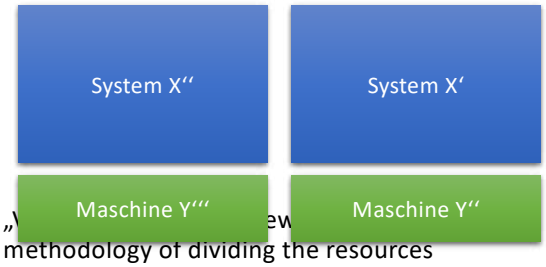
2. Virtualisierung

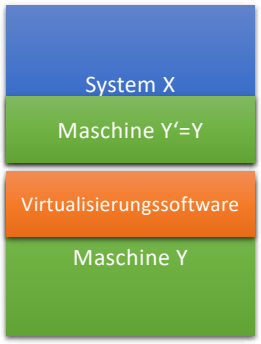
2

Virtualisierung



„V ew
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.”

Amith Singh



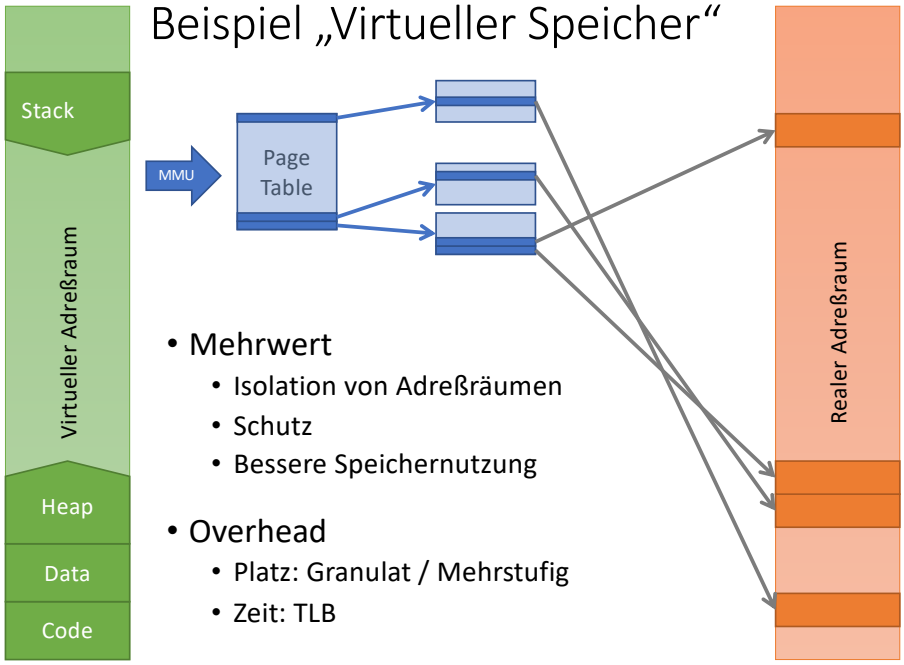
Technik ist recht verbreitet!

- Virtualisierung einzelner Ressourcen
  - Terminal = Window
    - Aktives Fenster hat Fokus (=Keyboard und Maus)
  - CPU = Virtueller Prozessor = Thread
  - Adreßraum Speicher = Virtual Memory
- Virtualisierung ganzer Rechner = Virtuelle Maschine
  - 16 Bit Windows und DOS-Anwendungen
  - Vielfältig

Ressourcen-Virtualisierung

- Bessere Ausnutzung
  - Kontextwechsel bei blockierendem Aufruf
  - Nebenläufige Anwendungen profitieren von Multiprozessoren
    - … aber laufen auch auf Monoprozessoren
- Eliminieren von Engpässen
  - Mehr Speicher durch Paging
  - Jede Anwendung bekommt ihr eigenes Terminal
- Schutz / Isolation
  - Anwendungen untereinander isoliert
  - Betriebssystem vor defekten/bösartigen Anwendungen geschützt

Beispiel „Virtueller Speicher“



- Mehrwert
  - Isolation von Adreßräumen
  - Schutz
  - Bessere Speichernutzung
- Overhead
  - Platz: Granulat / Mehrstufig
  - Zeit: TLB

Virtualisierungsarten

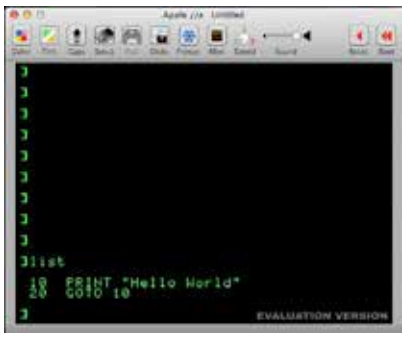
- Emulation
  - Vollständige Simulation anderer CPU und Hardware
- Native Virtualisierung (Full Virtualization)
  - Keine Änderung des Gastsystems (=Transparenz)
- Paravirtualisierung
  - Gastsysteme sind sich ihrer Virtualisierung bewußt
  - Transparenz oberhalb des Gastes
- OS-Level-Virtualisierung
  - Betriebssystem virtualisiert mehrere Instanzen seiner selbst
- Anwendungsvirtualisierung

Emulation

9

Emulation

- Interpretation
  - Emulatoren für Atari, VC64, Apple II, …

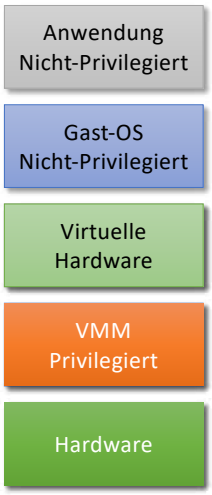


- Übersetzung
  - Rosetta
    (PowerPC -> Intel)
  - Rosetta 2
    (Intel -> ARM)
  - WOW64
    (32 Bit Windows auf Itanium 2)
  - …

