Das folgende Kapitel beschäftigt sich mit dem strukturellen Aufbau einer virtuellen Infrastruktur. Wir gehen den Fragen nach, welche Elemente dazugehören, wie sie ineinandergreifen und wie sie aufgebaut sind.

2 vSphere-Architektur

Autor dieses Kapitels ist Bertram Wöhrmann, Ligarion, bwoehrmann@ligarion.de

Mit der Einführung der VMware Virtual Infrastructure in der Version 3.x hat VMware einen Strategiewechsel vollzogen. Zusätzlich zu der eigentlichen Virtualisierungsplattform, dem ESX Server, und dem dazugehörigen Management hat VMware begonnen, zusätzlich Produkte um die Virtualisierung zu entwickeln. Es fing an mit der Service-Console-losen Version ESXi und dem Update Manager, VMware HA, VMware DRS und der Plug-in-Schnittstelle. Alles integriert sich nahtlos in das zentrale Management. Mit vSphere 4.0 ist dieser Ansatz konsequent weiterverfolgt worden. Auch der Lizenz-Server ist, nach einer kurzen externen Stippvisite, wieder integraler Bestandteil des vCenter-Servers.

Viele Themen reißen wir in diesem Kapitel nur kurz an. Wir sind der Meinung, dass es sinnvoller ist, die ausführlichen Erklärungen direkt in dem entsprechenden Abschnitt zu geben, in dem wir auch die passende Komponente beschreiben.

2.1 Bestandteile der virtuellen Infrastruktur

Die vSphere-Infrastruktur bietet alle Komponenten zum Virtualisieren von Betriebssystemen. Sie umfasst die Verwaltung sowie das Management der Virtualisierungsserver (vSphere-Server). Diese Grundfunktionen hat VMware noch weiter optimiert, um eine Ausfallsicherheit der Virtualisierungsserver zu erreichen (HA) und eine bestmögliche Lastverteilung zwischen den Virtualisierungsservern (DRS) zu ermöglichen. Des Weiteren werden die virtualisierten Systeme optimal mit den Server-Ressourcen (CPU, RAM etc.) versorgt. Damit Sie einen grundlegenden Überblick über die Grundbestandteile der Infrastruktur erhalten, erläutern wir diese nachfolgend.

2.2 vSphere-Host

Der physische Server, der seine Ressourcen wie CPU, Hauptspeicher (RAM), Netzwerkkarten und Festplattenspeicher über eine Virtualisierungsschicht (Hypervisor) den virtuellen Maschinen zur Verfügung stellt, ist der vSphere-Host. Viele werden den Vorgängernamen des vSphere-Hosts noch im Kopf haben, nämlich die Bezeichnung ESX-Host (Abbildung 2.1).

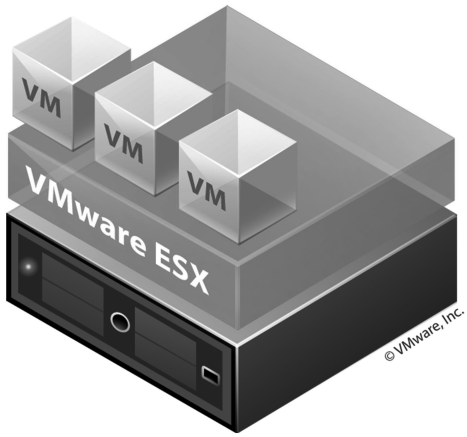


Abbildung 2.1 vSphere-Host

2.2.1 Hardware

Als Prozessorbasis für den Einsatz von VMware vSphere kommen nur 64-Bit-x86-Prozessoren zum Einsatz. Auf anderen CPUs ist das System nicht lauffähig, da sowohl VMkernel als auch Service Console einen 64-Bit-Kernel besitzen. Für die Installation benötigen Sie mindestens eine CPU und Minimum 2 GB Arbeitsspeicher. Des Weiteren ist ein unterstützter Storage-Kontroller nötig. Möglich sind IDE (lokal), SCSI, SAS, SATA und Fibre-Channel. Abschließend wird noch mindestens eine Netzwerkkarte benötigt, damit Sie auf das System zugreifen können. Im Normalfall werden aber wohl mehr Netzwerkports zum Einsatz kommen.

Intel® VT-x/AMD-V™

Alle Prozessoren, für die VMware Support anbietet, müssen eine Erweiterung zur Unterstützung von Virtualisierungstechnologien aufweisen. Durch diese Technologien wird im Wesentlichen der Virtual Machine Monitor (VMM) in seiner Arbeit unterstützt,ermöglicht dadurch wird der Overhead reduziert. Auch der Prozess der Migration einer aktiven virtuellen Maschine zwischen verschiedenen Prozessorgenerationen wird erleichtert.

Die unterstützten CPUs von beiden Herstellern bringen eine solche Technologie mit.

Die folgenden Prozessoren erhalten Support von VMware:

- AMD Opteron (mindestens Barcelona für Fault-Tolerance)
- Intel Xeon 3000/3200, 3100/3300, 5100/5300, 5200/5400, 7100/7300, 7200/7400
- Intel Nehalem

2.2.2 HCL

Wie andere Betriebssystemhersteller bzw. Hersteller von Hypervisoren pflegt die Firma VMware eine Hardware Compatibility List (HCL). Vergewissern Sie sich, dass Ihre Komponenten, die Sie einsetzen wollen, in dieser Liste aufgeführt sind. Sie müssen zwar keine Bedenken haben, dass ein nicht gelistetes System nicht funktionieren kann, aber Sie haben nur Support für Ihre Virtualisierungslandschaft, wenn Sie sich aus der Liste der unterstützten Hardware bedienen.

Die HCL finden Sie unter: *http://www.vmware.com/resources/compatibility/search.php*

2.2.3 Maximale Ausstattung eines ESX-Hosts

Auch wenn Sie sich an die HCL halten, so müssen Sie doch wissen, welche Ausstattung des Hosts mengenmäßig noch unterstützt wird. Diesen Punkt möchten wir in den folgenden Tabellen näher aufschlüsseln.

CPUs pro Host	Anzahl	Bemerkungen
logische Prozessoren	64	Die Anzahl berechnet sich aus: Sockel × Cores × Threads
virtuelle CPUs (vCPUs)	512	–
maximale Anzahl von virtuellen CPUs, pro Core	25	Die Anzahl ist abhängig von der Last, die die VMs verursachen. (Gilt für vSphere 4.0 Update 1, bei vSphere 4.0 sind es 20 vCPUs.)
maximale Anzahl von VMs pro Host	320	Die Anzahl ist abhängig von der Last, die die VMs verursachen.

Tabelle 2.1 Maximale CPU-Werte

Memory pro Host	Menge
Arbeitsspeicher	1 TB
Arbeitsspeicher Service Console	800 MB

Tabelle 2.2 Maximale Memory-Werte

Anhand der Menge der Karten geben wir in den Tabellen für die Netzwerkkarten nur den Treiber und den Chiphersteller an.

Netzwerkkarte	Chiphersteller	Speed	Anzahl	Bemerkungen
Physische Karten				
e1000	Intel	1 GBit	32	Gilt sowohl für PCI-X als auch für PCIe.
igb bnx2	Intel Broadcom	1 GBit	16	–
tg3	Broadcom	1 GBit	32	–
forcedeth	Nvidia	1 GBit	2	–
s2io nx_nic	Neterion NetXen	10 GBit	4	–
ixgbe bnx2x	Intel Broadcom	10 GBit	4	–

Tabelle 2.3 Maximale Anzahl physischer Netzwerkkarten

PCI VMDirectPath	Anzahl
PCI VMDirectPath Devices	8
vNetzwerk-Standard-Switch	**Anzahl**
virtuelle Switch-Ports gesamt	4.096
virtuelle Switch-Ports pro vSwitch	4.088
Portgruppen pro vSwitch	512
vSwitches	248
vNetzwerk Distributed Switch	**Anzahl**
virtuelle Switch-Ports gesamt	4.096
Hosts pro Switch	64

Tabelle 2.4 Maximale Anzahl virtueller Karten/Ports

Für den Storage gibt es ebenfalls Einschränkungen. Die Tabelle 2.5 trennt die verschiedenen Anbindungsmöglichkeiten voneinander und beschreibt ebenfalls das File-System VMFS.

VMFS allgemein	Maximalwert
RDM-Größe	2 TB
Volume-Größe	64 TB
VMs pro Volume	256
Extents pro Volume	32
VMFS-3	**Maximalwert**
Volumes per Hosts	256
Files pro Volume	~30.270
File-Größe	2 TB – 512 MB
Blockgröße	8 MB
Fibre-Channel	**Maximalwert**
LUNs pro Host	256
Anzahl Pfade pro LUN	32
maximale Anzahl pro Host	1.024
HBAs pro Host	8
maximale Anzahl von HBA-Ports	16
NFS	**Maximalwert**
Standardanzahl von NFS-Datastores	8
maximale Anzahl von NFS-Datastores	64
Hardware-iSCSI-Initiators	**Maximalwert**
LUNs pro Host	256
Initiator-Ports pro Host	4
Pfade pro Host	1.024
Pfade pro LUN	8
dynamische Targets pro Port	64
statische Targets pro Port	61

Tabelle 2.5 Maximalwerte im Storage-Umfeld

Software-iSCSI-Initiators	Maximalwert
LUNs pro Host	256
Initiator-Ports pro Host	8
Ziele pro Host	256
Pfade pro LUN	8
Pfade pro Host	1.024

Tabelle 2.5 Maximalwerte im Storage-Umfeld (Forts.)

