Betriebssysteme

Kapitel 7 Virtualisierung und Cloud

Virtualisierung

- Einführung
  - -Historischer Überblick
  - -Betriebssystemkonzepte
- Prozesse und Threads
  - -Einführung in das Konzept der Prozesse
  - -Prozesskommunikation
  - -Scheduling von Prozessen
  - -Threads
- Speicherverwaltung
  - -Einfache Speicherverwaltung
  - -Virtueller Speicher
  - -Segmentierter Speicher
- Dateien und Dateisysteme
  - -Dateien
  - -Verzeichnisse
  - -Implementierung von Dateisystemen
- Grundlegende Eigenschaften der I/O-Hardware
  - -Festplatten
  - -Terminals
  - -Die I/O-Software
- Deadlocks/Verklemmungen
- Virtualisierung und die Cloud
- IT-Sicherheit
- Multiprozessor-Systeme

Aufgabe/Frage

- Überlegen Sie welche Gründe es gibt, daß Virtualisierung sich in der Informatik durchgesetzt hat?

Bedingung:

→ 2 min

Nachteile Virtualisierung

- Viel HW-Leistung
- Lizensierungsproblematik
- Schnelles Netzwerk notwendig
- Programme, die nicht virtualsiert laufen
- Erhöhtes Ausfallrisiko (es fallen gleich viele Server aus)
- Anpassung des Toolings

Virtualisierung

- ‚Geschichte‘ der Virtualisierung
  - 1969: CP-67 ("control program") für IBM System/360-67 (1969!) CP/CMS wurde mit Einplatzsystem CMS für Teilnehmerbetrieb eingesetzt und später VM/370 genannt (Mainframe S/390, zSeries)
  - 1972: z/VM wird Nachfolger von VM/370 für die aktuelle Großrechner-Familie IBM zSeries
  - 1999: VMware WSX für Intel x86
  - 2003: Xen realisiert Paravirtualisierung für x86
  - ...

Virtualisierung

Definition Virtualisierung (Wikipedia)
Virtualisierung bezeichnet in der Informatik die Nachbildung eines Hard- oder Software-„Objekts“ durch ein ähnliches Objekt vom selben Typ mit Hilfe einer Software-Schicht. Dadurch lassen sich virtuelle (d. h. nicht-physische) Dinge wie emulierte Hardware, Betriebssysteme, Datenspeicher oder Netzwerkressourcen erzeugen. Dies erlaubt es etwa, Computer-Ressourcen (insbesondere im Server-Bereich) transparent zusammenzufassen oder aufzuteilen, oder ein Betriebssystem innerhalb eines anderen auszuführen.

Virtualisierung

Hardwarevirtualisierung

- Systemvirtualisierung (Computersystemvirtualisierung oder Betriebssystemvirtualisierung)
  - Computervirtualisierung: mithilfe von Software wird ein virtuelles Computersystem erstellt
  - Betriebssystemvirtualisierung erstellt virtuelle Instanzen einer Betriebsumgebung

- Prozessorvirtualisierung

- Speichervirtualisierung

Quelle: http://de.wikipedia.org/wiki/Virtualisierung_(Informatik)

Unterstützende Hardwaretechnologien

- Blades nutzen die gleichen Ressourcen, werden zentral verwaltet und haben eine gemeinsame Stromversorgung und Lüftung. So kann beispielsweise ein Standard-42U-19″-Rack bis zu 84 Blades aufnehmen und bis zu 1344 Prozessorkerne enthalten.

- Server muss keine Lüfter in den Blades haben, sondern die Lüfter sind redundant an der Rückseite des Gehäuses eingebaut

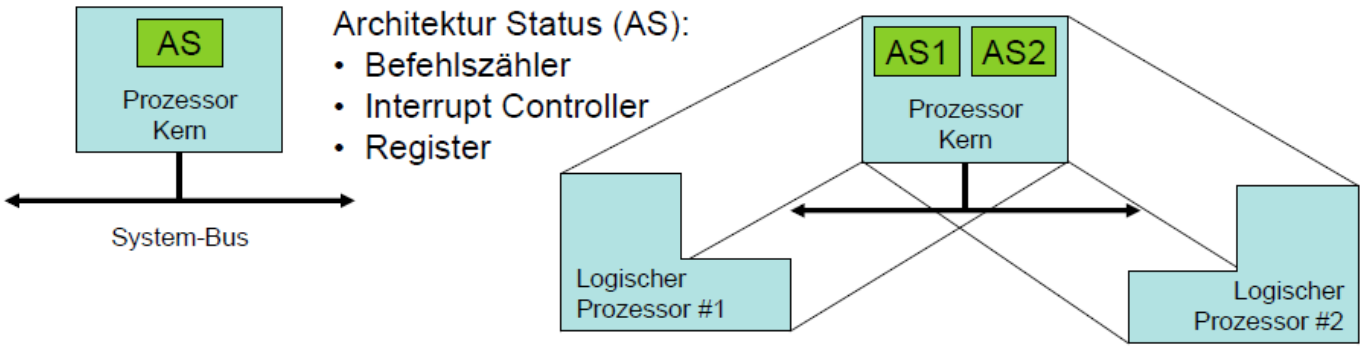
- Beispiel: Bladesystem (HP), BladeCenter (Lenovo)

Unterstützende Hardwaretechnologien: Hyperthreading (1/2)

- Prozessoren sollen besser ausgelastet werden, indem die Lücken in der Pipeline mit Befehlen eines anderen Threads aufgefüllt werden
- Softwareseitig verhält sich eine CPU mit Hyper-Threading wie ein Symmetrisches Multiprozessorsystem
- Hyper-Threading bringt jedoch nur für Anwendungen einen Geschwindigkeitsvorteil, deren Berechnungen parallelisierbar sind, das heißt die Berechnung eines Threads ist nicht abhängig vom Ergebnis eines anderen

Unterstützende Hardwaretechnologien: Hyperthreading (2/2)

- Durchschnittliche Auslastung einer Execution-Unit: ca. 35 %*
  Hyper-Threading Technologie



- Leistungssteigerung durch Hyper-Threading ca. 30 %*
- Grenzen bestehen durch sharing von Prozessor-Ressourcen;
  z.B. L1-, L2-Cache, Excecution Pipeline

**Quelle: Intel*

Softwarevirtualisierung

- Hardware-Emulation
- Betriebssystemvirtualisierung mittels Betriebssystem-Container
- Systemvirtualisierung mittels Virtual Machine Monitor (VMM)
- Hardware-Virtualisierung (native Virtualisierung, full Virtualisierung)
- Paravirtualisierung

- Anwendungsvirtualisierung

Quelle: http://de.wikipedia.org/wiki/Virtualisierung_(Informatik)

Aufgabe

- Wir sprechen in der IT immer wieder von Simulation, Emulation und Virtualisierung.

- Beschreiben Sie die Begriffe in Ihren eigenen Worten.
  - Was ist eine Simulation?
  - Was ist eine Emulation?
  - Was ist eine Virtualisierung?
  - Geben Sie Beispiele!