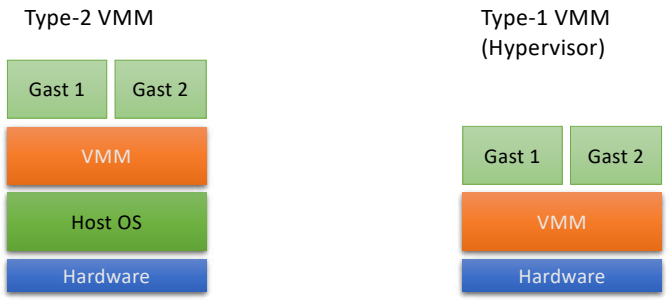
- Performanz

Native Virtualisierung

- Befehlssatz Gast = Befehlssatz Host
- Voraussetzungen
  - Privilegierter und nicht-privilegierter Modus
  - Gut virtualisierbare CPU ;-)
- Gast-OS führt privilegierte Instruktion aus
  - VMM interpretiert Befehl
- Für Gast-OS unsichtbar



Typ-1 und Typ-2



Vorteile

- Server-Konsolidierung
- Testen und Debugging
- Isolation
  - Sandboxing
  - Fault / Error Containment
- Ausführung von Legacy Software
  - Alte Anwendungen
  - Alte Betriebssysteme
- Indirektionsstufe
  - Migration
  - Quality of Service
  - Lastverteilung
  - Administration
  - Automatisierung
- Schulungen
- Auslieferungsmedium für Anwendungen
  - Einsichten in neue Software

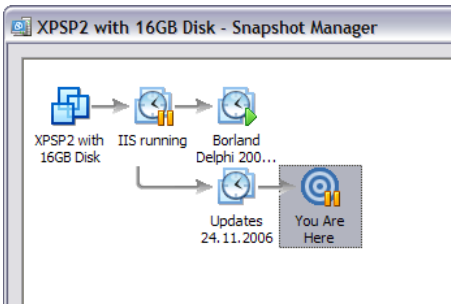
Nachteile

- Schlecht virtualisierbare Hardware
- Bereits im Gast-OS genutzte Virtualisierung
- Zeit- und Platzeffizienz
- Management
  - Updates

Details

- 1959 bis 1970, IBM federführend, aber auch MIT u.a.
- Geburtsstunde des Multi-Programming und Time-Sharing
  - Atlas Project, Manchester (1961)
  - Multics, MIT (1963)
  - m44/44X, IBM 704 Serie, CTSS, CP-40/67, VM/370 IBM (ca. 1965)
    - Mehrere identische Kopien der Hardware
- Kommende OS-Generation noch zu jung und instabil
- Schnelle Nutzung der besser werdenden Hardware
- Versionen mittels virtuellen Maschinen replizieren

Beispiele



- VMware, Parallels, ...
  - Fortgeschrittene Konzepte
    - Drag and Drop
    - Schnappschüsse
    - Clones
    - Multimedia
    - Virtuelle Rechnernetze
- Microsoft Virtual PC und Virtual Server
- Virtual Box von Sun

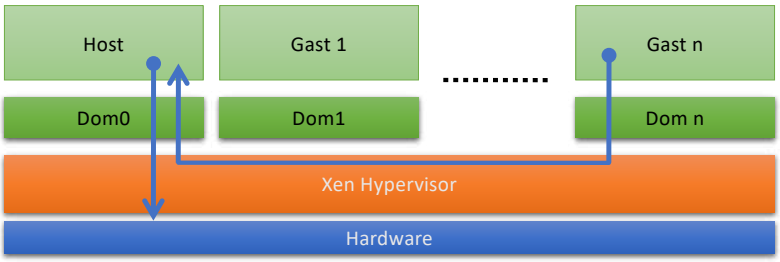
Kritische Instruktionen

- Reale CPUs mehr oder weniger gut virtualisierbar
- x86 eher weniger ☺
- Gründe
  - Nicht-privilegierte Instruktionen geben Auskunft über privilegierte Hardware-Informationen (Interrupts, …)
  - Instruktionsresultat abhängig vom Ausführungsmodus
  - Instruktionen verändern versteckten Prozessorzustand
- Intel insgesamt 17 kritische Instruktionen
  - Ausführung löst keine Exception aus
  - Für VMM schwer erkennbar (z.B. aufwendige Filterung)

Paravirtualisierung

- Gast-System muß angepaßt werden
- Vorteile
  - Ungünstige Hardware-Eigenschaften abschwächen bzw. aufheben
  - Kritische Instruktionen vermeiden
  - Geringe Effizienzverluste
- Nachteile
  - Zugang zum Sourcecode notwendig
- Bedeutendster Vertreter: Xen
- Starkes Interesse seitens VMware und Microsoft

Xen



- Open Source Projekt der Universität Cambridge
- Volle Virtualisierung mit HW-Unterstützung möglich
- Spezielles Hostsystem in Dom0
  - stellt in der Regel Treiber für E/A bereit
  - Führt diverse Xen-Prozesse aus
- Gäste greifen über virtuelle Treiber und Host auf Geräte zu
  - Shared Memory und Events

Windows Enlightenment

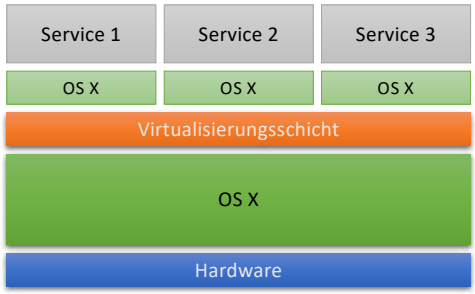
- Paravirtualisierbares Windows
- Seit Longhorn (Codebase Vista für Server)
- Verlagerung der Treiber in Gast-OS
  - Virtualization Provider
  - Virtualization Client
  - Direkter HW-Zugang für Gäste
    - Direct3D bzw. DirectX für Gäste zugänglich !!!