2.2.4 Version ESX vs. ESXi

VMware stellt zwei Varianten seiner ESX-Server-Software zur Verfügung, den ESX Server 4 und den ESX Server 4i (als *embedded* oder als *installable*). Viele reduzieren den Unterschied zwischen den beiden Versionen auf die Service Console, dabei gibt es aber weit mehr Unterschiede. Das beginnt schon bei der Installation der Systeme. Beide Versionen lassen sich auf einer Festplatte installieren. Einen Boot von einer SAN-LUN unterstützt die Embedded-Version nicht. Im Unterschied dazu können Sie die ESXi-Version auch direkt von einem USB-Stick oder einer SD-Karte booten. Eine skriptbasierte Installation z. B. über Kickstart-Mechanismen ist für die ESXi-Variante nicht möglich.

Eine ganze Reihe von Unterschieden ergibt sich durch die Möglichkeit der Installation von Tools in der Service Console. So fehlt dem ESXi die Anbindung an ein Active Directory. Eine zentrale Pflege von Benutzern an der Konsole ist somit nicht möglich. Das Patchen bei der abgespeckten Version ist wie ein Firmwareupdate auf einer Hardwarekomponente zu sehen. Eine Anmeldung über ein Webinterface ist auch nicht möglich. Achten Sie bitte darauf, dass Sie manche Funktionen im ESXi nur nutzen können, wenn Sie nicht die Basisversion einsetzen, sondern die erweiterten Lizenzpakete verwenden. Darunter fallen die Nutzung von SNMP und die lesende und schreibende Möglichkeit über das vCLI. Beim vCLI handelt es sich um das vSphere Command Line Interface. Das Interface wird auf einem im Netzwerk befindlichen Computer installiert und ermöglicht das Absetzen von Konfigurationsbefehlen gegen das vCenter beziehungsweise den vSphere Host. Die Software kann sowohl unter Windows als auch unter Linux installiert werden.

Analog dazu verhält es sich mit dem PowerCLI und dem vSphere SDK. Alle Kommandozeilentools verwenden dieselbe API, aus diesem Grund verhalten sich alle drei Tools in diesem Punkt identisch. Der Konfigurationsbefehlssatz zwischen der Service Console und dem vCLI ist leider nicht identisch. Nicht alle Befehle sind schon in der API implementiert. Daraus resultiert ein signifikanter Unter-

schied in den Optionen der Konfiguration über die Befehlszeile. Der Anschluss einer seriellen Konsole ist nur beim ESX Server möglich, ein Zugriff auf den ESXi-Host über diesen Weg ist nicht aktivierbar.

Es bleibt abzuwarten, wie sich die Unterschiede zwischen den beiden Versionen in Zukunft entwickeln. Es ist ja immer wieder zu lesen, dass VMware in der ESXi-Variante die Zukunft sieht. Aus unserer Sicht fehlen aber noch einige wichtige Optionen, die von der Full Version zur ESXi-Version portiert werden müssen.

2.3 vCenter-Server

Der vCenter-Server ist Dreh- und Angelpunkt der VMware Infrastruktur. Mit ihm verwalten Sie das komplette VMware-Datacenter von der Konfiguration der ESX-Server über das Erstellen von virtuellen Maschinen bis zum Einrichten der VMware-Features HA (High Availability) und DRS (Distributed Ressource Scheduling) sowie viele andere Funktionen. Des Weiteren bietet das vCenter eine Zugriffskontrolle auf Basis von Benutzern und Gruppen. Performance-Daten der vSphere-Server sowie der virtuellen Maschinen werden ebenfalls gesammelt und in der Datenbank abgelegt.

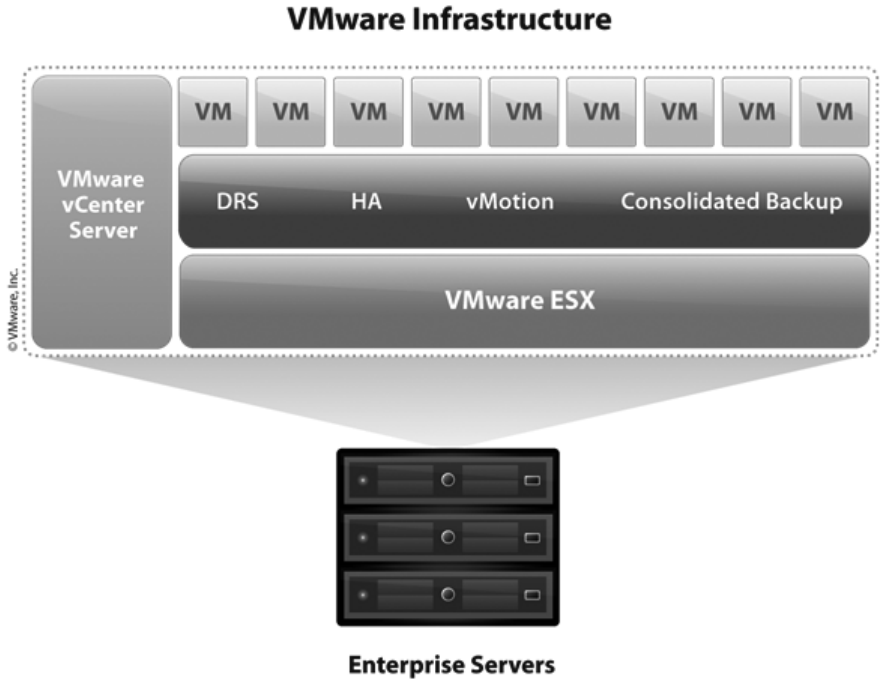


Abbildung 2.2 Struktur des vCenter-Servers

Der vCenter-Server ist nicht zwingend notwendig zum Betreiben von vSphere-ESX-Hosts, damit diese die Dienste bereitstellen können, um virtuelle Maschinen zu erstellen. Sie können jeden Host einzeln und unabhängig voneinander verwalten. Einige Dienste aber, wie z. B. DRS, setzen zwingend einen vCenter-Server voraus.

Die Software bietet eine zentralisierte Verwaltung aller im VMware-Datacenter zusammengefassten Ressourcen, deren virtueller Maschinen sowie der Benutzer.

Die zentralen Dienste des vCenters sind:

- Provisionierung von virtuellen Maschinen
- Konfiguration von vSphere-Servern (Hosts) und virtuellen Maschinen (VMs)
- Konfiguration von Resource-Pools
- Inventarisierung von vSphere-Ressourcen und virtuellen Maschinen
- Konsolenzugang zu den virtuellen Maschinen
- Statistiken und Protokolle zur Ressourcenauslastung
- Alarm- und Event-Management, um VI-Administratoren über Events und erhöhter Auslastung zu informieren
- rollenbasiertes Rechtemodell, um Objektgruppen zu verwalten
- Task-Scheduler, um bestimmte Aktivitäten zu bestimmten Zeiten automatisiert auszuführen

Diese zentralen Dienste des vCenters können um bestimmte Features erweitert werden. Dabei dient das vCenter als zentrale Schnittstelle, um die zusätzlichen Dienste, *Distributed Services* genannt, zu verwalten. Die erweiterten Dienste sind:

- VMware DRS
- VMware HA
- VMware Fault-Tolerance
- VMware VMotion
- VMware Storage VMotion

Des Weiteren bieten der Management-Server sowie der vSphere-Client eine Schnittstelle für Plug-ins zur Erweiterung der Funktionalität:

- vCenter Converter
- vCenter Update Manager
- vCenter Orchestrator
- Tools von Third–Party-Vendors oder auch Freewaretools (z. B. icomasofts PowerScripter)

Editionen

Es gibt zwei Versionen des vCenter Servers von VMware:

- vCenter Server for Essentials (maximal 20 Hosts)
- vCenter Server Standard

Beim vCenter Server for Essentials handelt es sich um eine vCenter-Version für kleine Installationen mit maximal 20 Hosts und abgespecktem Funktionsumfang.

Diese Version unterstützt nur die folgenden Funktionen:

- Thin Provisioning
- vCenter Agent
- vCenter Update Manager
- VMSafe
- vStorage-APIs
- VMware HA
- VMware Data Recovery

Wollen Sie weitergehende Funktionen nutzen, kommt nur der Einsatz der Standard-Version in Frage. Die Version vCenter Server Standard unterstützt zusätzlich zu den oben gelisteten Funktionen:

- Hot Add
- Fault-Tolerance
- vShield Zones
- VMotion
- Storage VMotion
- DRS
- vNetwork Distributed Switch
- Host-Profiles
- Third-Party Multipathing

Maximale Ausstattung

Auch das vCenter kann nicht unendlich viele Ressourcen verwalten; es gilt die eine oder andere Einschränkung. Die Angaben beziehen sich auf die Standard-Version des vCenter Servers.

vCenter Server Scalability – 32-Bit-OS-Server	Anzahl
Hosts	200
powered-on VMs	2 000
registered VMs	3 000
concurrent vSphere-Client-Verbindungen	15

Tabelle 2.6 Mögliche verwaltbare Infrastruktur für das vCenter mit 32-Bit-OS-Server

vCenter Server Scalability – 64-Bit-OS-Server	Anzahl
Hosts	300
powered-on VMs	3 000
registered VMs	4 500
concurrent vSphere-Client-Verbindungen	30

Tabelle 2.7 Mögliche verwaltbare Infrastruktur für das vCenter mit 64-Bit-OS-Server

vCenter Server Scalability – vCenter Linked Node	Anzahl
Linked vCenter Server	10
Hosts im *Linked Mode*	1 000
powered-on VMs	10 000
registered VMs	15 000

Tabelle 2.8 Mögliche verwaltbare Infrastruktur für das vCenter im *Linked Mode*

vCenter Server Scalability – allgemein	Anzahl
Hosts pro Datacenter	100
Provisioning pro Host	8
Provisioning pro Datastore	8
VMotion pro Host	2
VMotion pro Datastore	4
Storage VMotion pro Host	2
Storage VMotion pro Datastore	4
gleichzeitige Operationen im vCenter	96

Tabelle 2.9 vCenter – allgemeine Einschränkungen

Die Zahlen, die VMware hier ansetzt, sind an der einen oder anderen Stelle sicherlich sehr optimistisch gewählt. Sie sollen nur als Richtschnur dienen, damit Sie abschätzen können, wie viele Managementsysteme Sie benötigen.