Software-Emulator (Wikipedia)

Software-Emulatoren sind Programme, die einen Computer nachbilden und es so ermöglichen, Software für diesen Computer auf einem Computer mit einer anderen Architektur zu verwenden.

Beispiele:

+ Spiele für ältere Spielekonsolen können auf einem PC oder einer neueren Spielekonsole ablaufen.
+ Ein Softwareentwickler kann bei der Entwicklung eines Programmes für ein Gerät (z. B. ein Handy), das eine andere Architektur als der Entwicklungs-Rechner hat, dieses im Emulator testen und korrigieren, ohne es jedes Mal auf das Gerät kopieren zu müssen.

Hardware-Emulator (Wikipedia)

Ein Hardware-Emulator ist ein elektronisches Gerät, das ein System wie einen Drucker oder einen Prozessor (CPU) funktionell, elektrisch oder mechanisch (Gehäuse und Pins) nachbilden kann. Die Verbindung zur Prozessorbaugruppe wird mittels Sockel und passendem Stecker erstellt. Er wird auch als *In-Circuit-Emulator* (ICE) bezeichnet.

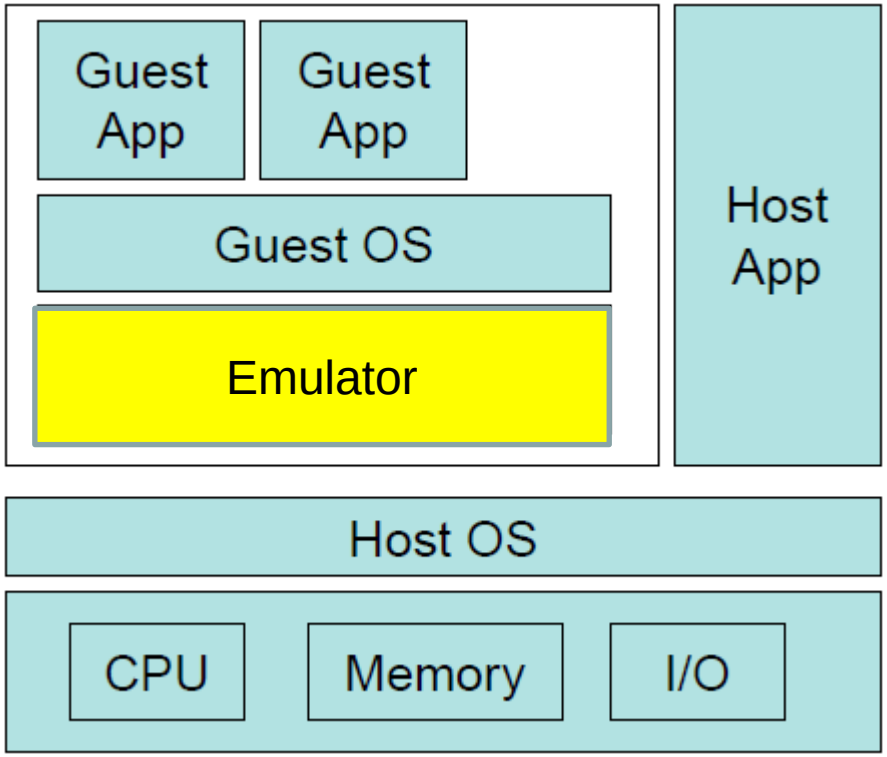
Terminal-Emulator (Wikipedia)

Ein Terminalemulator ist eine Software, welche die Funktion eines Terminals (Dateneingabe/Bildschirmausgabe) nachbildet, so dass man z. B. von einem PC auf eine entsprechende Anwendung zugreifen kann.

Hardware-Emulation

- Bei der Emulation wird in der meisten Fällen versucht, die komplette Hardware eines Rechensystems funktionell nachzubilden und so einem unveränderten Betriebssystem, das für eine andere Hardwarearchitektur (CPU) ausgelegt ist, den Betrieb zu ermöglichen
- **Vorteile**
  - Keine Anpassungen am Betriebssystem bzw. den Anwendungen nötig sind
  - andere Architekturen verwenden (nicht in Hardware existierende bzw. hardwaretechnisch vorhandene)
- **Nachteile**
  - Entwicklung von Emulationsumgebungen sehr aufwändig ist
  - Ausführungsgeschwindigkeit in der Regel deutlich geringer ist, gegenüber Virtualisierungslösungen

Virtualisierungstechnik: Emulation



Idee

Architektur wird unabhängig von tatsächlicher Hardware vollständig in Software abgebildet

Vorteil

- Unabhängigkeit

Nachteile

- aufwändige Entwicklung
- Performanceverluste
- Lizenzrechte

Auswahl an Emulatoren (kein Anspruch auf Vollständigkeit!

Name	Lizenz	Host	Emulierte Architektur	Gast-System
Bochs v2.3.6	LGPL	Linux, Solaris, MacOS, Windows, IRIX, BeOS	x86, AMD64	Linux, DOS, BSD, Windows, BeOS
QEMU v0.9.0	GPL	Linux, BSD, Solaris, BeOS, MacOS-X	x86, AMD64, PowerPC, ARM, MIPS, Sparc	Linux, MacOS-X, Windows, BSD
DOSBox v0.72	GPL	Linux, Windows, OS/2, BSD, BeOS, MacOS-X	x86	DOS
DOSEMU v1.4.0	GPL	Linux	x86	DOS, Windows bis 3.11
PearPC v0.4.0	GPL	Linux, MacOS-X Windows	PowerPC	Linux, MacOS-X, BSD
Basilisk II v0.9-1	GPL	Linux, diverse UNIX, Windows NT4, BeOS, Mac OS, Amiga OS	680x0	MacOS $\leq$ 8.1
Wabi v2.2	proprietär	Linux, Solaris	x86	Windows 3.x
MS Virtual PC v7	proprietär	MacOS-X	x86	Windows, (Linux)
M.A.M.E. v0.137	MAME-Lizenz	Linux, Windows, DOS, BeOS, BSD, OS/2	diverse Arcade	diverse Arcade
SheepShaver	GPL	Linux, MacOS-X, BSD Windows, BeOS	PowerPC, 680x0	MacOS 7.5.2 bis MacOS 9.0.4
Hercules 3.07	QPL	Linux, MacOS-X, BSD Solaris, Windows	IBM-Großrechner	IBM System/360, 370, 390

GPL = GNU **General Public License**

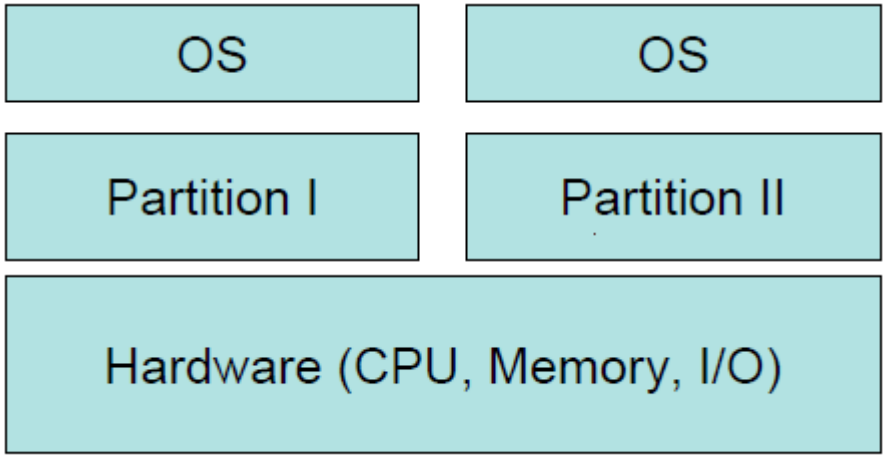
Beispiel: QEMU (Wikipedia)

