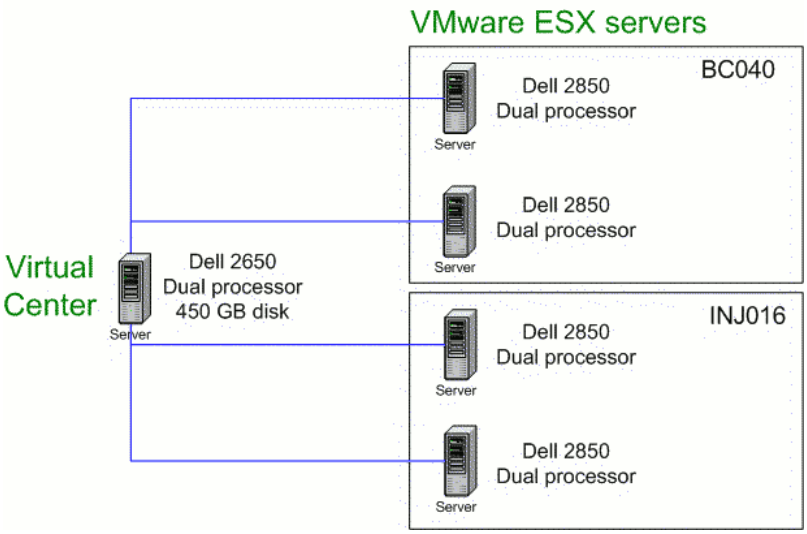
Pada penjelasan ini, istilah *Virtual Machine*(VM) adalah suatu jenis mekanisme virtualisasi suatu mesin di atas mesin lainnya. Jadi bukan jenis produk dari salah satu vendor dengan nama *Virtual*

*Machine*. Terdapat beberapa kegunaan dari *Virtual Machine*(VM), pada umumnya tampak untuk menggambarkan program yang bertindak selayaknya mesin.

# 8.5. VMware

Pada GNU/Linux salah satu *virtual machine* yang terkenal adalah VMware http://www.vmware.com. VMware memungkinkan beberapa sistem operasi dijalankan pada satu mesin PC tunggal secara bersamaan. Hal ini dapat dilakukan tanpa melakukan partisi ulang dan boot ulang. Pada *Virtual Machine*(VM) yang disediakan akan dijalankan sistem operasi sesuai dengan yang diinginkan. Dengan cara ini maka pengguna dapat memboot suatu sistem operasi (misal Linux) sebagai sistem operasi tuan rumah ( *host*) dan lalu menjalankan sistem operasi lainnya misal MS Windows. Sistem operasi yang dijalankan di dalam sistem operasi tuan rumah dikenal dengan istilah sistem operasi tamu ( *guest*).

**Gambar 8.1. Contoh skema penggunaan pada VMware versi ESX Servers**




Kebanyakan orang berpikir bahwa secara logisnya VMware diibaratkan sebagai *software* yang sering digunakan untuk keperluan percobaan *game*, aplikasi, untuk meng-install dua sistem operasi dan menjalankannya (misalnya Windows maupun Linux) pada *harddisk* yang sama tanpa memerlukan logout dari sistem operasi yang lainnya, secara gampang kita hanya tinggal menekan Alt + Tab untuk mengganti SO.

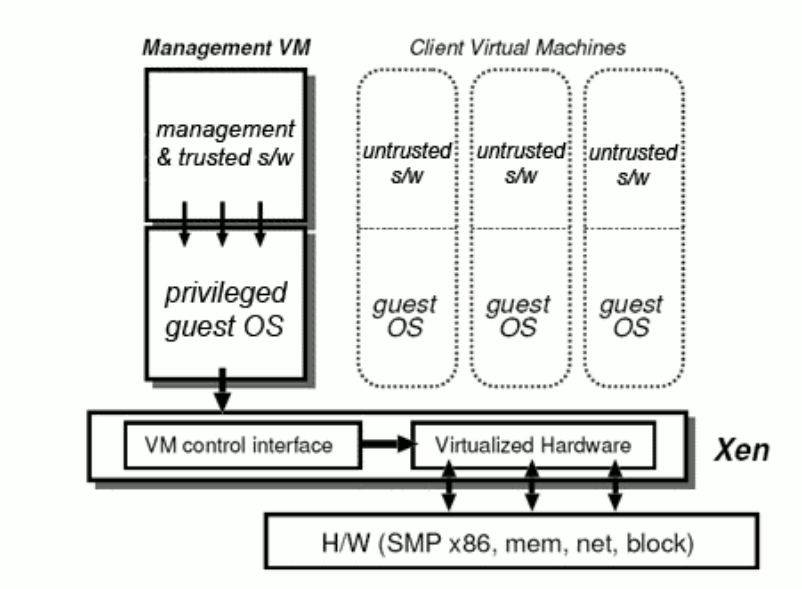
Akan tetapi pada dasarnya VMware bukanlah emulator, karena tidak mengemulasikan CPU dan perangkat keras di dalam suatu *Virtual Machine*(VM), tetapi hanya membolehkan sistem operasi lainnya dijalankan secara paralel dengan sistem operasi yang telah berjalan. Setiap *Virtual Machine*(VM) dapat memiliki alamat IP sendiri (jika mesin tersebut di suatu jaringan), dan pengguna dapat menganggapnya sebagai mesin terpisah.