OS Layer

24

OS-Level Virtualisierung

- Dünne Virtualisierungsschicht oberhalb OS
  - Virtual Environments, Virtual Private Servers, Jails, Zones, Containers, …
- Kommerzielle Lösungen
- Bemerkungen
  - Leichtgewichtig
  - Vergleichsweise komplex
  - Vorgeschaltete Kerneltreiber fangen alle Aufrufe ab

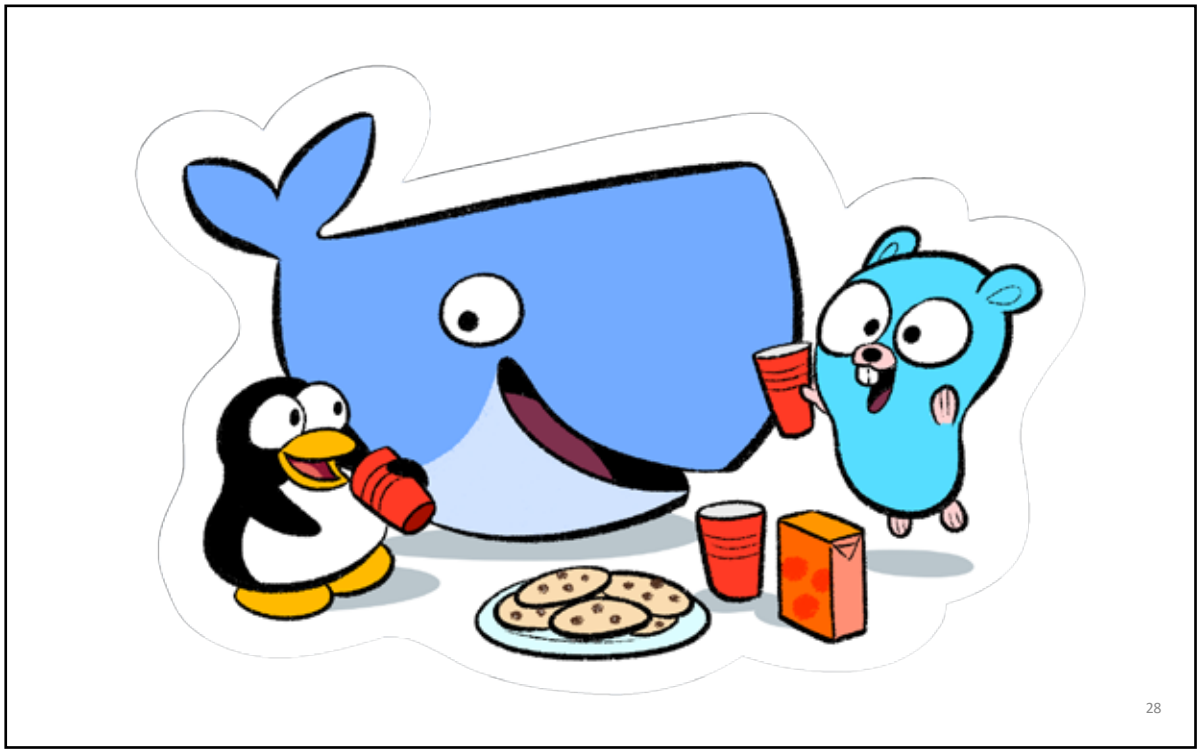


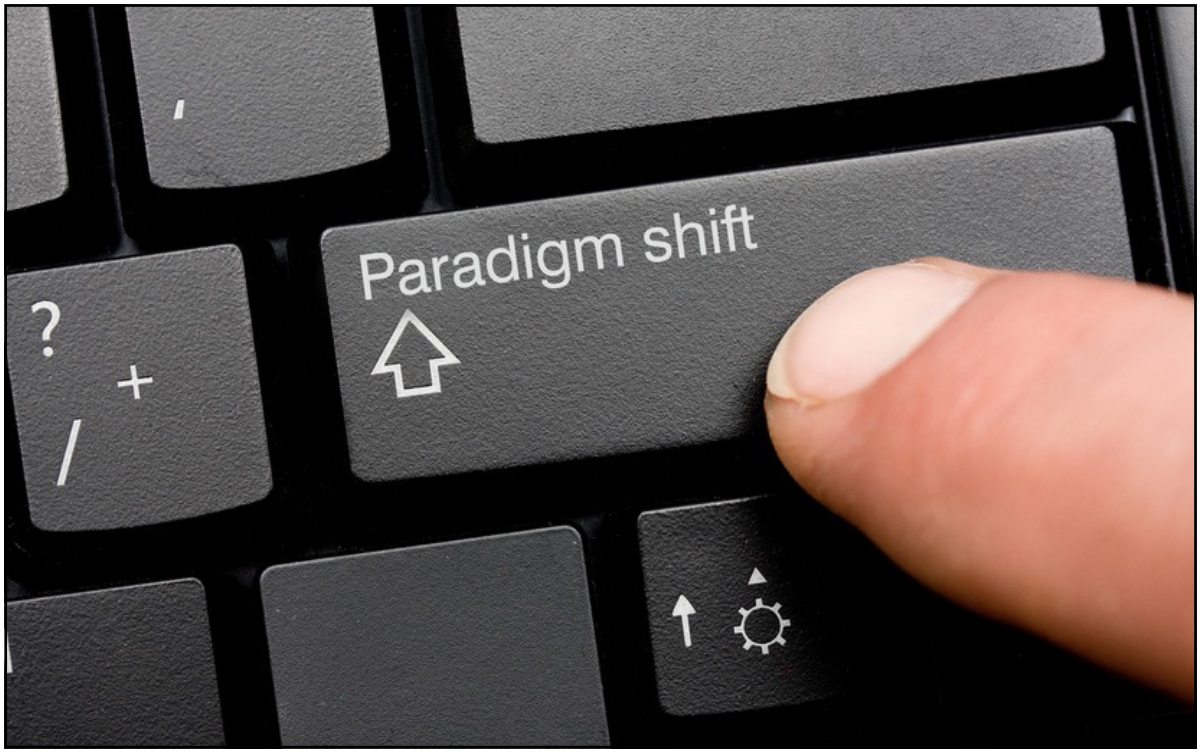
Produkte

Mechanism	Operating system	Features						
		File system isolation	Disk quotas	I/O rate limiting	Memory limits	CPU quotas	Network isolation	Partition checkpointing and live migration
chroot	most UNIX-like operating systems	Yes	No	No	No	No	No	No
FreeVPS	Linux	Yes	?	?	?	?	?	?
Linux-VServer (security context)	Linux	Yes	Yes	Yes/No [1]	Yes	Yes	Yes	No