- QEMU (von englisch „Quick Emulator“) ist eine **freie virtuelle Maschine**, die die gesamte Hardware eines Computers emuliert und durch die dynamische Übersetzung der Prozessorinstruktionen des Gastprozessors in Instruktionen für den Host-Prozessor eine sehr gute Ausführungsgeschwindigkeit erreicht.
- QEMU emuliert Systeme mit den folgenden Prozessorarchitekturen: x86, AMD64 und x86-64, PowerPC, ARM (32 + 64 Bit), Alpha, CRIS, LatticeMico32, m68k (Coldfire), MicroBlaze, MIPS, Moxie, SH-4, S/390, Sparc32/64, TriCore, OpenRISC, Unicore und Xtensa (Stand 2015).
- QEMU ist auf den Betriebssystemen GNU/Linux, Windows, FreeBSD, NetBSD, OpenBSD, OpenSolaris, OS/2/eComStation, DOS, Mac OS X und Haiku lauffähig. Es kann den gesamten Status einer virtuellen Maschine so speichern, dass diese auf ein anderes Host-System übertragen werden kann und dort weiterlaufen kann (Live-Migration).

Beispiel: KVM (Wikipedia)

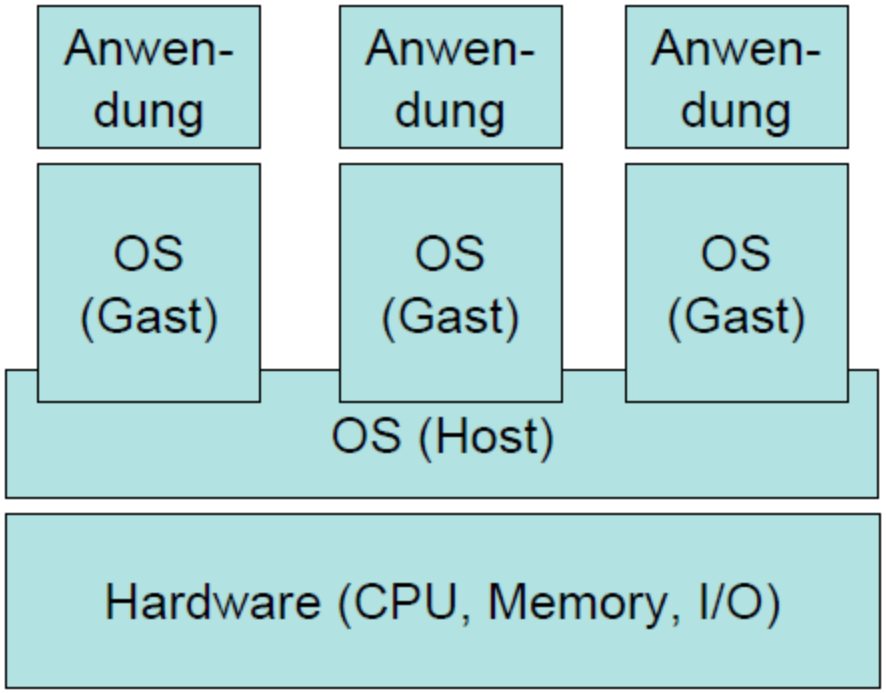
- Die **Kernel-based Virtual Machine** (**KVM**) ist eine Infrastruktur des Linux-Kernels zur Virtualisierung, die auf den Hardware-Virtualisierungstechniken von Intel (VT) oder AMD (AMD-V) ausgestatteten x86-Prozessoren sowie auf der System-z-Architektur lauffähig ist.
- KVM wurde im Oktober 2006 veröffentlicht und ist ab Version 2.6.20 des Linux-Kernels in diesem enthalten. Es wurde unter der Federführung von Avi Kivity bei dem israelischen Unternehmen *Qumranet* entwickelt. Qumranet wurde im September 2008 von Red Hat gekauft.
- KVM wurde zunächst für die x86-Plattform entwickelt und besteht für diese aus dem Kernel-Modul kvm.ko sowie aus den hardwarespezifischen Modulen kvm-intel.ko (für Intel-Prozessoren) oder kvm-amd.ko (für AMD-Prozessoren). Inzwischen gibt es KVM auch für weitere Plattformen wie PowerPC[5] und ARM.
- KVM selbst nimmt keine Emulation vor, sondern stellt nur die Infrastruktur dazu bereit; QEMU ist derzeit die einzige Möglichkeit, diese zu nutzen. Dazu stellt QEMU für virtualisierte Gastsysteme die notwendigen Geräte wie Festplatten, Netzwerk-, Sound- und Grafikkarten zur Verfügung.
- Nach dem Laden des Moduls arbeitet der Linux-Kernel selbst als Hypervisor für virtuelle Maschinen. Als Gastsysteme unterstützt KVM Linux (32 und 64 Bit), Windows (32 und 64 Bit), Haiku, AROS, ReactOS, FreeDOS, Solaris und diverse BSD-Derivate. KVM läuft auch auf SMP-Hostsystemen, SMP-Gastsysteme sind ebenfalls möglich.

Ansätze der Servervirtualisierung: Partitionierung



- In jeder Partition kann eine eigene Betriebssysteminstanz betrieben werden
- Aufteilung kann nur entlang von Modulgrenzen durchgeführt werden
- eine VM enthält mindestens eine CPU und ein Memory-Modul
- gute physikalische Trennung der einzelnen Systeme untereinander
- keine Performanceverluste z.B. durch Hypervisor
- keine Anpassung des Betriebs-systems notwendig

Betriebssystemvirtualisierung: Container



- Es können nur gleiche Betriebssysteme virtualisiert werden
- einige Teile des Betriebssystems werden gemeinsam genutzt
- Treiber werden vom Host-Kernel verwaltet
- Systemaufrufe werden abgefangen und vom Host ausgeführt.
- geringer Overhead (Virtualisierungsaufwand 1-3%)