Zugriff

Damit Sie die virtuelle Infrastruktur auch verwalten können, benötigen Sie ein Werkzeug, um auf die einzelnen Komponenten zuzugreifen. Hier gibt es zwei Möglichkeiten: Eine der Optionen ist die Nutzung eines Webbrowsers. Seien Sie sich dabei im Klaren, dass Sie nicht alle Funktionen mit dem Webclient durchführen können. Außerdem ist der Webzugriff von Haus aus deaktiviert. Die zweite Möglichkeit ist die Nutzung des vSphere-Clients. Mit diesem Client können Sie direkt einen Host administrieren oder sich damit am vCenter-Server anmelden.

Lizenzierung

Für die Verwaltung der VMware-Lizenzen in einer vSphere-Infrastruktur wird der im vCenter integrierte Lizenz-Server eingesetzt. In dieser Verwaltungskonsole tragen Sie alle VMware-Lizenzen ein und weisen sie den zugehörigen Hardwarekomponenten zu. Dieses gilt aber nicht für Hosts der Vorgängerversion, wie Sie im folgenden Absatz nachlesen können.

License Server 3

Soll ein vCenter-Server der aktuellen Version (4.x) auch ESX 3.x-Hosts verwalten, so müssen Sie nach wie vor eine zusätzliche Komponente installieren. Es handelt sich um den alten Lizenz-Server, der schon in der Virtual Infrastructure 3 zum Einsatz kam. Dieser Lizenz-Server wird im Netzwerk bereitgestellt und in den vCenter-Eigenschaften konfiguriert.

VMware Infrastructure SDK

Es gibt die verschiedensten Möglichkeiten, in der virtuellen Infrastruktur Aufgaben zu automatisieren, auch außerhalb des vCenters und dessen Komponenten. Sie haben als Anwender verschiedene Optionen zur Verfügung, um eigene Anforderungen abzubilden. Dabei ist es egal, ob Sie mit der PowerShell skripten oder mit C programmieren möchten. Es gibt noch viele andere Möglichkeiten, das Management der Infrastruktur zu optimieren. Abbildung 2.3 zeigt nur eine Auswahl von Softwarepaketen, die VMware zur Verfügung stellt, damit Anwender das Management an ihre Bedürfnisse anpassen können.

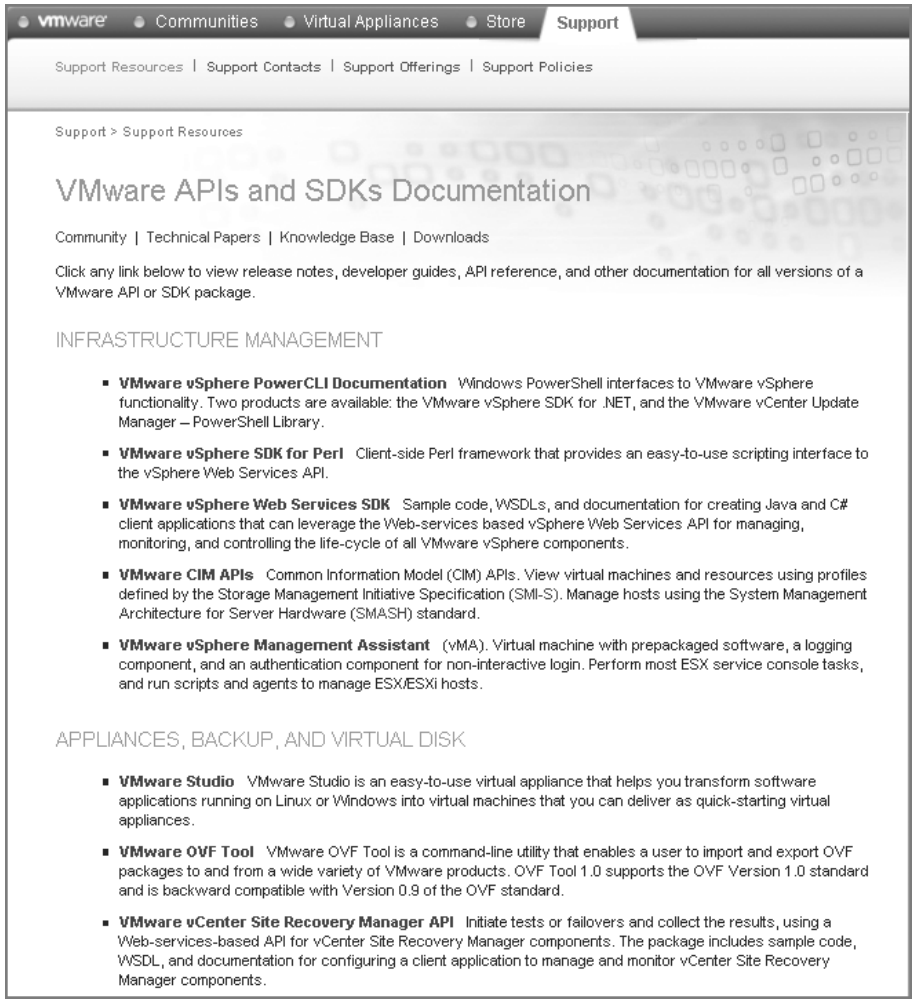


Abbildung 2.3 Webseite mit Informationen zu APIs und SDKs

Wenn Sie sich für alle Informationen interessieren, finden Sie auf der eigens dafür eingerichteten Webseite Näheres zu dem gewünschten Thema. Die Webseite erreichen Sie unter *http://www.vmware.com/support/pubs/sdk_pubs.html*, wo Sie auch die Applikationen herunterladen können.

Benötigte Netzwerkports – Implementierung

Das vCenter kommuniziert über das Netzwerk mit den zu verwaltenden Komponenten. Die Verbindungen werden über einen Windows-Dienst (*vpxd.exe*) hergestellt. Die folgenden Ports werden für die Kommunikation benötigt.

vCenter-Server		
Port	**Protokoll**	**Beschreibung**
80	HTTP	Dieser Port wird für den direkten Webzugriff benötigt. Es erfolgt aber nur eine Umleitung auf Port 443.
389	TCP	Für die Kommunikation mit dem Active Directory wird dieser Port benötigt.
443	HTTPS	Port für die initiale Anmeldung über der vSphere-Client
636		SSL-Verbindung zwischen den vCenter-Servern beim *Linked Mode*
902	UDP	Kommunikation zwischen vCenter und vSphere-Hosts
902/903	TCP	Verbindung zwischen vCenter und vSphere-Client
1433	TCP	Verbindung zum Datenbankserver MS SQL
1521	TCP	Verbindung zum Datenbankserver Oracle
8080	HTTP	Web Services HTTP
8443	HTTPS	Web Services HTTPS
27000/ 27010	TCP	Lizenz-Server für VI 3.x-Umgebungen

Tabelle 2.10 vCenter-Kommunikationsports

Update Manager		
Port	**Protokoll**	**Beschreibung**
9084	TCP	Webserver Update Manager
8084	TCP	SOAP-Server Update Manager
80/443	TCP	Download von Patches aus dem Internet von den Websites: *www.vmware.com* , *www.shavlik.com*

Tabelle 2.11 Update-Manager-Kommunikationsports

Beim vCenter Converter werden unterschiedliche Ports genutzt. Dies hängt unter Umständen sogar davon ab, welches Betriebssystem importiert werden soll. In Tabelle 2.12 finden Sie die Ports für die Übernahme eines eingeschalteten Windows-Servers.

vCenter Converter (P2V Windows-Server eingeschaltet)			
Ports	**Quelle**	**Ziel**	**Bemerkungen**
445	Converter-Server	Quelle	Nutzt die Quelle NetBIOS, dann wird dieser Port nicht benötigt.
137/138 UDP	Converter-Server	Quelle	Kommt NetBIOS nicht zum Einsatz, werden diese Ports nicht benötigt.
139	Converter-Server	Quelle	
9089	Converter-Server	Quelle	–
443	Converter-Server	vCenter-Server	Dieser Port wird genutzt, wenn das Ziel ein vCenter-Server ist.
443	Converter-Client	Converter-Server	Liegen der Client und der Converter auf unterschiedlichen Maschinen, dann wird dieser Port genutzt.
443/902	Quelle	vSphere-Host	Ist der vCenter-Server das Ziel, wird nur Port 902 benötigt.

Tabelle 2.12 vCenter-Converter-Kommunikationsports Windows online

Betrachten wir nun die benötigten Ports für die Übernahme eines aktiven Linux-Servers.

vCenter Converter (P2V Linux-Server eingeschaltet)			
Ports	**Quelle**	**Ziel**	**Bemerkungen**
22	Converter-Server	Quelle	Der Converter-Server muss eine Verbindung per SSH zur Quelle aufbauen können.
443	Converter-Client	Converter-Server	Ist nur relevant bei einer angepassten Installation und wenn Server und Client nicht auf einem Server liegen.
443	Converter-Server	vCenter-Server	Wird nur benötigt, wenn das Ziel ein vCenter-Server ist.
443/902/903	Converter-Server	vSphere-Hosts	Ist das Ziel ein vCenter-Server, dann wird Port 443 nicht benötigt.
443	Converter-Server	Helper-VM	–
22	Helper-VM	Quelle	Die Hilfs-VM baut eine SSH-Verbindung zur Quelle auf.