OpenVZ (virtualization, isolation and resource management)	Linux	Yes	Yes	No	Yes	Yes	Yes[2]	Yes
SWsoft Virtuozzo	Linux, Windows	Yes	Yes	Yes	Yes	Yes	Yes[2]	Yes
ContainerZone	Solaris	Yes	Yes	No	Yes	Yes	Yes[2]	?
FreeBSD Jail	FreeBSD	Yes	No	No	No	No	Yes	No

Wikipedia


27


28



... kriegen wir später

30

Application-Layer

31

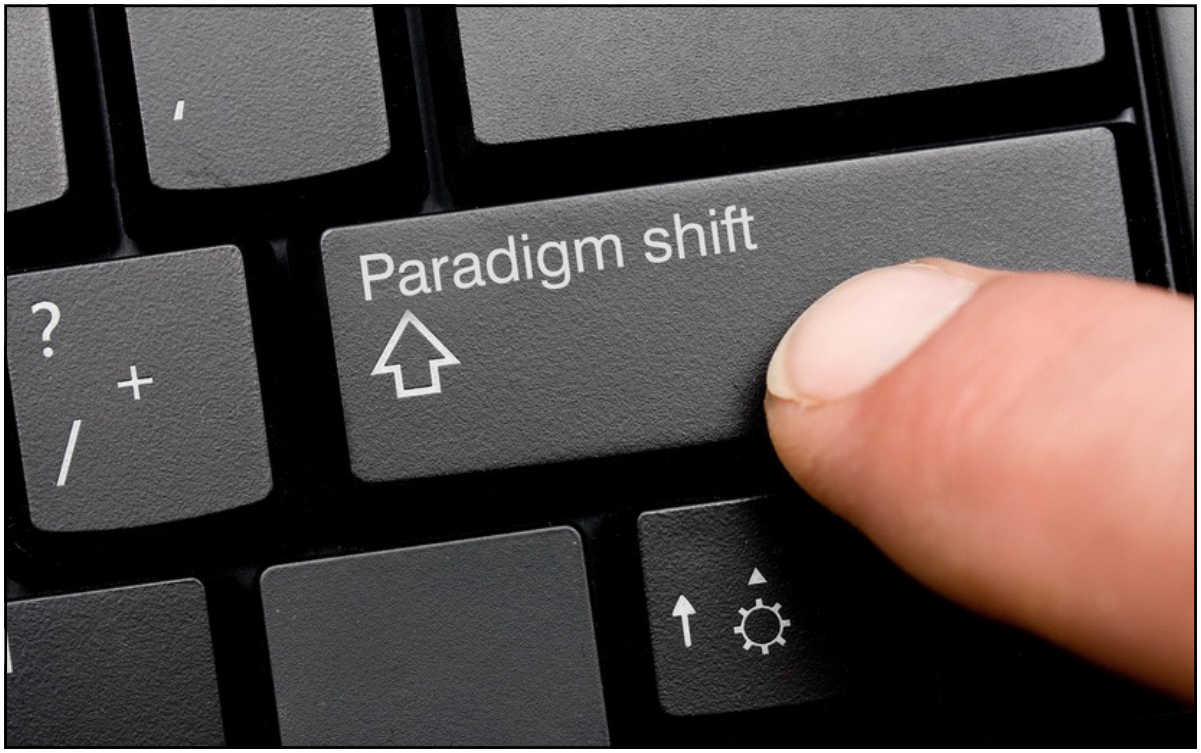
Application Virtualization

- Viel Ähnlichkeit mit Emulation, aber …
  - keine Nachbildung eines vorhandenen Befehlssatzes
  - sondern eigenständige, problemspezifische Lösung
- Abstrakte Maschinen
- Wichtigste Vertreter
  - Java Virtual Machine (JVM)
  - .NET Common Language Runtime (CLR)
- DIE Laufzeitplattformen der Gegenwart

