Virtual Machines

Peter Schmid

Hochschule für Technik Zürich
Master of Advanced Studies, Informatik

21.12.2007

Outline

Definition, Abgrenzung

Es gibt zwei Kategorien von VMs:

- Virtuelle Hosts (Gast-Systeme)
- Laufzeitumgebungen für Programme (z.B. Java Virtual Machine)

Wir beschränken uns hier auf Virtuelle Hosts (System Virtual Machines, Hardware Virtual Machines).

Geschichtlicher Rückblick

- 1967: IBM CP/CMS auf CP-40
- 1972: IBM VM-CP, CMS (Single User OS) als Gast
- 199x: Virtualisierbare HW auf UNIX Sytemen (Sun, HP, IBM)
- 1997: Virtual PC für Aplle Mac von Connectix
- 1999: Virtual Platform von VMware
- 2002: VMware GSX
- 2003: MS übernimmt Connectix und stellt Virtual PC und Virtual Server vor
- 2003: Xen Open Source virtual machine monitor
- 2007: Citrix übernimmt XenSource

Virtualisierungstechnologien

- HW Emulation (z.B. Bochs, QEMU)
- HW Virtualisierung (z.B. VMware)
- Paravirtualisierung (z.B. Xen)
- Betriebssystemvirtualisierung (z.B. Solaris Containers)

VMM, Hypervisor

Die SW die die Virtualisierung vornimmt, nennt man Virtual Machine Monitor (VMM) oder Hypervisor.
Der Hypervisor läuft auf:

- direkt auf der HW (Type 1 oder Native VM, Bare Metal Environment)
- auf einem OS (Type 2 oder Hosted VM)

HW Emulation

- komplette HW wird emuliert
- Prozessor wird nachgebildet, Prozessor-Instruktionen werden emuliert
- sehr schlechte Performance
- nicht für produktive Systeme
- Bekannte Emulatoren: Bochs, QEMU, PearPC, DOSbox

HW Virtualisierung

- Andere Bezeichnungen: Full, Transparent oder Native Virtualization
- die meisten Instruktionen werden direkt auf dem Prozessor ausgeführt "direct execution"
- VMM läuft im priviligierten Modus (Kernel, Ring 0 auf x86)
- Gast-OS können keine priviligierten Befehle ausführen, diese werden vom VMM abgefangen und dem Prozessor übergeben (Modifikation OS während der Laufzeit), Ring 3 auf x86
- Bekannte VMMs: VMware, MS Virtual PC/Server

Paravirtualisierung

- Gast-OS muss angepasst werden (priviligierte Befehle)
- priviligierte Befehle werden in Hypercalls umgesetzt
- HW Unterstützung durch Prozessoren (AMD-V und Intel VT-x), Gast-OS läuft auf Ring 0D
- der grösste Teile der Instruktionen wird direkt auf dem Prozessor ausgeführt "direct execution"
- VMM läuft im priviligierten Modus (Kernel, Ring 0P auf x86)
- Bekannte Paravirtualisierer: Xen (und damit XenSource, XenEnterprise, OracleVM, xVM), VirtualIron

Virtualisierung auf Betriebssystemebene

- Nur ein BS-Kernel
- ist am effizientesten (Leistungseinbusse 1..3 %)
- Abhängig vom Hostsystem
- Bekannte BS-Virtualisierer: Sun Solaris Container, FreeBSD Jails, Virtuozzo, Linux VServer

Installation von Gast-BS

- VM Umgebungen stellen standardisierte Treiber zur Verfügung (Disk, Graphik usw.)
- Disks werden von VMM verwaltet, Snapshots möglich
- einfach von Maschine zu Maschine verschiebbar
- Nicht alle HW wird unterstützt (z.B. USB)
- Aufwändige Management-SW für kommerzielle VMM verfügbar

Vorteile und Nutzen der Virualisierung

- geringere HW-Investitionen
- weniger HW-Wartungsverträge
- reduzierter Platzbedarf
- geringerer Energiebedarf (kleinere USV, kleinere Klimaanlage)
- weniger Verkabelung und Netzwerkports
- Steigerung der Flexibilität
- schnelle Bereitstellung
- schnellere Wiederherstellung
- kostengünstige Testumgebung
- Ablösung von Legacy Servern (z.B. fehlende Treiber für WinNT4)

Grenzen der Virtualisierung

- zusätzliche Abstraktionsstufe
- Performance-Verlust
- Pferd von hinten aufgezäumt, BS sollten eigentlich Dienste genügend voneinander isolieren.
- HW wird nicht optimal genutzt (v.a. neuere)
- fehlende Treiber