Tabelle 2.13 vCenter-Converter-Kommunikationsports Linux online

Lassen Sie uns als Letztes dokumentieren, welche Freischaltungen Sie benötigen, wenn Sie eine bestehende virtuelle Maschine importieren möchten.

vCenter Converter (P2V einer bestehenden VM)			
Ports	**Quelle**	**Ziel**	**Bemerkungen**
445/139	Converter-Server	Share oder Pfad	Nutzt die Quelle NetBIOS, dann wird dieser Port nicht benötigt.
137/138 UDP			Kommt NetBIOS nicht zum Einsatz, werden diese Ports nicht benötigt.
443	Converter-Client	Converter-Server	Ist nur relevant bei einer angepassten Installation und wenn Server und Client nicht auf einem Server liegen.
443	Converter-Server	vCenter-Server	Wird nur benötigt, wenn das Ziel ein vCenter-Server ist.
443/902	Converter-Server	vSphere-Host	Ist das Ziel ein vCenter-Server, dann wird Port 443 nicht benötigt.

Tabelle 2.14 vCenter-Converter-Kommunikationsports beim Import einer virtuellen Maschine

2.4 Architektur eines vSphere-Hosts

Die Architektur eines vSphere-Hosts definiert sich aus verschiedenen Kernkomponenten. Auf diese wollen wir im Folgenden eingehen.

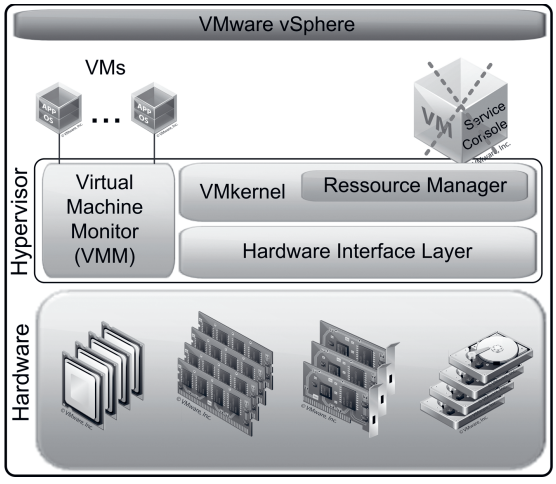


Abbildung 2.4 Struktur VMware vSphere

VMkernel

Der VMkernel ist eine sehr schlanke Implementierung des Hypervisors. Er kontrolliert und verwaltet die meisten Ressourcen eines vSphere ESX-Servers. Die Regelung des Zugriffs auf die Ressourcen CPU, Memory und Disk wird mit Hilfe eines Schedulers erreicht. Der Kernel hat neben einem TCP/IP-Stack zur Netzwerkkommunikation auch einen Storage-Stack für die Kommunikation mit Speichermedien. Der VMkernel ist eine Eigenentwicklung von VMware und nicht, wie viele meinen, ein Linux-Derivat.

VMkernel Resource Manager

Der Resource Manager partitioniert die Hardware, um den virtuellen Maschinen mit Hilfe eines Share-Mechanismus die Ressourcen zur Verfügung zu stellen. Dabei werden die Einstellungen zur Reservierung und zur Limitierung der Ressourcen CPU und Memory beachtet sowie die Shares aller Core Four (CPU, Memory, Network und Disk) berücksichtigt.

Der Resource Manager wird als Teilprozess des VMkernels gestartet.

Virtual Machine Monitor (VMM)

Der Virtual Machine Monitor ist für die Virtualisierung der CPU zuständig. Er gibt die CPU-Befehle der virtuellen Maschine an die Physik weiter. Außerdem kümmert er sich um die Verwaltung der virtuellen Maschine nach deren Start.

VMkernel Hardware Interface Layer

Der Hardware Interface Layer setzt die Hardwareanfragen der VM in die physische Adressierung um und ermöglicht so eine Adressierung der Ressourcen. Außerdem koordiniert er die Bereitstellung des VMFS und der spezifischen Gerätetreiber.

Service Console

Die Service Console ist eine priorisierte virtuelle Maschine zur Verwaltung des vSphere-Servers und läuft mit einer abgespeckten Version des Red Hat Enterprise Linux 5.1.

Die Service Console bietet dem VI-Administrator ein kommandozeilen-basiertes Verwaltungs-Interface für alle host-bezogenen Tätigkeiten, speziell für den VMKernel. Die Interaktion mit dem Anwender erfolgt über verschiedene Wege. Lokal am Server selbst erreichen Sie die Service Console über den Konsolenzugang. Dazu steht nach dem Booten des vSphere-Servers ein Linux-Prompt zur Anmeldung bereit. Auch über native Unix-/Linux-Wege etablieren Sie einen SSH-Zugang zur Service Console. Dazu existieren auf dem Markt verschiedene Tools

(z. B. puTTY). VMware stellt ein Webmanagement-Interface bereit, das heißt, sie können Teile der Verwaltungsaufgaben über einen Webbrowser erledigen.

Neben dem administrativen Zugang zur Console dient diese als Plattform für Agenten von Drittherstellern. Hersteller von Backup-Lösungen oder Überwachungssoftware integrieren vSphere-Server, indem Sie einen Agenten auf dem Betriebssystem der Service Console installieren.

2.5 Grundlagen der CPU-Virtualisierung

Bevor wir jetzt näher auf die CPU-Virtualisierung eingehen, möchten wir Ihnen zeigen, wie der logische Aufbau eines Prozessors aussieht. Als Beispiel haben wir das Schema einer Hexa-Core-CPU (6 Kerne) von AMD ausgewählt (Abbildung 2.5).

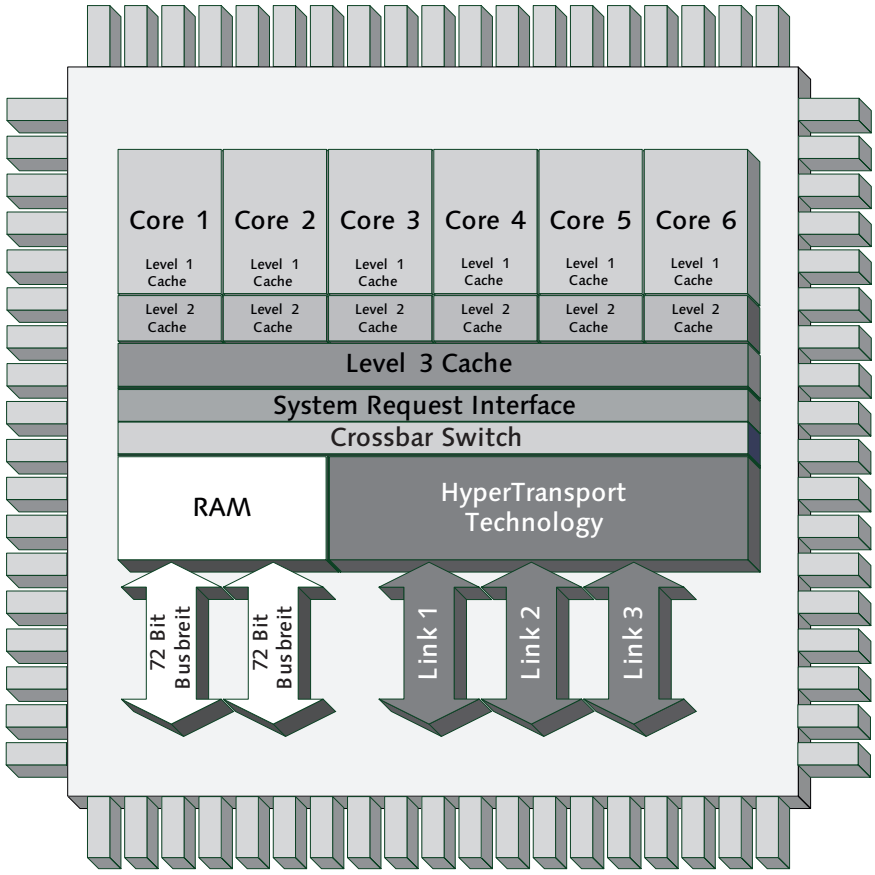


Abbildung 2.5 Logischer Aufbau einer AMD-CPU

Unterhalb eines jeden Prozessor-Cores liegt der zugehörige Cache. Der Cache bildet das Bindeglied zwischen der CPU und dem Arbeitsspeicher. Der Arbeitsspeicher kann nicht so schnell getaktet werden wie der Cache. Damit aber der Zugriff auf den Speicher nicht das System ausbremst, werden im Cache Daten abgelegt, auf die das System öfter zugreifen muss. Spezielle Mechanismen sorgen dafür, dass die Daten im Cache getauscht werden, wenn das System andere Informationen häufiger benötigt als die derzeit gespeicherten.

Der Cache gliedert sich in drei unterschiedliche Ebenen: Der Level-1-Cache ist direkt im Core integriert. Der Level-2-Cache ist ebenfalls dem einzelnen Core direkt zugeordnet. Erst der Level-3-Cache ist übergreifend allen Cores einer CPU zugänglich.

Über das System Request Interface (SRI) und den Crossbar Switch kommunizieren die einzelnen Cores untereinander und mit den weiteren Komponenten der CPU, wobei der SRI die Priorisierung der CPU-Anfragen vornimmt.

Der Speicher-Kontroller ist direkt auf dem Prozessor integriert, somit kann die CPU direkt mit dem Speicher Daten austauschen, ohne über einen zusätzlichen externen Speicherkontroller die Informationen verschicken zu müssen. Das erhöht zum einen die Performance, und zum anderen findet kein konkurrierender Zugriff der CPUs auf den Arbeitsspeicher statt.

Die Hypertransport Technology (HT) bildet das Bindeglied zwischen dem Prozessor und der Peripherie im Server. Sie hat aber auch die Aufgabe, sich um die Kommunikation der einzelnen Prozessoren untereinander zu kümmern. Soll ein Befehl ausgeführt werden, wird eine Broadcast-Meldung an alle Prozessoren geschickt, damit gewährleistet ist, dass wirklich die aktuellsten Daten verarbeitet werden.