Beispiele

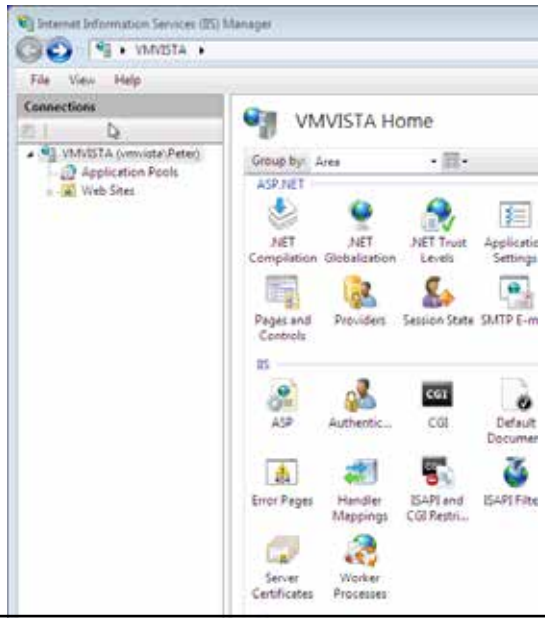
Virtual machine	Languages	Comments
Common Language Runtime	C#, Visual Basic .NET, J#, Managed C++, Python	reference implementation by Microsoft
Corn concurrent runtime machine	Corn language	
Forth virtual machine	Forth	
Glulx	Glulx, Z-code	
Inferno	Limbo	
Java virtual machine	Java	reference implementation by Sun
Low Level Virtual Machine (LLVM)	currently C, C++, Stacker	
Lua		
Macromedia Flash Player	ActionScript, SWF (file format)	interactive web authoring tool
MMIX	MMIXAL	
Neko virtual machine	currently Neko and haXe	
O-code machine	BCPL	
P-Code machine	Pascal	
Parrot	Perl 6, and others experimentally	
Perl virtual machine	Perl	
Portable.NET	C#, Visual Basic .NET, J#, Managed C++	
Python virtual machine	Python	
ScummVM	Scumm	computer game engine
SECD machine	ISWIM, Lispkit Lisp	
Smalltalk virtual machine	Smalltalk	
Sqlite virtual machine	Sqlite opcodes	
Squeak virtual machine	Squeak	
TrueType virtual machine	TrueType	font rendering engine
Valgrind	x86/x86-64 binaries	checking of memory acesses and leaks under Linux
VX32 virtual machine		application-level virtualization for native code
Waba		Virtual machine for small devices, similar to Java
Warren Abstract Machine	Prolog	
Z-machine	Z-Code	

Wikipedia



Weitere Beispiele

- Web-Server
  - Virtual Directory
  - Virtual Host
- Application Server
  - Enterprise JavaBeans
  - Weitere Indirektionsstufe
- WPF und WF aus .NET 4.5
  - Instruktionssätze
    - XAML
    - XAML Presentation
    - XOML



ABI/API-Virtualisierung

- WINE
  - Windows API auf UNIX/Linux und X
- CrossOver
  - Kommerzielle Version
- SUN WABI
  - Windows Application Binary Interface
  - für x86
  - Emulation auf SPARC
- …

Diskussion

37

Sicherheit

- VMWare Software
  - Remote Heap Exploit in vmnat.exe
  - Angreifer kann eine virtuelle Maschine verlassen und Host kompromittieren
- Angriffe auf JVM und .NET CLR durch Fehler in Speicherverwaltung
  - S. Govindavajhala, A.W. Appel, Princeton, 2003

Blue Pill Attack

- Joana Rutkowska, COSEINC
- Nutzt Hardwareunterstützung für Virtualisierung
  - Malware wird kleiner Hypervisor
  - OS kommt in eine virtuelle Umgebung und wird kontrollierbar
  - Geht „On the fly“
- Exploit auf AMD64 SVM über Vista
- Schutz mit zusätzlicher HW-Unterstützung möglich
  - Authenifizierung auf HW-Ebene beim Einrichten neuer VMs

Mehr Funktionalität

- Migration virtueller Maschinen im laufenden Betrieb
- Konvertierung
  - Aus realer Installation wird virtuelle Installation
  - Aus virtueller Installation wird reale Installation
  - Zwischen verschiedenen virtuellen Formaten

Mehr Standards

- Offenlegung der Formate virtueller Festplatten
- Management komplexer virtueller Infrastrukturen

Mehr HW-Support

- Paravirtualisierung bei Hardwaretreibern
  - Zugriff auf 3D-Features moderner Graphikkarten in der virtuellen Maschine
- Direkter Support in modernen CPUs

Hardwareunterstützung

- Support für virtuelle Maschinen
  - Unzulänglichkeiten der x86-Architektur beseitigen
  - Intel VT (früher Vanderpool)
  - AMD V (früher Pacifica)
- IOMMU
  - Virtuelle Adreßabbildung und Schutz bei DMA-Zugriffen
  - Legacy 32-Bit-Support auf 64-Bit-Systemen
  - Direkter aber geschützter Zugriff virtueller Gäste auf Hardware
  - Direkter aber geschützter Zugriff von Anwendungen auf Hardware