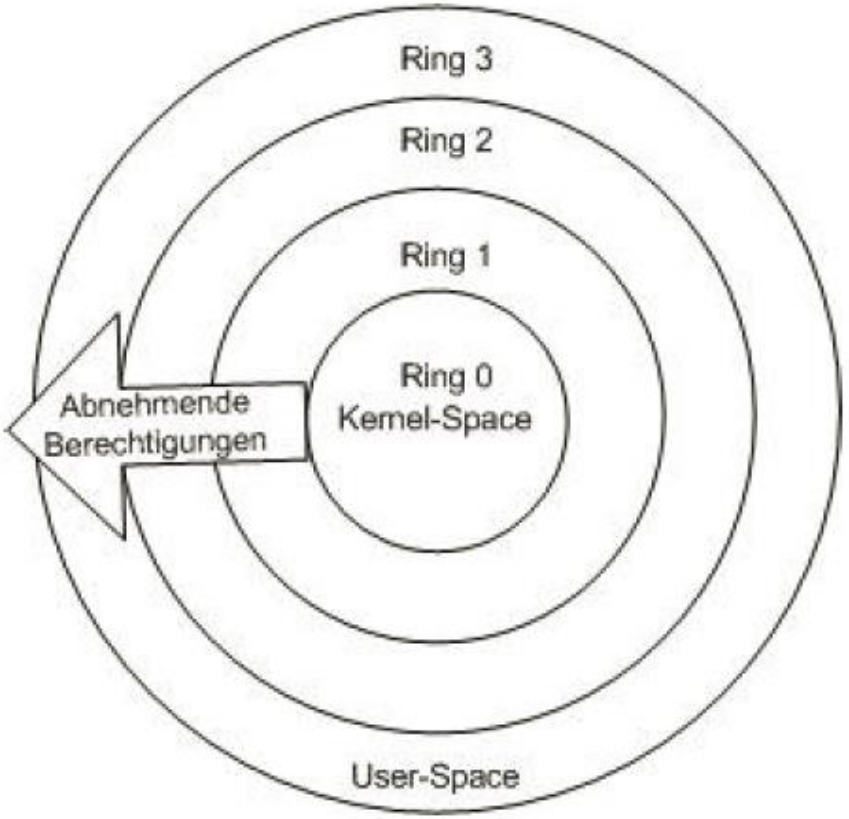
- Einsatz: z.T. Internet-Hosting

Containerimplementierungen

- OpenVZ
- Parallels Virtuozzo Containers
- FreeBSD jails
- Linux-Vserver
- Solaris 10 Containers/Zones
- IBM AIX6 WPARs (Workload Partitions)

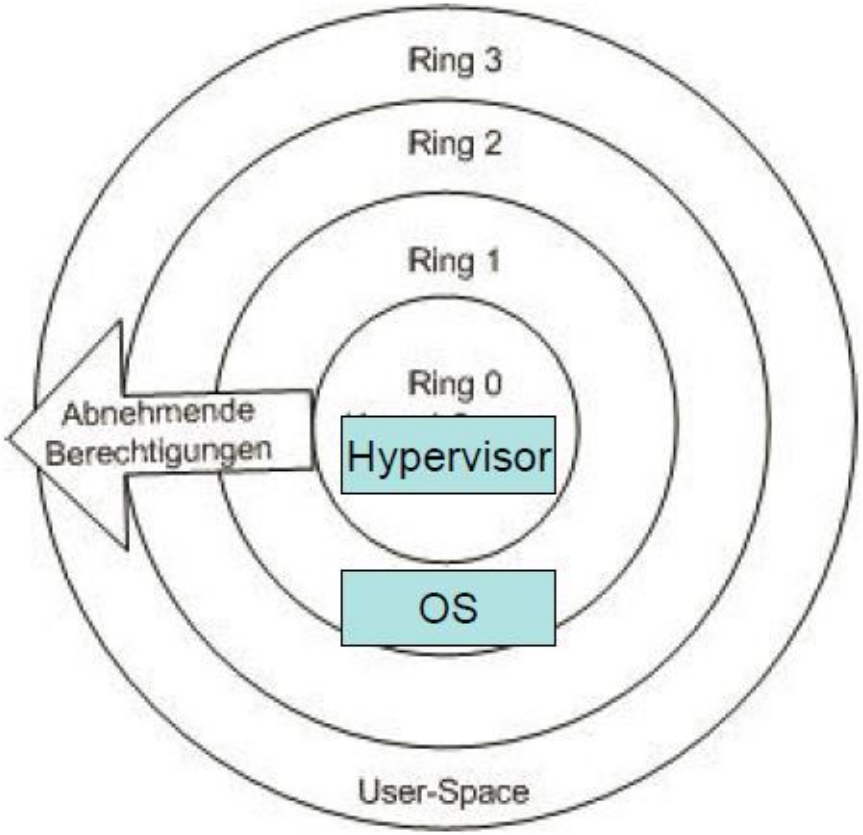
Hierarchisches Privilegiensystem der x86-Architektur (1/2)

- Regelung des Zugriffs auf Speicher und Befehlssatz des Prozessors
- Ring 0: direkter Zugriff auf Hardware und Speicherbereiche; Kernel des Betriebssystem
- Ring 1 und 2: diese werden normalerweise nicht genutzt
- Ring 3: Benutzerapplikation

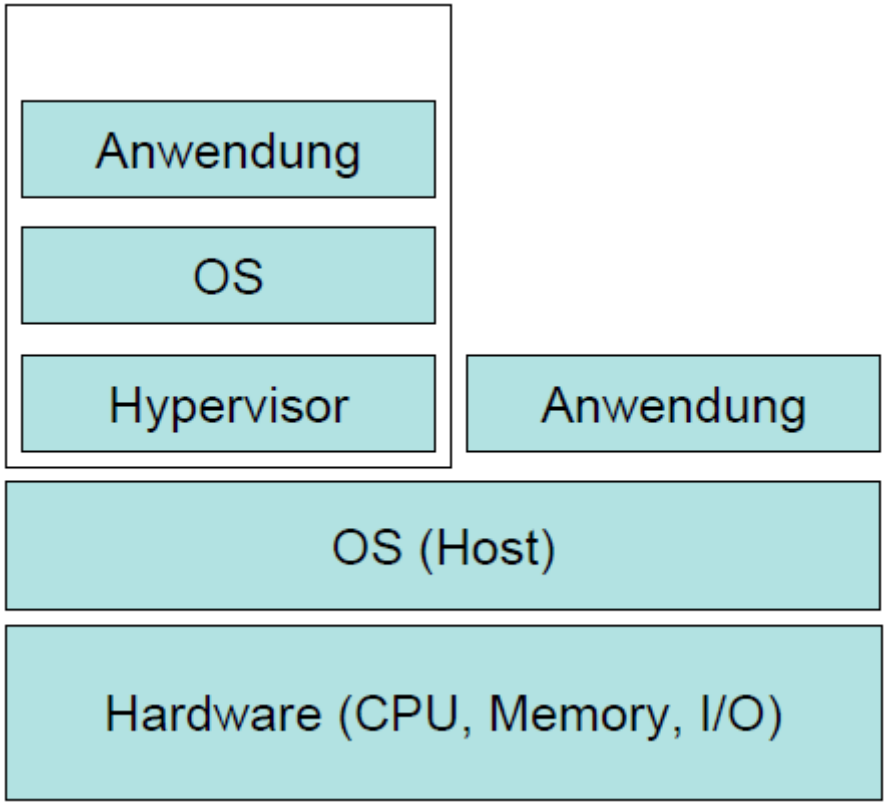


Hierarchisches Privilegiensystem der x86-Architektur (2/2)