Pro Core wurden früher mindestens zehn Nachrichten für den Abgleich benötigt. In einem Vier-Sockel-System (das heißt 48 Cores bei Hexa-Core-Systemen) bremsen die Nachrichten unter Umständen das System aus. An dieser Stelle greift die HT ein, koordiniert den Abgleich innerhalb der CPU und reduziert so die Anzahl der Nachrichten auf zwei bis drei für die Verbindung der CPUs untereinander.

Nach diesem kurzen, aber wichtigen Ausflug in die Hardware eines Systems kommen wir nun zu dem Unterschied zwischen einer Virtualisierung und einer Emulation.

Eine Emulation bildet Prozessoranfragen des Gasts über Software ab. Der Gast hat in diesem Fall keinen direkten Zugriff auf die CPU. Ein Virtualisierer leitet die Prozessoranfragen des Gasts direkt an die Hardware weiter.

VMware vSphere ist ein Virtualisierer. Unter vSphere wird die CPU einer virtuellen Maschine direkt vom Host-System abgeleitet und auch für bestimmte Arten von CPU-Instruktionen teilweise physisch verwendet. Aus diesem Grunde sieht eine VM dieselbe CPU, wie sie im Host vorhanden ist.

Die virtuelle CPU einer VM kann CPU-Instruktionen in zwei verschiedenen Modi abarbeiten: im Direct Execution Mode und im Virtualization Mode. In den meisten Fällen werden die CPU-Instruktionen im Direct Execution Mode ausgeführt, der nahe an der Geschwindigkeit der realen CPU liegt. Sollte die Ausführung des Befehls nicht in diesem Modus durchführbar sein, wird der Virtualization Mode verwendet. Eine virtualisierte CPU bedient sich so oft wie möglich der realen physischen CPU-Ressource, und die Virtualisierungsschicht greift nur bei der Ausführung von bestimmten CPU-Instruktionen ein.

Durch diese Umsetzung entsteht der oft erwähnte Virtualisierungs-Overhead. Den Virtualisierungs-Overhead beziffern wir näher in Abschnitt 2.6.2, »Memory-Overhead«, im Zusammenhang mit dem Arbeitsspeicher.

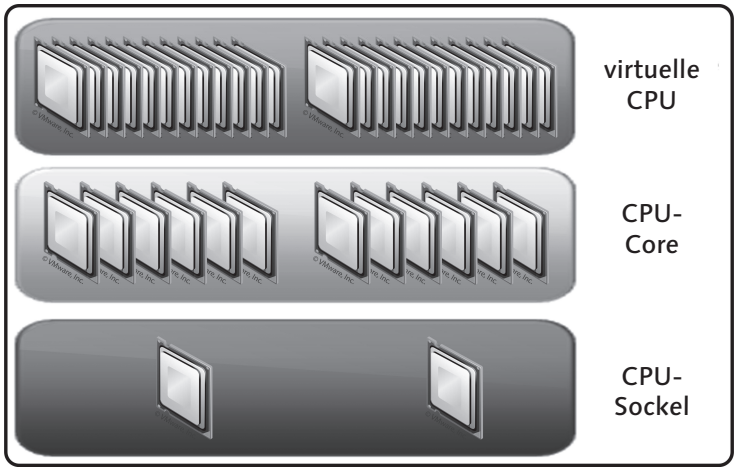


Abbildung 2.6 Zusammenhang zwischen physischen, logischen und virtuellen CPUs

Dazu sei als Hintergrund erwähnt, dass eine CPU grundsätzlich vier Priviligierungsstufen hat, sogenannte Ringe oder auch Domains (Abbildung 2.7). Die höchste Priorität hat Ring 0. Hier liegt der sogenannte Supervisor Mode, der manipulativ auf Hauptspeicher und Interrupts zugreifen darf. In dieser Stufe läuft normalerweise der Kernel des Betriebssystems, im Falle von VMware vSphere also der Hypervisor-VMkernel. In den Ringen 1 bis 3 liegt der User-Mode, wobei normalerweise nur Ring 3 genutzt wird; es gibt nur wenige Applikationen, die direkt auf Ring 1 oder Ring 2 zugreifen.

Bei einer virtuellen Maschine verhält sich das etwas anders: Die eigentlich an Ring 0 gestellten Anfragen des Betriebssystems werden an Ring 3 umgeleitet. Damit die Daten in Ring 1 bis 3 verarbeitet werden können, wird der physische Speicher in virtuelle Speicherseiten aufgeteilt. Der Memory-Controller (Memory Management Unit, MMU) übernimmt an dieser Stelle die Umsetzung von physischen in virtuelle Speicherinhalten. Damit der Programmcode auch richtig ausgeführt werden kann, enthält jede Speicherseite die Information, auf welchem Ring der Code ausgeführt werden muss. Um zu verhindern, dass ein solch komplexes System beeinflusst wird – z. B. durch Schadcode –, wurde das sogenannte NX-Flag kreiert (No Execution Flag). Diese Information hilft dem System, Daten von Programmcode zu unterscheiden. Dieser Mechanismus verhindert, dass Programmcode im Bereich der Daten ausgeführt werden kann.

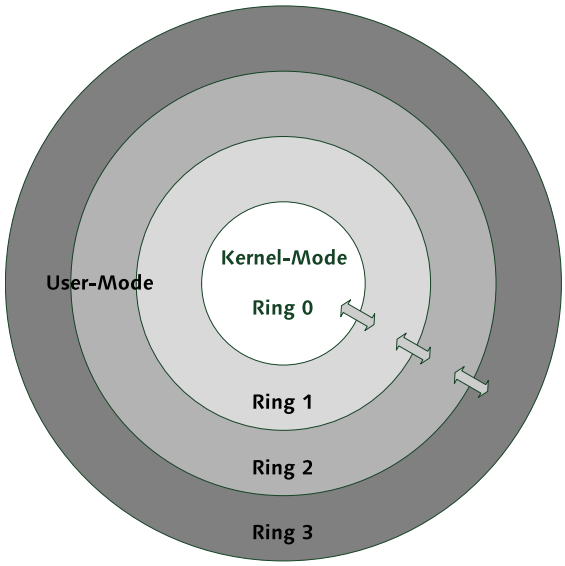


Abbildung 2.7 Ringstruktur der CPU

Applikationen rufen in der Regel den unprivilegierten Ring einer CPU an, daher laufen diese Befehle im Direct Execution Mode. Wird hingegen eine Instruktion vom Betriebssystem ausgeführt, geschieht dies in der Regel modifizierend auf den privilegierten Ring der CPU. Diese Anfragen werden von der Virtualisierungsschicht, dem VMM, abgefangen.

Dieser Managementaufwand wird als der Virtualisierungs-Overhead bezeichnet. Er hängt ab von der Arbeitslast der virtuellen CPU und der Menge der Aufrufe an den privilegierten Ring. Die Auswirkungen zeigen sich in verlängerten Laufzeiten der einzelnen Befehle und durch eine erhöhte CPU-Last.

Die reale CPU wird an das Betriebssystem der VM durchgereicht. Aus diesem Grund sind dem Betriebssystem auch die Besonderheiten der eingesetzten CPU bekannt. Verschiedene Betriebssysteme nutzen diese CPU-spezifischen Befehle. Es kann auch während der Installation der Gast auf diese Besonderheiten hin optimiert worden sein. Ein Verschieben einer solchen speziellen VM auf andere vSphere-Server mit unterschiedlichen CPUs, insbesondere beim Wechsel zwischen Intel- und AMD-Prozessoren, beeinträchtigt unter Umständen die Funktionalität des Betriebssystems beziehungsweise der Applikation.

2.5.1 CPU-Affinity

Die CPU-Affinität bezeichnet eine Konfigurationsoption der virtuellen Maschine, und zwar die direkte Zuweisung einer physischen CPU bzw. eines Kerns. Diese Technik sollten Sie nur in Ausnahmefällen (z. B. Troubleshooting) verwenden, weil sie etliche Auswirkungen auf andere Bereiche der virtuellen Infrastruktur hat. Zum einen wird dadurch die CPU-Lastverteilung des ESX-Servers außer Kraft gesetzt. Zum anderen kollidiert diese CPU-Zuordnung mit eventuell vorgenommenen Einstellungen von CPU-Shares und CPU-Reservierung. Durch das Umgehen der CPU-Lastverteilung kann der Hypervisor den Forderungen seitens der VM eventuell nicht mehr nachkommen. Die mögliche Virtualisierungsquote und die Flexibilität reduzieren sich. Die Nutzung von VMotion ist durch die CPU-Affinität eingeschränkt, und DRS verhindert diese sogar.

2.5.2 Hyperthreading

Der vSphere-Server unterstützt die Hyperthreading-Technologie von Intel. Diese bietet bei Nutzung von Ein-Sockel-Prozessoren der Pentium-4- und der Xeon-Reihe ein auf Hardwareebene realisiertes Multithreading zur Verbesserung der CPU-Performance. Lange Zeit war Pause mit hyperthreading-fähigen CPUs, bis Intel diese in den neuen 5500-Xeon-Prozessoren wieder integrierte. Dabei kann eine physische CPU – im Intel-Wortgebrauch wird sie als Hyperthread bezeichnet – gleichzeitig zwei Threads ausführen. Sie verhält sich mit aktiviertem Hyperthreading ähnlich wie zwei logische CPUs. Sofern ein Betriebssystem und die darauf laufenden Applikationen zwei CPUs nutzen können, sind hierdurch Geschwindigkeitsvorteile möglich. Dabei reicht die Performance nicht an eine Verdoppelung heran, wie sie durch einen Dual-Core-Prozessor erreicht würde, die Leistung einer CPU verbessert sich aber je nach Anwendung auf dem Betriebssystem signifikant. Ungeeignete Applikationen werden durch die Hyperthreading-Technologie unter Umständen auch verlangsamt, wenn sie zu viel der gemeinsam genutzten Ressourcen eines Cores verwenden.