Intel VT

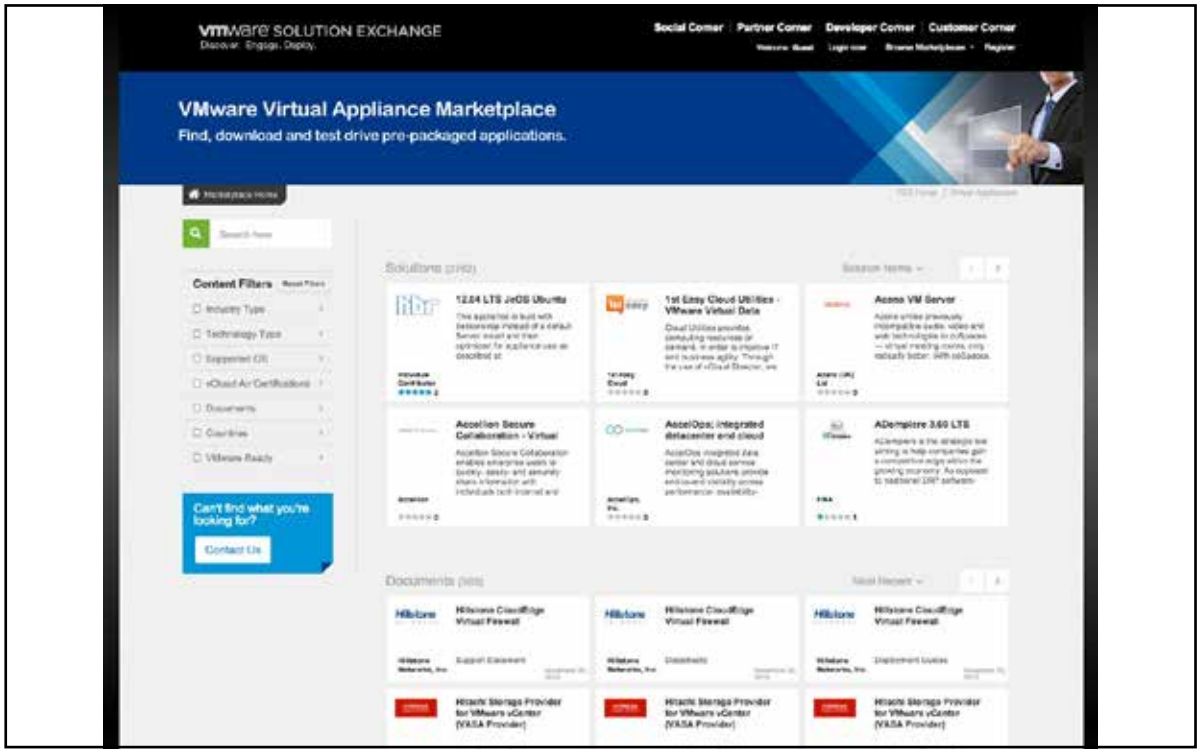
- Zwei Realisierungen
  - x86-Architektur: VT-x
  - Itanium: VT-i
- VM Exits sind konfiguierbar
  - Welche Exception soll Exit auslösen?
  - Welcher I/O-Zugriff?
  - Welche maschinenspezifischen Register sind geschützt?
  - Welche kritischen Instruktionen?

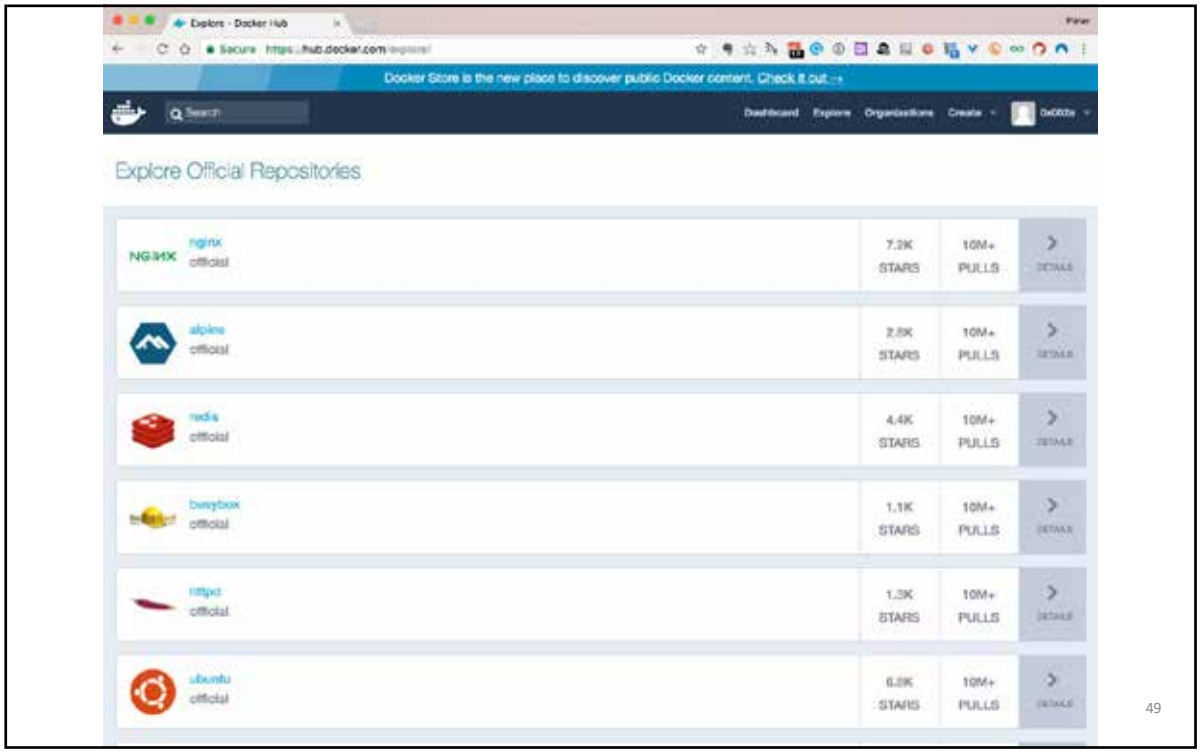
Visionen?!

