Inhaltsverzeichnis

Virtualisierung

Virtualisierung

Virtualisierung bezeichnet Methoden, die es erlauben, Ressourcen eines Computers zusammenzufassen oder aufzuteilen.

Beispiele:

- Partitionierung
- Raids
- Virtual Local Area Networks, Virtual Private Network
- Speichervirtualisierung (MMU)
- Multitasking (Ringe)
- `chroot`, Jails

Motivation

- 1970er: Partitionierung von Mainframes
- Aufteilen der Ressourcen
- besseres Ausnutzen von Hardware
- Sicherheit
- Ausführen von alter Software
- Ausführen von Code auf anderen Plattformen
- Testen und Debuggen
- Virtualisierungs-Rootkits
- Blue Pill soll laufendes Windows in eine virtuelle Umgebung ohne Neustart verschieben können
- Erkennen evtl. über verlorene Rechenzeit

Grundidee

Emulation:

- Programm emuliert Prozessor („spielt CPU“)
- langsam, da Code erst analysiert werden muss
- Verbesserungen durch JIT-Compiler

Virtualisierung

- Code wird (größtenteils) nativ ausgeführt werden
- Gastsystem darf kein Unterschied feststellen können
- nur der Hypervisor hat zugriff auf alle Ressourcen
- Betriebssystem läuft im Userland, Hyperviser im Kernel-Mode
- Code, der Kernel-Mode-Rechte benötigt löst Exception aus $\Rightarrow$ Interrupt
- höher priviligiertes Programm (Hypervisor) entscheidet dann, was passiert

Problem auf x86

- es gibt Maschinenbefehle, die sich im User-Mode anders verhalten als im Kernel-Mode
- Beispiel: `popf` (Status der Flags widerherstellen)
- Problem: Userlevel-Prozess darf nicht Interrupt-Flag verändern
- Lösung: Maske im User-Modus
- das passiert allerdings ohne Interrupt (eigentlich Designfehler)
- Gastbetriebssystem versucht Interrupts zu deaktivieren
  $\Rightarrow$ schlägt schief, weil Gastsystem im Ring 3 läuft
  $\Rightarrow$ Hypervisor bekommt nichts davon mit
- Virtualisierungslücke (ca. 17 kritische Instruktionen)
- keine Virtualisierung unter x86

Vortrag zu Ende

Danke für die Aufmerksamkeit!

Lösung à la XEN

- Modifizieren der Gastsysteme
- anstatt kritische Befehle direkt auszuführen, wird der Hyperviser aufgerufen
- weniger Komplexität, evtl. höhere Performance
- properitäre Betriebssysteme können meist nicht ausgeführt werden
- $\Rightarrow$ Paravirtualisierung
- Hypervisor vertraut Gastsystem
- Userland läuft wie auf einem naiven System, nur Kernel muss angepasst werden

Lösung à la VMWare

- VMWare scannt den kompletten auszuführenden Code
- Scannen bis zum nächsten Sprung
- Überprüfen, ob kritische Befehle dabei sind
- Ersetzen des Codes zur Laufzeit durch Interrupts oder optimierten Code, der Befehl emuliert
- zwei Möglichkeiten:
  1. Ändern des Codes: Überschreiben durch Debug-Interrupt und Auffülen durch NOPs
  2. Erzeugen neuen Codes $\Rightarrow$ Länge des Codes ändert sich, Sprungcodes müssen angepasst werden
- Gastcode ist nach kurzer Zeit bereinigt (Schleifen. . .)
- zudem: Optimierung des Codes
- Guest-Tools: Paravirtualisierte Treiber (Grafik, Netzwerk, Festplattenzugriffe)
- komplex

Lösung à la Intel und AMD

- Prozessorvirtualisierung
- Hardware-Erweiterung wird von fast allen aktuellen CPUs von Intel und AMD unterstützt
  (Ausnahmen: einige Atom-CPUs, Semprons, Celerons)
- bei kritischen Befehle wird ein Interrupt ausgelöst
- Zustand beim wechseln zwischen Gast- und Host-Kontext wird gesichert und widerhergestellt
- Code wird nativ auf der CPU ausgeführt
- Hardwareerweiterung schließt Virtualisierungslücke
- Source-Code von Virtualisierungslösungen kann öffentlich sein
- bei Paravirtualisierung ist Hypervisor Single-Point-of-Failure und sicherheitskritisch