- x86 Betriebssysteme sind gebaut um direkt auf der Hardware zu laufen, daher gehen sie davon aus, die Hardware vollständig und exklusiv zu besitzen.
- Hypervisor soll die Kontrolle über die Hardware übernehmen.
- Betriebssystem wird in einen äußeren Ring verschoben



Ansätze der Servervirtualisierung: Hosted Hypervisor (Type 2)



- Virtuelle Maschine simuliert realen Rechner mit allen Komponenten
- Basiert auf herkömmlichen Betriebssystemen

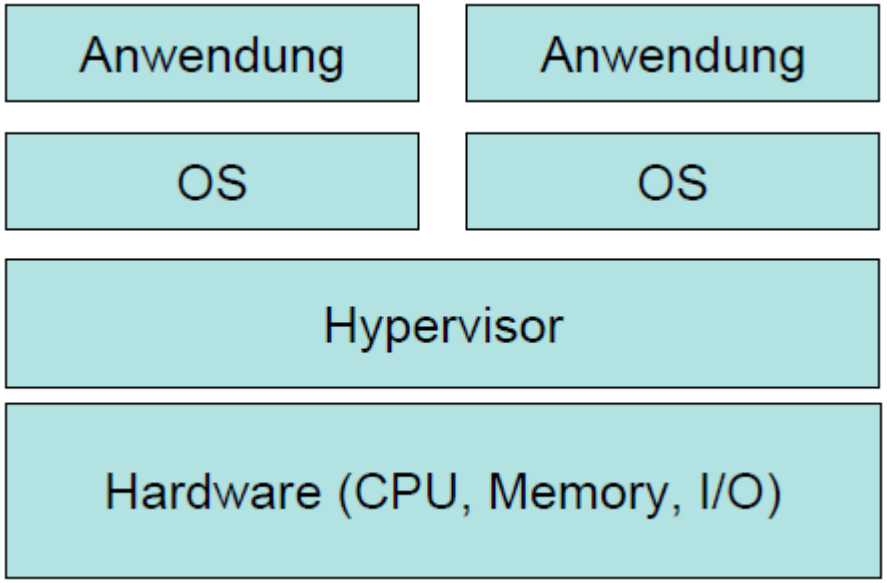
Vorteile

- Keine Änderung an Betriebssystemen notwendig
- Flexibilität

Nachteile

- „Schlechtere“ Performance

Ansätze der Servervirtualisierung: Vollständige Virtualisierung



- Virtuelle Maschine simuliert realen Rechner mit allen Komponenten
- Virtueller Maschinen-Monitor (VMM) koordiniert virtuelle Maschinen und Ressourcen

Vorteile

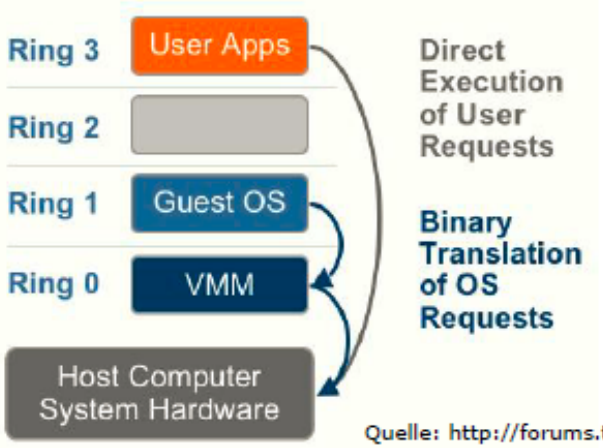
- Kaum Änderungen an Betriebssystemen
- Flexibilität

Nachteile

- Hypervisor/VMM teuer
- „Schlechtere“ Performance

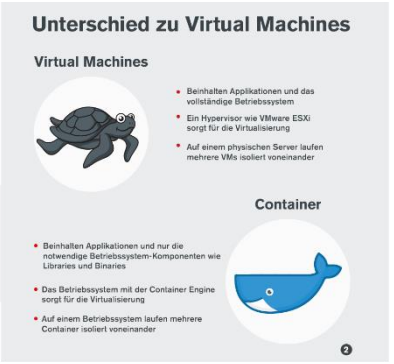
Vollständige Virtualisierung

- Nachbildung einer kompletten Hardwareumgebung für die virtuelle Maschine um die Zugriffe der Gastbetriebssysteme zu steuern
- Virtuelle Maschine simuliert realen Rechner mit allen Komponenten
- Virtueller Maschinenmonitor (VMM) koordiniert virtuelle Maschinen und Ressourcen
  - Jedes Gast-Betriebssystem hat einen eigenen virtuellen Rechner mit CPU, Hauptspeicher, Laufwerken, Netzwerkkarten, usw. zur Verfügung
- VMM und Hostbetriebssystem in Ring 0, Gastbetriebssysteme höher

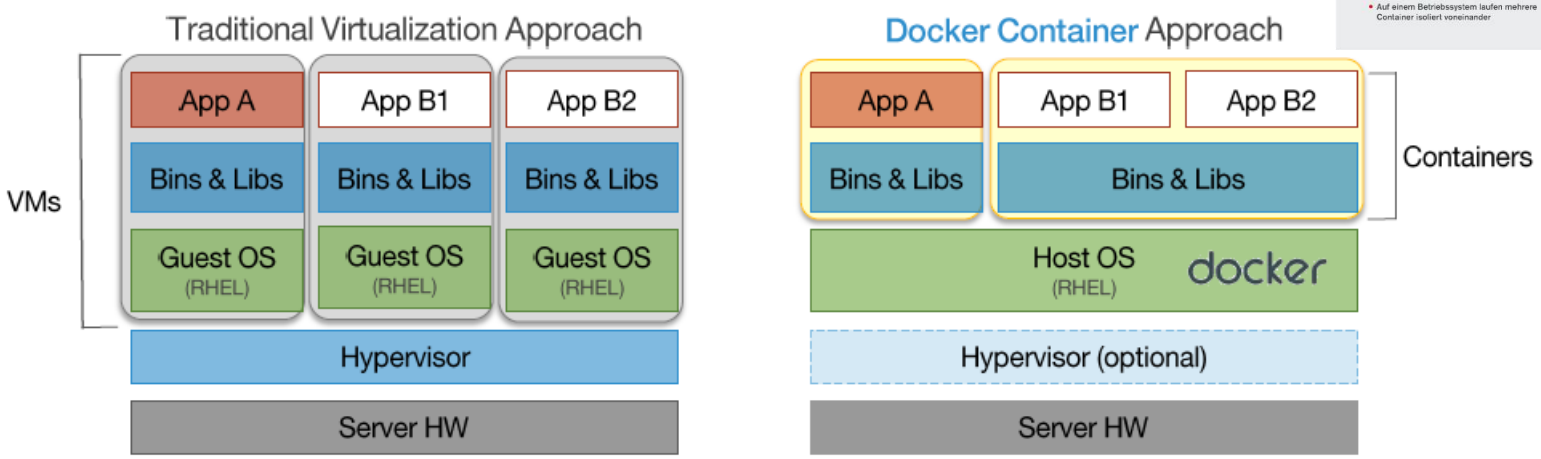


Quelle: http://forums.techarena.in/guides-tutorials/1104460.htm

Vergleich VM zu Container



Containers are a **lightweight** alternative to Virtual Machines for running software in **portable** and **isolated** virtual environments



Attribute	VM	Container
Start-up time & Performance	Slow (minutes) HV overhead	Fast (seconds) no HV overhead
Footprint	Large (nothing shared)	Small (OS kernel shared)
Resource Constraints	Yes	Yes (CPU, Memory)
Isolation & Security	High	High
Portability	Low	High

Aufgabe/Frage

- Wir haben gerade die Containerarchitektur kennengelernt.