Auf der Hardwareebene muss das Hyperthreading im BIOS aktiviert sein. Im Host ist Hyperthreading per Default aktiv; bei Bedarf deaktivieren Sie es über den vSphere-Client im Tab Configuration eines vSphere-Hosts unter den Eigenschaften der CPU. Bedenken Sie bitte beim Einsatz von Hyperthreading, dass ein vSphere-Host nur eine bestimmte Anzahl von CPUs unterstützt.

Der vSphere-Server verteilt die Last zwischen den Cores, um eine ausgewogene Auslastung zu erreichen. Wenn für eine logische CPU keine Last gefordert wird, wird sie in einen speziellen halt state geschaltet. Dabei kann eine andere VM auf dem Core von den zusätzlichen freien Ressourcen dieser CPU profitieren.

Um virtuelle Maschinen mit für Hyperthreading problematischen Anwendungen ohne dieses Feature zu betreiben, bietet vSphere auf Ebene der VM drei verschiedene Modi des Verhaltens an, siehe Tabelle 2.15.

Parameter	Funktion
any	Dies ist die Standardeinstellung und ermöglicht das Teilen der logischen CPUs eines Cores mit entweder einer weiteren virtuellen CPU der identischen VM oder einer virtuellen CPU einer anderen VM. Diese Einstellung bietet, sofern die Anwendungen dafür ausgelegt sind, die optimale Performance.
none	Diese Einstellung schaltet das Hyperthreading pro virtueller Maschine aus. Eine virtuelle CPU wird einer logischen CPU eines Cores zugeordnet, die zweite logische CPU wird in den halted state geschaltet. Da diese Einstellung eine virtuelle CPU vom restlichen System isoliert, wird diese Konfiguration für hyperthreading-problematische Applikationen verwendet und sollte nur nach Aufforderung durch den VMware-Support oder den Support des Anwendungsherstellers implementiert werden.
internal	Diese Einstellung beschränkt die Nutzung der zwei logischen CPUs auf eine VM und gilt daher nur für VMs mit aktiviertem vSMP. Eine VM teilt sich den Core nicht mit anderen VMs, sondern der Core wird nur für die virtuellen CPUs einer VM verwendet. Haben Sie diese Einstellung für eine Uniprocessor-VM ausgewählt, schaltet vSphere diese Einstellung automatisch auf None.

Tabelle 2.15 Hyperthreading-Parameter

Diese Einstellungen haben keinen Einfluss auf die Verteilung und Priorisierung von CPU-Ressourcen an die virtuelle Maschine.

2.5.3 Virtual SMP (vSMP)

Auch in virtuellen Umgebungen ist es nicht nur möglich, virtuelle Maschinen mit einer vCPU zu erstellen. Die aktuelle Version von VMware vSphere unterstützt

bis zu acht virtuelle CPUs pro VM. VMware nennt diese Funktion Virtual SMP (Symmetric MultiProcessing) oder auch vSMP. Dabei ist einiges zu beachten: Grundsätzlich – das unterscheidet eine virtuelle Maschine nicht von einem physischen Server – ist nicht jede Applikation multiprozessorfähig. Vor der Erzeugung einer vSMP-Maschine sollten Sie dies abklären und dabei nicht nur das Betriebssystem (auf die HAL bzw. den Kernel achten), sondern auch die Anwendung beachten.

Schauen wir noch einmal zurück auf den Beginn des Abschnitts 2.5, in dem wir den logischen Aufbau einer CPU erklärt haben. Da es allen CPUs einer VM möglich sein muss, auf identische Speicheradressen zuzugreifen, auch beim Cache, wird sofort klar, dass eine virtuelle Maschine mit mehreren CPUs am leistungsfähigsten arbeiten kann, wenn alle virtuellen Prozessoren auf einer logischen oder physischen CPU liegen. Liegen die Prozessoren auf unterschiedlichen Sockeln, dann können die virtuellen CPUs nicht in optimaler Geschwindigkeit miteinander kommunizieren. Die Ursache dafür ist, dass der Informationsaustausch der CPUs untereinander über den Frontside-Bus erfolgen muss.

Auch beim Betriebssystem müssen Sie auf einiges achten. Denken Sie bitte daran, dass Sie bei mehreren CPUs in einer VM einen Multiprozessor-Kernel installieren müssen. Einen Weg zurück, zumindest bei Windows-VMs, unterstützt Microsoft nicht.

Lassen Sie uns nun betrachten, wie VMware mit dem Thema vSMP und der Tatsache, dass freie Ressourcen anderen VMs zur Verfügung gestellt werden, umgeht. Während eine CPU im physischen Umfeld exklusiv einem Betriebssystem zur Verfügung steht, teilen sich die virtuellen Maschinen die CPU-Zyklen. Zur optimalen Abarbeitung der Prozesse werden diese in einem SMP- bzw. vSMP-System parallelisiert. Steht eine teilprozessabarbeitende Instanz nicht zur Verfügung, müssen alle anderen Teilprozesse so lange warten, bis auch dieser Prozess parallel zu den anderen abgearbeitet wurde. Diese Art der parallelen Abarbeitung wird auch *Co-Scheduling* genannt und dient grundsätzlich dazu, die Performance eines Systems zu erhöhen.

Es könnte vorkommen, dass ein Watchdog-Timer auf einen Schwesterprozess warten muss. Reagiert dieser Prozess aber nicht in einem passenden Zeitfenster, stirbt er. Zur Messung dieser Varianzen wird der sogenannte Skew herangezogen. Dieser Wert repräsentiert den zeitlichen Unterschied zwischen den Prozessteilen. Überschreitet der Skew einen definierten Schwellenwert, dann wird die CPU der VM mit angehalten (co-stopped). Sie wird erst wieder mitgenutzt (co-started), wenn genügend Ressourcen für die Abarbeitung auf der physischen CPU vorhanden sind. Der Co-Stop verhindert, dass der Skew-Wert sich erhöht, er

kann nur sinken. Mit dem Relaxed Co-Scheduling wurde mit ESX 3 nämlich eine Funktion eingeführt, dass angehaltene vCPUs keine Skew-Wert-Erhöhung mehr erfahren. Somit wird ein zu häufiges Co-Scheduling verhindert.

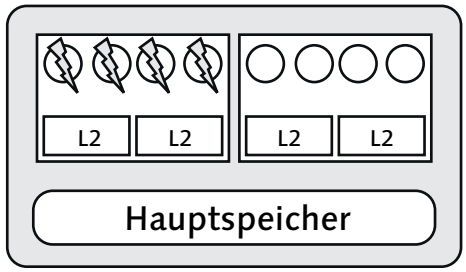


Abbildung 2.8 SMP-Handling unter ESX 3

Der Skew-Wert hat aber noch eine weitere Funktion: Der VMkernel nutzt diesen Wert, um die Arbeitslast auf die physischen CPUs zu verteilen. Eine geskewte CPU hat Rechenzeit übrig, die andere VMs nutzen können.

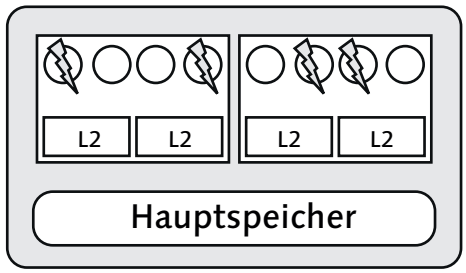


Abbildung 2.9 SMP-Handling unter ESX 4

vSphere bringt wesentliche Änderungen gegenüber ESX 3 mit, da neben der deutlichen Minderung des Co-Stoppings nun auch die Nutzung aller Prozessorkerne ermöglicht wurde. Dadurch wurde ein wesentliches Problem der ESX 3-Welt gelöst, da CPU-Anfragen der VMs teilweise unnötig warten mussten, weil nicht genügend Kerne einer physischen CPU verfügbar waren. Somit hat der CPU-Scheduler unter ESX 4 wesentlich mehr Möglichkeiten, CPU-Anfragen zu verteilen.

ESX 3.x	ESX 4
V0, V1, V2, V3	V0, V1, V2, V3
V0, V1, V2	V0, V1, V2
V0, V1, V3	V0, V1, V3

Tabelle 2.16 SMP-Vergleich zwischen ESX 3 und ESX 4

ESX 3.x	ESX 4
V0, V1	V0, V1
	V0, V2
	V1, V2
	V0
	V1
	V2

Tabelle 2.16 SMP-Vergleich zwischen ESX 3 und ESX 4 (Forts.)

Es gilt zwar auch weiterhin unter vSphere vSMP, dass weniger mehr ist, allerdings ist es wesentlich entspannter geworden, Mehrprozessor-VMs zu verwenden. Das Hauptkriterium sollte immer noch die Anforderung der Anwendung und des Systems sein und nicht der Gedanke, dass mehr CPUs auch automatisch mehr Leistung bedeuten.

2.5.4 Best Practices

Nachfolgend finden Sie einige Empfehlungen zum Umgang mit CPU-Reservation, -Limits und -Shares:

- Erfahrungsgemäß werden Prozessoren nicht zurückgerüstet, auch wenn sie eigentlich nicht benötigt werden. Bei Mehrprozessor-VMs fangen Sie einfach mit einer Zweiprozessor-Maschine an. Weitere Prozessoren lassen sich immer noch später hinzukonfigurieren. Vergeben Sie niemals mehr Prozessoren, als sich Cores auf der CPU befinden.
- Einer virtuellen Maschine sollten Sie zu Beginn grundsätzlich niedrige CPU-Ressourcen zuweisen, um im laufenden Betrieb die Ressourcenauslastung anhand der vCenter-Performance-Messung zu analysieren.
- Es ist besser, mit CPU-Shares anstelle von CPU-Reservierungen zu arbeiten.
- Beim Einsatz von CPU-Reservierungen sollten Sie das aktuelle Minimum definieren, das eine VM benötigt, nicht aber die gewünschte absolute Menge an CPU in MHz. Wenn eine VM mehr Ressourcen benötigt, so weist der vSphere-Server diese, je nach definierten Shares, bis zu einem eventuell definierten CPU-Limit dynamisch zu. Des Weiteren ist zu beachten, dass der Einsatz von CPU-Reservierungen die auf einem Host zur Verfügung stehenden CPU-Ressourcen limitieren und dadurch weniger VMs gestartet werden können. Zu hohe Reservierungen behindern möglicherweise auch Funktionen wie DRS oder HA. Das Verschieben von virtuellen Maschinen kann durch die Ressourcenauslastung der vSphere verhindert werden.