QEMU

- freie virtuelle Maschine
- *emuliert* komplette Hardware eines Computers
- unterstützte Prozessorarchitekturen: x86, AMD64 und x86-64, PowerPC, ARM, Alpha, m68k, MIPS, Sparc32/64
- QEMU-Manager und AQEMU als graphische Oberfläche

Hardware

- CD-ROM/DVD-Laufwerk (Image oder reales Laufwerk)
- Diskettenlaufwerk
- Grafikkarte (Cirrus CLGD 5446 PCI oder Standard-VGA-Grafikkarte)
- Netzwerkkarte (NE2000-PCI-Netzwerkadapter) und DHCP-Server
- Parallel- und Serielle-Schnittstelle
- Systemlautsprecher
- zwei PCI-ATA-Schnittstellen
- PCI und ISA-System
- PS/2-Maus und -Tastatur
- Soundkarte (Soundblaster 16, ES1370 PCI, GUS)
- USB-Controller (Intel SB82371, UHCI)

KVM

- Linux-Kernel-Infrastruktur für Virtualisierung
- stellt Schnittstelle zur Hardware-Virtualisierungstechniken bereit
- Kernel-Module kvm.ko (nutzt hardwarespezifischen Module kvm-intel.ko oder kvm-amd.ko)
- modifiziertes QEMU kann KVM nutzen
- Linux-Kernel wird zum Hypervisor für virtuelle Maschinen
- Code kann direkt auf der CPU ausgeführt werden

einfache Optionen

- `-fd{a,b}`
  Datei (Diskettenimage oder Device)
- `-hd{a|b|c|d}`
  Datei (Festplatten-Image oder Device)
- `-cdrom`
  Datei (CD-/DVD-Image oder Device)
- `-boot {a,b,c,d}`
  Bootreihenfolge festlegen
- `-m Mbits`
  Größe des Arbeitsspeichers für das Gast-System festlegen
- `-full-screen`
  im Vollbildschirm starten
- `-snapshot`
  Speichermedien vor Änderungen schützen
- `-enable-kvm`
  KVM Unterstützung aktivieren

Speichermedien

- Image aus vorhandener CD erstellen:
  `$ dd if=/dev/cdrom of=image.img`
- Image aus vorhandener Diskette erstellen:
  `$ dd if=/dev/fd0 of=image.img`
- Festplatteimage zur Verwendung in qemu erstellen:
  `$ qemu-img [-O cloop|vmdk|qcow|cow|raw] → name.img 5G`

Beispiele

- Auf Festplatte installiertes Betriebssystem starten:
  ```
  $ qemu -m 1024 -snapshot -hda /dev/sda
  -enable-kvm
  ```
- Knoppix mit SMB-Freigabe und Netzwerk starten:
  ```
  $ qemu -user-net -smb /home/michael/
  ```
- Eigene Linux Distribution starten:
  ```
  $ qemu -hda festplatte.img -kernel →
  /boot/vmlinuz -initrd /boot/initrd -append →
  'root=/dev/hda'
  ```

Monitor

- Erreichbar über `Strg+ALT+2` (`1` Emulation)
- `stop / c`
  Starten/Stoppen der Emulation
- `{load|save}vm`
  Speichern/Laden des Zustandes
- `sendkeys`
  Tasten an Emulation senden (z.B. `Strg+Alt+Entf`)
- `change`
  Geräte auswechseln
- `commit`
  Änderungen speichern
- `Screendump file`
  Screenshots schießen

Ausblick

- qemu unterstützt viele weitere Optionen (`man qemu`!)
- u.a.: USB, serielle Konsole
- kann als Daemon laufen
- VNC, serielle Schnittstelle
- kann mehrere CPUs emulieren
- dynamisch wachsende Images, Unterstützung für Images anderer Lösungen
- vielfältige Netzwerkoptionen
- Single-Step, Register betrachten, gdb

Vortrag zu Ende

Danke für die Aufmerksamkeit!

Links

- Chaosradio Express 092, Virtualisierung
- Webseite von KVM
- Webseite von QEMU
- online Buch über KVM und QEMU
- heise-Artikel über Virtualisierungs-Rootkit Blue Pill