- Welches sind die Vor- und Nachteile von Containerarchitekturen?

Virtualisierung: Vorteile

Bessere Ausnutzung der Hardware

- Server- und PC-Konsolidierung, Zusammenlegen vieler virtueller Server auf möglichst wenigen physikalischen Servern (auf aktuelle Serverblades passen bis zu 40 aktive Server-Instanzen)
- Bessere Energie-Effizienz, Kosten-Senkung bei Hardware, Stellplatz, Administration

Vereinfachte Administration

- Anzahl der physischen Server reduziert sich

Vereinfachte Bereitstellung

- Neue Infrastruktur und Server können sehr schnell manuell oder automatisch erzeugt werden

Erhöhung der Verfügbarkeit

- Migration von Servern im laufenden Betrieb
- Virtuelle Maschine (VM können leicht vervielfätigt und gesichert werden)
- Snapshots vom aktuellen Zustand

Höhere Sicherheit

- VM sind gegenüber anderen VMs und dem Host-System isoliert

Virtualisierung: Nachteile/Grenzen

- VM bieten eine geringere Performance als reale Maschine
- Nicht jeder Hardware kann aus einer Virtuelle Maschine emuliert werden
- Bei der Serverkonsolidierung können virtuelle Maschinen einen Single Point of Failure darstellen. Beim Ausfall eines Hosts würden mehrere virtuelle Server ausfallen
- Zu komplex. Zusätzliches Know-how ist notwendig

Virtualisierung

Hypervisor Comparison 2019: KVM vs Hyper-V vs XenServer vs vSphere

Feature	Windows Hyper-V 2019	vSphere 6.7	XenServer 7.6	KVM
RAM/Host	24TB	12 TB	5TB	12TB
RAM/VM	12 TB for generation 2;	6 TB	1.5TB	6 TB
1 TB for generation 1				
CPUs/VM	240 for generation 2;	128	32	240
64 for generation 1;				
VM Disk	64 TB for VHDX format;	62TB	2TB	10TB
2040 GB for VHD format				
VM Live Migration	Yes	Yes	Yes	Yes
VM Replication supports	Yes	Yes	Yes	Yes
Overcommit resources	No	Yes	No	Yes
Disk I/O Throttling	Yes	Yes	Yes	Yes
Hot plug of virtual resources	Yes	Yes	Yes	Yes

https://www.acte.in/citrix-xenserver-vs-vmware-vsphere-article

Beispiel: Datenspeichervirtualisierung

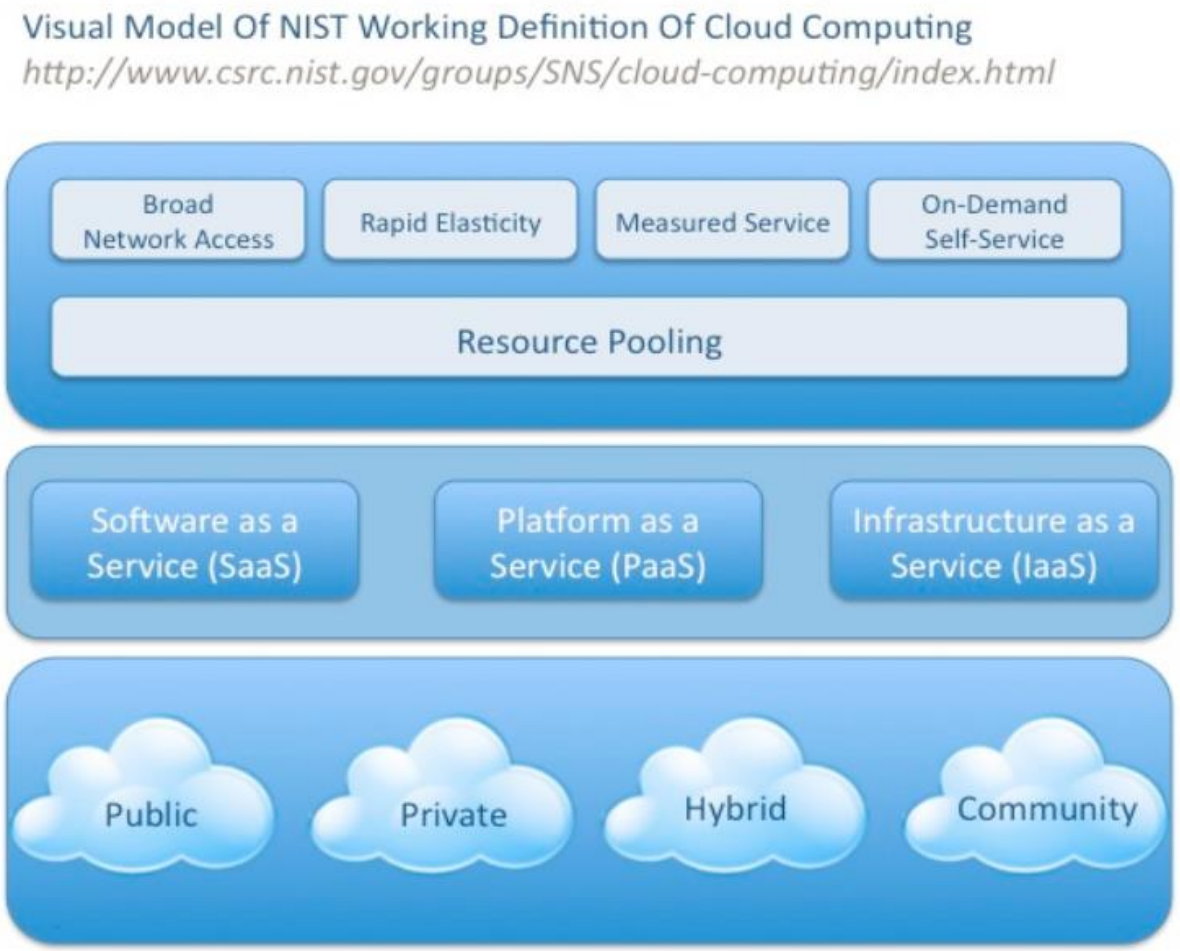
- **Idee**: Trenne Server und Speicher und verwende Speichernetz SAN (Storage Area Network)
- Wichtiges Technologie: Zuordnung des physikalischen Speichers zum virtuellen (Mapping)
- Speicher kann den Servern dynamisch zugeteilt werden
- Migration von Speichersubsystemen zur Laufzeit
- **Vorteile**
  - Optimale Auslastung und Konsolidierung