2.6 Grundlagen der Memory-Virtualisierung

Der physische Speicher eines Hosts wird in drei Segmente unterteilt: System, Virtual Machines und Service Console. Der Speicherbereich für *System* wird vom VMkernel und von den Gerätetreibern verwendet und ist nicht konfigurierbar. Er wird mit einer Größe von mindestens 50 MByte beim Starten des vSphere-Hosts angelegt und variiert je nach Anzahl und Art der verwendeten PCI-Geräte und deren Treibern. Der Speicherbereich für die *Service Console* ist konfigurierbar – bis zu einem Maximalwert von 800 MB – und bleibt für die Verwendung innerhalb der Service Console reserviert. Diese benötigt je nach installierten Agenten, z. B. für Backup oder Monitoring, eine gewisse Menge an Speicher. Der Speicherbereich für *Virtual Machines* ist der Rest des physischen Speichers und wird komplett für die VMs genutzt.

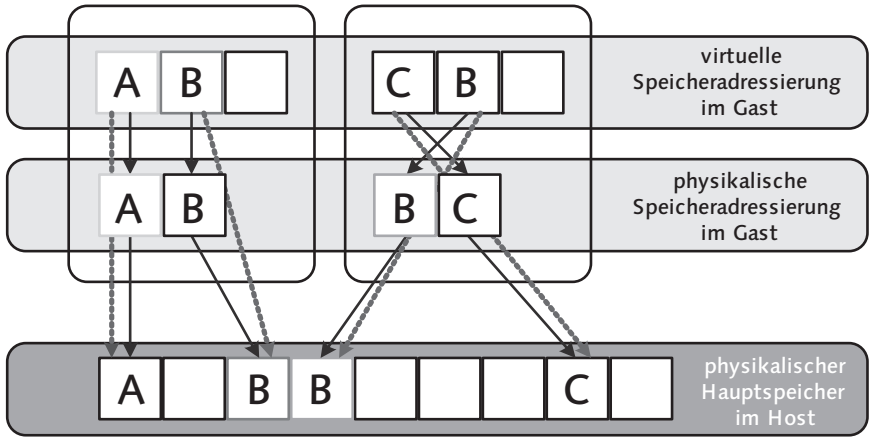


Abbildung 2.10 Speicheradressierung zwischen VM und Host

Zur Verdeutlichung hier zuerst eine Erklärung zur generellen Nutzung von Speicher innerhalb eines Betriebssystems. Dieser wird in einem Betriebssystem über virtuelle Speicheradressen erreicht, die auf physische Adressen verweisen. Ein Zugriff von einer virtuellen Maschine auf den physischen Speicher eines vSphere-Hosts ist nicht erlaubt. Um den virtuellen Maschinen Speicher zur Verfügung zu stellen, bietet vSphere eine weitere, virtuelle Schicht. Diese gaukelt der VM die physischen Speicheradressen vor. Im VMware-Wortgebrauch heißt der physische Speicher im Host Machine Memory Pages, und die der VM virtualisiert vorgegaukelten physischen Speicherseiten nennen sich Physical Memory Pages. Die Physical Memory Pages für eine VM sind, so wie es ein Betriebssystem erwartet, mit durchgängigen Nullen gefüllt. Sie sind durch die Virtualisierungsschicht aus verschiedenen Bereichen, aber nicht zusammenhängend, zusammen-

gefasst. Diese Bereiche sind z. B. normale, physische Speicherbereiche (Machine Memory Pages), von vSphere Shared Pages oder auch Swapped Pages. Das virtuelle Speichermanagement erfolgt durch den Host, unabhängig vom in der VM laufenden Betriebssystem, über den VMkernel. Dieser greift von der VM alle Befehle ab, die auf den Speicherbereich schreibend zugreifen möchten, und leitet diese auf die der VM vorgegaukelten Physical Memory Pages um.

Der Speicher wird normalerweise in 4-KB-Blöcke eingeteilt. Es werden aber auch Memory-Blöcke von 2 MB unterstützt. Diese Funktion können Sie nur pro VM konfigurieren. Dazu aktivieren Sie in dem Konfigurations-File die Funktion `Mem.AllocGuestLargePage=1`. Dies ist empfehlenswert, wenn die VM große Speicherseiten benötigt, wie z. B. ein Datenbank-Server.

2.6.1 Virtual Machine Memory

Der Speicherbereich, der für die VMs zur Verfügung steht, wird Virtual Machine Memory genannt und bietet allen VMs die Speicherressourcen des vSphere-Servers abzüglich eines Virtualisierungs-Overheads. Dem Betriebssystem wird vorgegaukelt, dass der Speicher, der in der Konfiguration festgelegt wurde, auch vorhanden ist. Der physisch zugewiesene Speicher kann aber variieren, bis zum konfigurierten Maximum.

Auch hier setzen Sie über die Einstellung der Shares-Werte eine Priorität gegenüber den anderen VMs, die auf demselben Host arbeiten. Eine Reservierung weist den Speicher der virtuellen Maschine fest zu.

2.6.2 Memory-Overhead

Der Memory-Overhead hängt ab von der Anzahl der CPUs, der Art des Gast-Betriebssystems (32 oder 64 Bit) und natürlich dem der VM zugewiesenen Speicher. Dieser Memory-Overhead stellt einen Speicherbereich zur Verfügung, um VM-Frame-Buffer sowie auch verschiedene Virtualisierungsdatenstrukturen abzulegen.

Die Nutznießer dieses Speichers sind zum einen der VMkernel und zum anderen der VMM, die beide für die korrekte Arbeit der virtuellen Maschinen benötigt werden.

Der einzukalkulierende Overhead ist in Tabelle 2.17 aufgeführt. Die Aufstellung berücksichtigt die Abhängigkeit zwischen der Anzahl der vCPUs und dem zugewiesenen Arbeitsspeicher. Durch Nutzung eines 64-Bit-Betriebssystems verdoppeln sich die Werte.

Memory (MB)	1 vCPU	2 vCPUs	3 vCPUs	4 vCPUs	5 vCPUs	6 vCPUs	7 vCPUs	8 vCPUs
256	113,17	159,43	200,53	241,62	293,15	334,27	375,38	416,50
512	116,68	164,96	206,07	247,17	302,75	343,88	385,02	426,15
1.024	123,73	176,05	217,18	258,30	322,00	363,17	404,34	445,52
2.048	137,81	198,20	239,37	280,53	360,46	401,70	442,94	484,18
4.096	165,98	242,51	283,75	324,99	437,37	478,75	520,14	561,52
8.192	222,30	331,12	372,52	413,91	591,20	632,86	674,53	716,19
16.384	334,96	508,34	550,05	591,76	900,44	942,98	985,52	1.028,07
32.768	560,27	863,41	906,06	948,71	1.515,75	1.559,42	1.603,09	1.646,76
65.536	1.011,21	1.572,29	1.616,19	1.660,09	2.746,38	2.792,30	2.838,22	2.884,14
131.072	1.912,48	2.990,05	3.036,46	3.082,88	5.220,24	5.273,18	5.326,11	5.379,05
262.144	3.714,99	5.830,60	5.884,53	5.938,46	10.142,83	10.204,79	10.266,74	10.328,69

Tabelle 2.17 Memory-Overhead bei virtuellen Maschinen

2.6.3 Memory-Overcommitment

vSphere bietet die Möglichkeit, mehr RAM an virtuelle Maschinen zu vergeben, als physisch im Host selbst vorhanden sind. Dieses Feature nennt sich Memory-Overcommitment und setzt sich zusammen aus drei verschiedenen Techniken, dem Page-Sharing, dem Memory-Ballooning und dem Memory-Swapping. Das Bestreben aller Techniken ist das Verteilen von ungenutzten Speicherbereichen von einer VM zu anderen Maschinen, die aktuell mehr Speicher benötigen. Die Priorisierung erfolgt auch in diesem Fall über die Eingestellten Share-Werte.

2.6.4 Content-based Page-Sharing

Die Page-Sharing-Technik wird beim Betrieb von mehreren VMs auf einem Host verwendet. Es wird versucht, identische Memory-Pages der VMs zusammenzufassen. Die dabei beobachtete Speicher-Blockgröße ist so klein, dass es vollkommen unerheblich ist, ob auf den virtuellen Servern identische Software installiert ist oder nicht.

Trotzdem gelingt dies umso besser, je homogener die verschiedenen Gastbetriebssysteme sind, also wenn mehr identische Server-Applikationen darauf laufen. Ein gutes Beispiel ist eine Server-Farm mit identischen Webservern, die aber alle unterschiedlichen Webcontent hosten. Es ist zu erwarten, dass diese Systeme eine große Anzahl von identischen Speicherblöcken haben, die vom VMM zusammengefasst werden können. So werden redundante Speicherinhalte eliminiert. Will nun eine der virtuellen Maschinen einen solchen Speicherbereich

beschreiben, dann wird für diesen Server eine Kopie des Speicherblocks exklusiv für ihn angelegt, so dass er ihn frei nutzen kann.

Bei dieser Technik sind bis zu 30 % Speicherersparnis erreichbar. Bei weniger homogenen Memory-Inhalten reduziert sich die Ersparnis auf ca. 5 %.