# 8.6. Xen VMM

Xen adalah open source *virtual machine monitor*, dikembangkan di University of Cambridge. Dibuat dengan tujuan untuk menjalankan sampai dengan seratus sistem operasi ber-fitur penuh ( *full featured OSs*) di hanya satu komputer. Virtualisasi Xen menggunakan teknologi paravirtualisasi menyediakan isolasi yang aman, pengatur sumberdaya, garansi untuk *quality-of-services* dan *live migration* untuk sebuah mesin virtual.

Untuk menjalankan Xen, sistem operasi dasar harus dimodifikasi secara khusus untuk kebutuhan tersendiri dan dengan cara ini dicapai kinerja virtualisasi sangat tinggi tanpa hardware khusus.

**Gambar 8.2. Contoh dari penggunaan Xen VMM**




# 8.7. Java VM

Program Java yang telah dikompilasi adalah *platform-neutral bytecodes* yang dieksekusi oleh *Java Virtual Machine*(JVM). JVM sendiri terdiri dari: *class loader*, *class verification*, *runtime interpreter*, *Just In-Time*(JIT) untuk meningkatkan kinerja kompilator.

Bahasa mesin terdiri dari sekumpulan instruksi yang sangat sederhana dan dapat dijalankan secara langsung oleh CPU dari suatu komputer. Sebuah program yang dibuat dengan bahasa tingkat tinggi tidak dapat dijalankan secara langsung pada komputer. Untuk dapat dijalankan, program tersebut harus ditranslasikan kedalam bahasa mesin. Proses translasi dilakukan oleh sebuah program yang disebut *compiler*.

Setelah proses translasi selesai, program bahasa-mesin tersebut dapat dijalankan, tetapi hanya dapat dijalankan pada satu jenis komputer. Hal ini disebabkan oleh setiap jenis komputer memiliki bahasa mesin yang berbeda-beda. Alternatif lain untuk mengkompilasi program bahasa tingkat tinggi selain menggunakan compiler, yaitu menggunakan *interpreter*. Perbedaan antara *compiler* dan *interpreter* adalah *compiler* mentranslasi program secara keseluruhan sekaligus, sedangkan *interpreter* menstranslasi program secara instruksi per instruksi. Java dibuat dengan mengkombinasikan antara *compiler* dan *interpreter*.

Program yang ditulis dengan java di- *compile* menjadi bahasa mesin. Tetapi bahasa mesin untuk komputer tersebut tidak benar-benar ada. Oleh karena itu disebut *"Virtual"* komputer, yang dikenal dengan *Java Virtual Machine (JVM)*. Bahasa mesin untuk JVM disebut Java *bytecode*. Salah satu keunggulan dari Java adalah dapat digunakan atau dijalankan pada semua jenis komputer. Untuk menjalankan program Java, komputer membutuhkan sebuah *interpreter* untuk Java *bytecode*.

Interpreter berfungsi untuk mensimulasikan JVM sama seperti *virtual computer* mensimulasikan PC komputer. Java *bytecode* yang dihasilkan oleh setiap jenis komputer berbeda-beda, sehingga diperlukan interpreter yang berbeda pula untuk setiap jenis komputer. Tetapi program Java *bytecode* yang sama dapat dijalankan pada semua jenis komputer yang memiliki Java *bytecode*.

# 8.8. *.NET Framework*

*.NET Framework* merupakan suatu komponen Windows yang terintegrasi yang dibuat dengan tujuan pengembangan berbagai macam aplikasi serta menjalankan aplikasi generasi mendatang termasuk pengembangan aplikasi XML Web Services.