- Virtuelle Maschinen werden immer „billiger“
  - Ausreichend viele CPUs vorhanden
  - Kompakte Images
    - Snapshots werden als Deltas gespeichert
    - Mehrere VMs nutzen gleiche Codebasis
- Hypervisor + Virtuelle Maschinen wird Normalfall
- „Wenn der Prophet nicht zum Berg kommt, …“
  - Betriebssysteme erlauben saubere Anwendungsstrukturen
  - Aber viele Anwendungen werden „unsauber“ realisiert
  - Jede Anwendung bekommt eigene virtuelle Hardware
- Virtuelle Maschinen werden Einheiten des Deployments

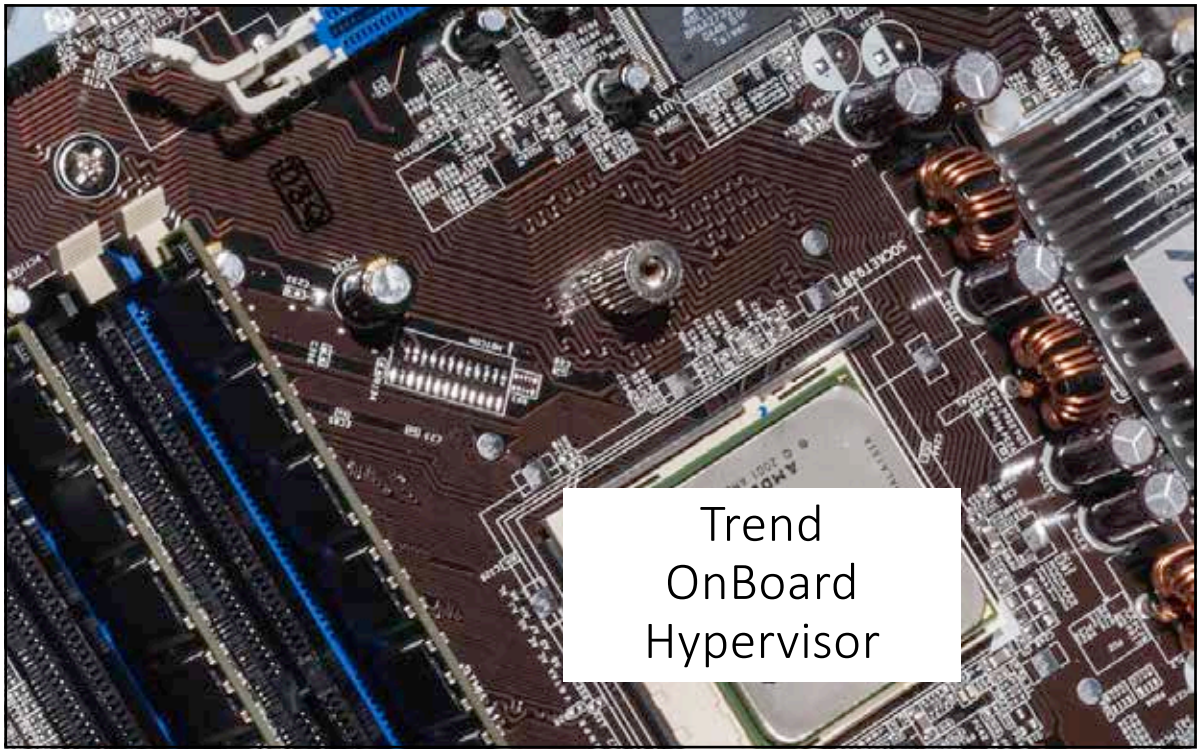
Trend: VM als Einheit des Deployment?

- **O**pen **V**irtual Machine **F**ormat (OVF)
  - Portabel und HV-unabhängig
  - XML-basierter Standard zum Austausch virtueller Maschinen
  - Unterstützt „Virtual Appliances“ / „Software Appliances“
    - *„Nicht nur Betriebssysteme sondern Applikationen“*
- Virtual Appliances
  - VM-Ensembles als einheitliche Applikation
  - Beispiel CRM-Applikation
    - Web Server, Datenbank-Server, Frontend, …
    - Jede Komponente eine VM
  - Einfache „Installation“


vmware SOLUTION EXCHANGE
Discover. Engage. Deploy.
Social Corner | Partner Corner | Developer Corner | Customer Corner
VMware Virtual Appliance Marketplace
Find, download and test drive pre-packaged applications.
Content Filters
Can't find what you're looking for?
Contact Us

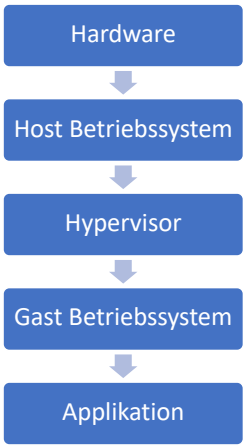

Docker Store is the new place to discover public Docker content. Check it out.→
Explore Official Repositories
nginx
official
STARS
PULLS
alpine
official
redis
official
busybox
official
httpd
official
ubuntu
official

49



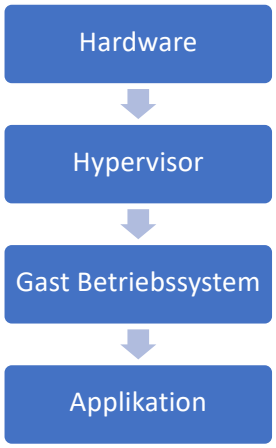
Trend: Hypervisor wandelt sich

- 1. Generation
  - Hosted Typ-2-Hypervisor



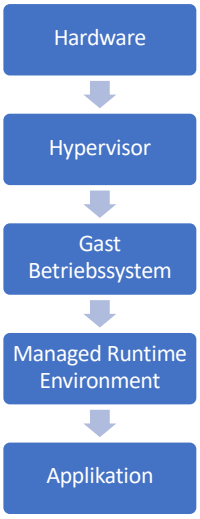
Mehr Typ 1?