Beispiel: Netzwerkvirtualisierung

- **VPN**
  - Getrennte virtuelle Netzwerke über gemeinsame einheitliche virtuelle Infrastruktur
- **Kanten-Virtualisierung**
  - Mehrere unabhängige virtuelle Verbindungen werden über eine gemeinsame physikalische Verbindung (Kante) transportiert
- **Knoten-Virtualisierung**
  - Verteilung der Ressourcen auf die virtuellen Knoten
- **Vorteile**
  - Benutzerfreundlichkeit
  - Mehr Flexibilität und Personalisierbarkeit
  - Schnelleren und sicheren Zugriff auf Anwendungen und Daten

https://www.acte.in/citrix-xenserver-vs-vmware-vsphere-article

Cloud Computing: Ein kurzer Überblick



Essential Characteristics

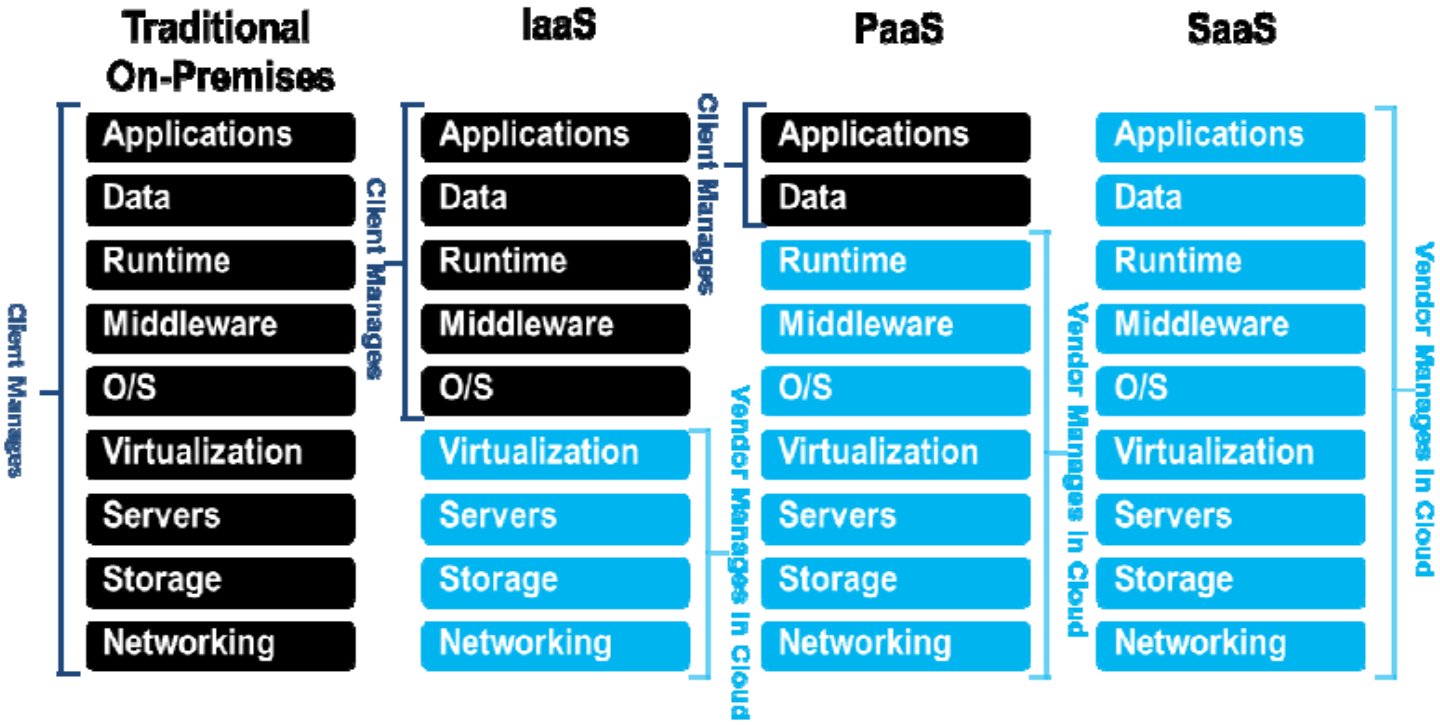
Service Models

Deployment Models

NIST: National Institute of Standards and Technology

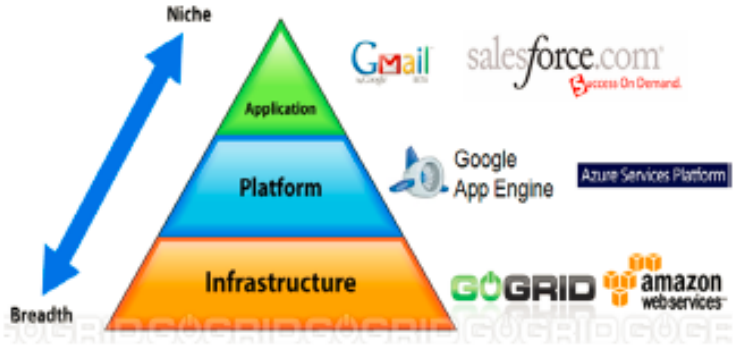
CSA: Cloud Security Alliance

Cloud Service Model



Cloud Computing: Nutzungsmodelle

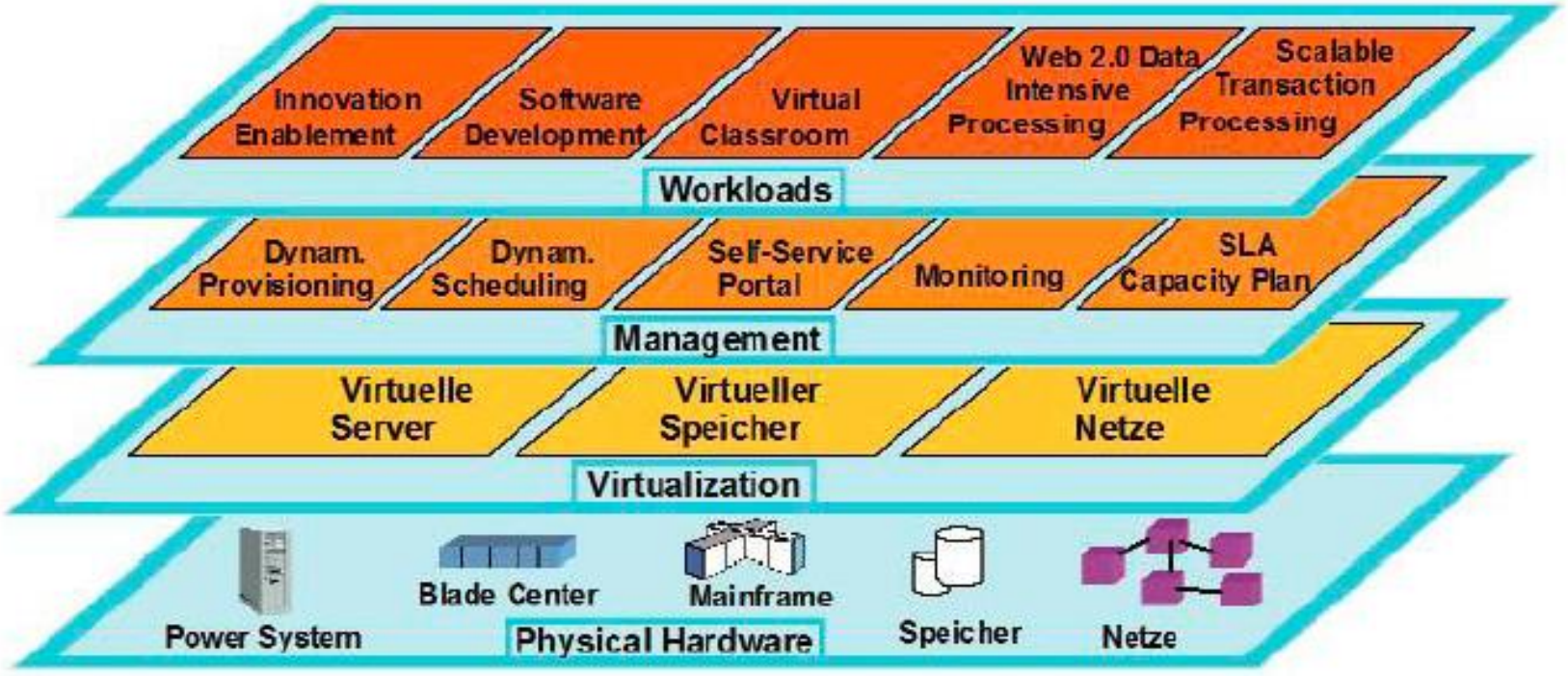
- **Software as a Service (SaaS)**
  - Die Bereitstellung der Funktionalität von Software via Internet
  - Webbasierte Zugriff auf Anwendungen (Endbenutzer)
  - Kosten fallen nur in Bedarfsfall an
  - immer aktuelle Version