Keuntungan Menggunakan *.NET Framework*

1. **Mudah.** Yang dimaksud mudah di sini adalah kemudahan developer untuk membuat aplikasi yang dijalankan di .NET Framework. Mendukung lebih dari 20 bahasa pemrograman : *VB.NET, C#, J#, C++, Pascal, Phyton (IronPhyton), PHP (PhLager).*
2. **Efisien.** Kemudahan pada saat proses pembuatan aplikasi, akan berimplikasi terhadap efisiensi dari suatu proses produktivitas, baik efisien dalam hal waktu pembuatan aplikasi atau juga efisien dalam hal lain, seperti biaya (cost).
3. **Konsisten.** Kemudahan-kemudahan pada saat proses pembuatan aplikasi, juga bisa berimplikasi terhadap konsistensi pada aplikasi yang kita buat. Misalnya, dengan adanya *Base Class Library*, maka kita bisa menggunakan objek atau *Class* yang dibuat untuk aplikasi berbasis windows pada aplikasi berbasis web. Dengan adanya kode yang bisa dintegrasikan ke dalam berbagai macam aplikasi ini, maka konsistensi kode-kode aplikasi kita dapat terjaga.
4. **Produktivitas.** Semua kemudahan-kemudahan di atas, pada akhirnya akan membuat produktivitas menjadi lebih baik. Produktivitas naik, terutama produktivitas para *developer*, akan berdampak pada meningkatnya produktivitas suatu perusahaan atau project.

# 8.9. Rangkuman

Konsep *Virtual Machine*(VM) adalah dengan menggunakan pendekatan lapisan-lapisan (layers) dari sistem komputer. Adapun beberapa hal yang berhubungan dan termasuk dalam *vitual machine* antara lain vitualisasi penuh dan paruh, IBM VM, VMware, Xen VMM, Java VM dan *.NET Framework.*

Virtualisasi adalah metode untuk membuat sesuatu menjadi lepas dari ketergantungan secara fisik. Contoh; virtual machine adalah komputer, yang sebetulnya hanya berupa sebuah file di hard disk kita. Dengan virtualisasi, maka sebuah komputer (fisik) bisa menjalankan banyak komputer virtual sekaligus pada saat yang bersamaan. *Virtual Machine*(VM) sendiri mulai dikenalkan oleh IBM ketika meluncurkan sistem operasi mainframenya pada tahun 1965-an. Diperkenalkan untuk sistem S/370 dan S/390 dan disebut sebagai sistem operasi VM/ESA ( *Enterprise System Architecture*). VMware adalah suatu aplikasi yang memungkinkan kita untuk meng- *install* dua sistem operasi dan menjalakan aplikasinya (misalnya Windows and Linux) pada hardisk yang sama tanpa perlu logout dari SO yg lain. Xen adalah open source *Virtual Machine Monitor*, dikembangkan di University of Cambridge, untuk menjalakannya harus melalui dengan sebuah proses yakni pemodifikasian sistem operasi untuk lebih *full featured OSs*. Bahasa mesin untuk JVM disebut *Java bytecode* dengan keunggulannya yang bisa dijalankan pada berbagai jenis komputer atau *platform*. *.NET Framework* merupakan salah satu komponen yang tersedia dan terintegrasi pada sistem operasi Windows untuk kepentingan berbagai pengembangan aplikasi.

# Rujukan

[Silberschatz2005] Avi Silberschatz, Peter Galvin, dan Grag Gagne. 2005 . *Operating Systems Concepts*. Seventh Edition. John Wiley & Sons.

[WEBWiki2007] From Wikipedia, the free encyclopedia. 2007 . *Full Virtualization – http://en.wikipedia.org/wiki/Full_virtualization*. Diakses 15 Februari 2007.

[WEBWiki2007] From Wikipedia, the free encyclopedia. 2007 . *Partial Virtualization – http://en.wikipedia.org/wiki/Partial_virtualization.*. Diakses 15 Februari 2007.

[WEBCap-lore2007] Cap-lore. 2007 . *Short History of IBMs Virtual Machines – http://cap-lore.com/Software/CP.html.*. Diakses 15 Februari 2007.

[WEBWiryana2007] Wiryana Pandu. 2007 . *Komputer di dalam komputer – http://wiryana.pandu.org/indexe371.html*. Diakses 15 Februari 2007.

[WEBHarry2007] Harry Sufehmi. 2007 . *Virtualisasi – http://harry.sufehmi.com/archives/2006-07-29-1222/*. Diakses 28 Februari 2007.

[WEBFaculte2007] Sourythep Samoutphonh. 2007 . *VMware in the I and C School – http://ic-it.epfl.ch/bc2004/serveurs/vmware/index_en.php*. Diakses 12 Maret 2007.