- Native Typ-1-Hypervisor



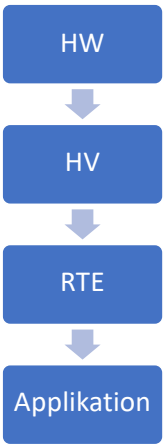
Trend …

- Managed Code



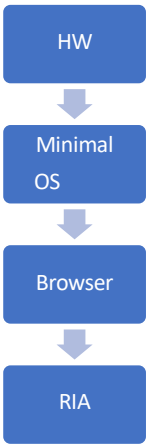
Trend: Gast-OS wandelt sich

- Beispiel ESX Server 3i
  - Teilmenge der POSIX OS-API
  - VSockets über VMCI (zukünftig)
  - Microkernel++?
- Canonical JeOS
  - **J**ust **E**nough **O**perating **S**ystem
  - Minimal-Ubuntu für Virtual Appliances
  - Für ESX Server 3i Hypervisor
- BEA Liquid VM
  - JVM für den Hypervisor
  - Für Virtual Appliances
- Microsoft Server Core
  - Konfigurierbarer minimal Windows Server ohne GUI



Oder noch schlimmer ...

- ... Browser ist das Betriebssystem
  - Vergleichbar JVM oder CLR
- Beispiel Chrome
- Trend „Rich Internet Applications“ (RIAs)
  - Aspekt verteilter Systeme
  - Integration von Clouds bzw. *aaS

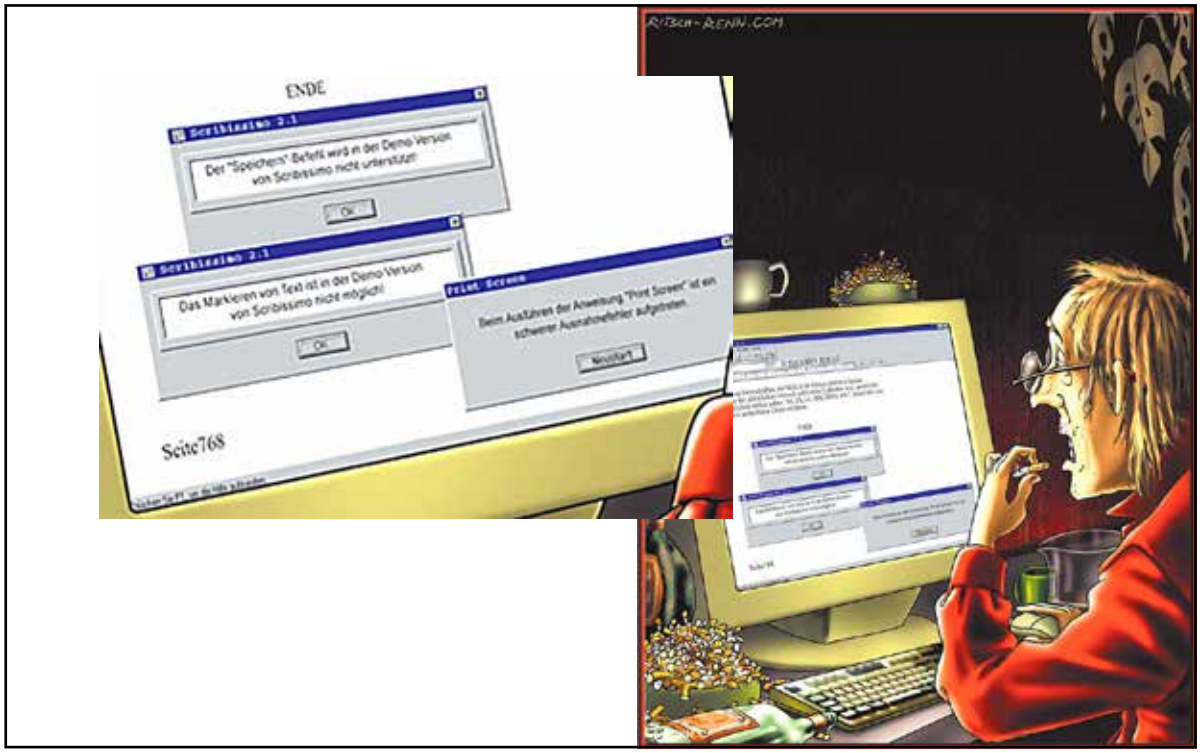


Virtuelle Zeiten

- Anwendungen werden als vorkonfigurierte virtuelle Maschinen ausgeliefert
  - Bessere Isolation
  - Weniger Wechselwirkungen
  - Erhöhte Flexibilität
  - Erhöhte Mobilität
- Viele VMs laufen auf jeweils einem Rechner
  - Zeit- und Platzeffizienz
- Renaissance in Forschung und Entwicklung
- Virtualisierungstechniken sind neuer alter Trend

Record/Replay (VMWare)

- Komplette Kontrolle der Zeit aus Sicht einer VM
  - Vollkommen deterministische Ausführung
- Einsatzszenarien
  - Software-Tests / Debugging
    - Exakte Reproduzierbarkeit von Fehlern
    - Race-Conditions
    - Als Log an Entwickler liefern
    - Debugger nachträglich einhängen
  - Consumer
    - Recovery im täglichen Einsatz?
    - VM-basierte Wiederherstellungspunkte?


ENDE
Der "Speichern" Befehl wird in der Demo Version von Scribissimo nicht unterstützt
Ok
Das Markieren von Text ist in der Demo Version von Scribissimo nicht möglich!
Ok
Print Screen
Beim Ausführen der Anweisung "Print Screen" ist ein schwerer Ausnahmefehler aufgetreten.
Neustart
Seite768
RITSCH-RENN.COM