- **Platform as a Service (PaaS)**
  - Plattformbenutzung (Entwickler) über Internet
  - Von Kunden erstellte Anwendungen, die im Internet verteilt werden

- **Infrastructure as a Service (IaaS)**
  - Infrastrukturbenutzung: Das Mieten von Ressourcen für Rechenleistung, Speicher, Netzwerk, und andere Aufgaben
  - Kosten fallen nur in Bedarfsfall an
  - Skalierbarkeit

Cloud Computing: Architektur



Highlevel-Architektur von Cloud Computing nach Vorstellung von IBM.

Cloud Computing: Charakteristika

- Nutzer werden Mieter
- Nutzer besitzen keine physikalische Infrastruktur
- Nutzer zahlen nur Dienste, die sie benutzt haben (Utility Computing)
- Zusätzliche Ressourcen (Speicher, Rechnerleistung, Anwendungen etc.) sind immer zur Verfügung, nur bei Bedarf
- Benutzer haben Anspruch auf Erbringung durch Quality of Service oder Service Level Agreements

Cloud Anwendungen

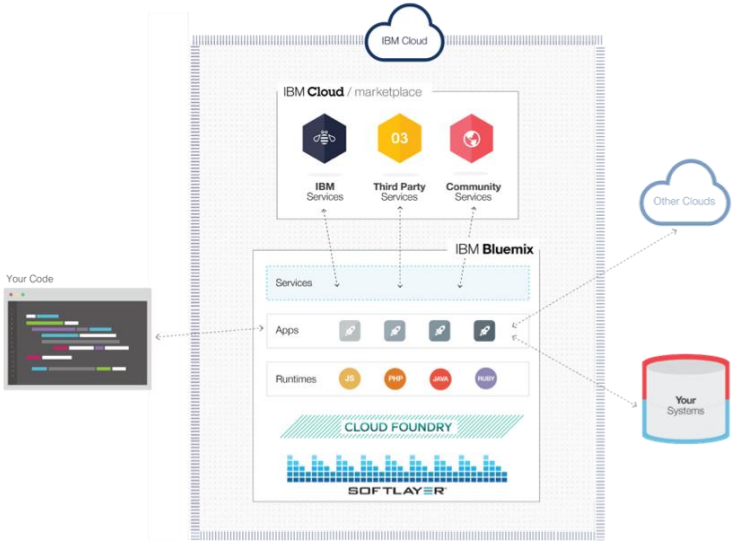
- Amazon Web Services

- Google Apps

- Microsoft Azure

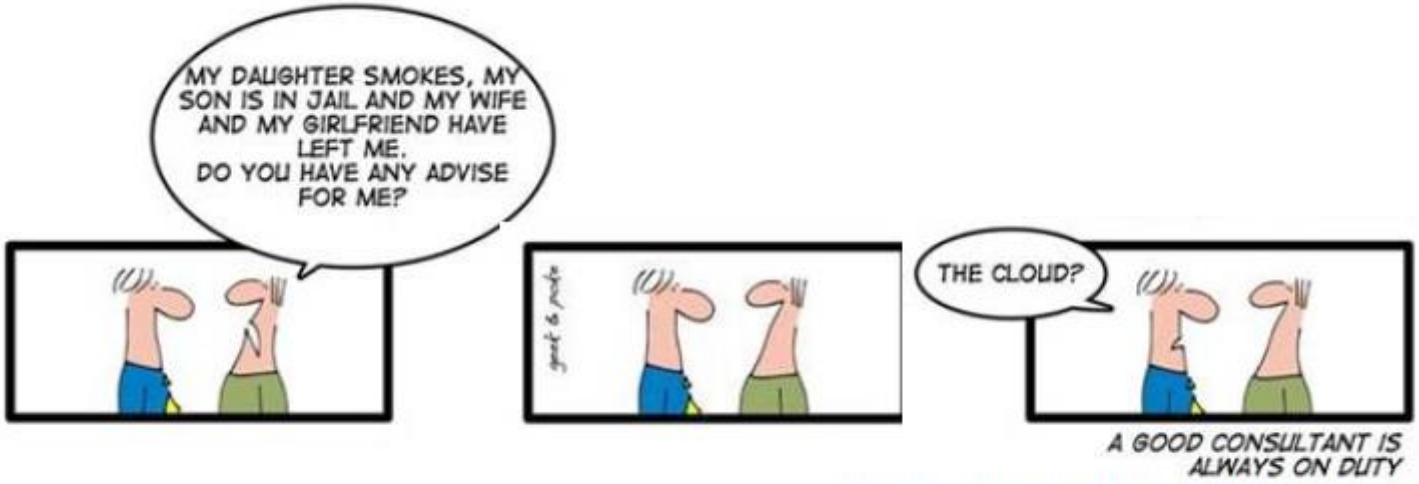
Quelle: http://www.microsoft.com/germany/net/WindowsAzure/

- IBM Cloud



- Alibaba

Virtualisierung



Quelle: geek and poke , http://geekandpoke.typepad.com