2.6.5 Memory-Ballooning

Das in Abschnitt 2.6.3, »Memory-Overkommitment«, beschriebene Memory-Overcommitment kann nur einwandfrei funktionieren, wenn dem Host ein Mechanismus zur Verfügung steht, das das Management des Arbeitsspeichers im virtuellen System übernimmt, und das natürlich im laufenden Betrieb. Dafür ist das sogenannte Memory-Ballooning zuständig.

Der Memory-Balloon-Treiber (*vmmemctl*) kommt ins Spiel, wenn der Speicher eines Hosts zu knapp wird oder wenn eine VM an ihre Speichergrenzen stößt.

Braucht der Host Speicher, dann hat Ballooning immer Vorrang vor dem Swappen.

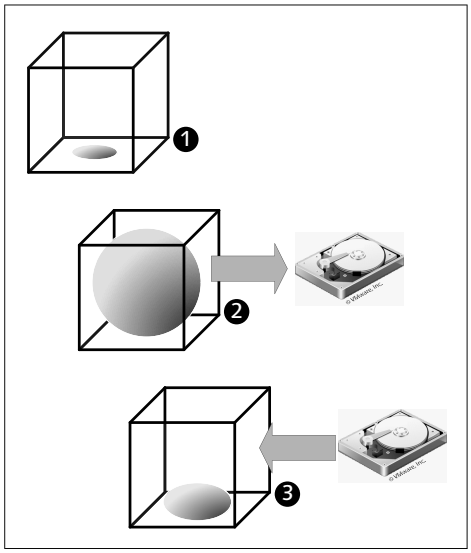


Abbildung 2.11 Darstellung des Memory-Ballonings

Wird Speicher benötigt, dann gibt der VMM dem Ballooning-Treiber das Kommando zur Anforderung von Speicher vom OS (Abbildung 2.11, ❶). Ist genug Speicher vorhanden, gibt die VM demjenigen Treiber Speicher, der in der Free-List steht. Ist kein freier Speicher vorhanden, wird dem Gast-OS überlassen, welcher Speicher freigegeben werden kann.

Der vSphere-Kernel gibt im Hintergrund die vom Ballooning-Treiber markierten Speicherseiten frei, bis genug Speicher für den Host akquiriert worden ist (❷). Anschließend beginnt der Ballooning-Treiber, den reservierten Speicher wieder freizugeben (❸).

Das Verhalten des Memory-Ballonings können Sie pro vSphere-Server durch den Parameter SCHED.MEM.MAXMEMCTL festlegen. Dieser Wert bestimmt die maximale durch diese Technik von einer virtuellen Maschine abziehbare Speichermenge und wird in Megabytes angegeben.

2.6.6 Memory-Swapping

Das Memory-Swapping dient ebenso wie das Ballooning zum Zuweisen von mehr Arbeitsspeicher an die VM. Diese Technik ist für den Host die letzte, aber auch langsamste Möglichkeit, Speicher für andere virtuelle Maschinen zur Verfügung zu stellen. Beim Start einer VM wird automatisch ein solches Swapfile angelegt. Swapping tritt zu dem Zeitpunkt in Aktion, wenn der Hypervisor nicht die Möglichkeit hat, über den Ballooning-Treiber festzustellen, welche Speicherseiten zurückgegeben werden können. Die Ursache dafür kann auch sein, dass keine VMware Tools installiert oder kein Ballooning-Treiber vorhanden ist. Bootet die VM (zu diesem Zeitpunkt sind noch keine VMware Tools aktiv), ist der Treiber auch nicht produktiv. Des Weiteren kommt diese Technik auch zum Zuge, wenn das Memory-Ballooning zu langsam ist, um den Speicherbedarf einer VM zu decken. Das Swapping ist generell langsamer als das Ballooning. Es wird eine Swap-Datei pro VM auf dem Volume abgelegt, und zwar im Verzeichnis der virtuellen Maschine. Das Swapping garantiert einer VM einen mindestens verfügbaren Speicherbedarf, damit die VM starten kann.

Dieser Speicherbereich, die Swap-Datei, ist der der VM jetzt neu zugewiesene Speicher und wird beim Einschalten einer VM angelegt. Die Größe variiert je VM und ist die Differenz zwischen dem Reservierungswert und dem zugewiesenen Speicher einer VM.

Eine Besonderheit bei der Verwendung von Memory-Swapping sollten Sie beachten: Im Falle eines Ausfalls des ESX-Servers werden diese Swap-Dateien nicht mehr automatisch gelöscht. Sie müssen sie dann manuell löschen, wozu das Stoppen und Starten einer VM notwendig wird.

2.6.7 Best Practices

Nachfolgend finden Sie einige Empfehlungen zum Umgang mit Memory-Reservation, Memory-Limits und Memory-Shares:

- Grundsätzlich sollten Sie den Einsatz von Memory-Overcommitment vermeiden. Dies bewirkt auf jeden Fall eine Verlangsamung. Sollte die Überbelegung des Speichers unvermeidbar sein, achten Sie darauf, dass nicht das Memory-Swapping genutzt wird, denn dies reduziert die Performance einer VM deutlich.
- Sie sollten Memory-Shares gegenüber von Memory-Reservierungen den Vorzug geben.
- Beim Einsatz von Memory-Reservierungen sollten Sie ein Minimum an RAM definieren, den die eine VM benötigt. Falls eine VM mehr Ressourcen benötigt, so werden diese vom Host, je nach Shares, bis zum eventuell definierten Memory-Limit dynamisch zugewiesen. Beachten Sie außerdem, dass der Einsatz von Memory-Reservierungen die auf einem vSphere-Server zur Verfügung stehenden Speicherressourcen limitiert. Somit können weniger VMs gestartet werden, selbst dann, wenn andere VMs den reservierten Speicherbereich nicht nutzen. Auch kann DRS in seiner Funktion behindert werden, da hier die Ressourcenauslastung des Hosts ein Verschieben von virtuellen Maschinen verhindert.
- Ein Delegieren von Ressourcen-Management erreichen Sie idealerweise durch die Einführung von Resource-Pools. Dabei geben Sie die Grenzen des Resource-Pools an, also die Reservierung und das Limit, um die darin laufenden virtuellen Maschinen von den weiteren Ressourcen eines Hosts zu isolieren.

2.7 Grundlagen der Hardwarevirtualisierung

Wie wir bis jetzt gezeigt haben, wird bei der klassischen Virtualisierung dem Gast eine virtuelle Hardware zur Verfügung gestellt. Das sehen Sie sehr schön, wenn Sie eine virtuelle Maschine booten: Sofort ist der vom Computer bekannte BIOS-Schirm sichtbar. Gehen Sie in die Tiefen des BIOS, stellt sich die virtuelle Maschine wie ein ganz normaler Computer dar. Alle Elemente eines Computers werden in der VM emuliert, seien es der Festplatten-Controller, die Netzwerkkarte oder andere Hardwareelemente. Wie schon beschrieben, handelt es sich um einen »normalen« PC, nur eben virtuell. Der Vorteil besteht darin, dass Sie ein Betriebssystem – die passenden Treiber vorausgesetzt – einfach in die virtuelle Hülle bringen können. Anschließend installieren Sie die Applikation, und fertig ist der virtuelle Server.

Es gibt aber noch andere Varianten von virtuellen Maschinen, die sogenannten paravirtualisierten VMs.

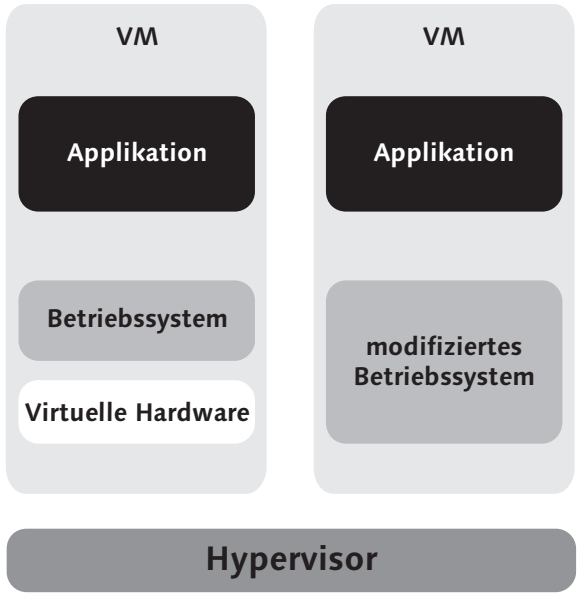


Abbildung 2.12 Unterschied zwischen klassischer und paravirtueller VM

Abbildung 2.12 stellt den Unterschied zwischen beiden Varianten dar. Sie veranschaulicht, dass bei der paravirtuellen Maschine der Layer der virtuellen Hardware fehlt. Dafür existiert eine definierte Schnittstelle. Sie steuert die Ressourcen und den direkten gemeinsamen Zugriff auf die physische Hardware. Ein solcher Mechanismus kann aber nur funktionieren, wenn dem Betriebssystem der Hypervisor »bekannt« ist. Als Ergebnis erhöht sich die Performance der virtuellen Maschine, denn es fehlt die Schicht der virtuellen Hardware.

Die direkte Kommunikation zwischen dem Gastsystem und dem Hypervisor wird als Paravirtualisierung bezeichnet.

Es gibt aber nur wenige Betriebssysteme, die die Paravirtualisierung unterstützen. Ursache dafür sind die starken Eingriffe, die in dem Kernel erforderlich sind. Einzig einige freie Betriebssysteme unterstützen die Paravirtualisierung. Es handelt sich um verschiedene Linux-Derivate. Der Grund ist relativ logisch, denn der Kernel ist frei, und somit kann, das Wissen vorausgesetzt, der Kernel für die Paravirtualisierung angepasst werden.

Im Gegensatz zu dem kompletten Ansatz, dass der Layer der virtuellen Hardware vollständig entfällt, gibt es Teilansätze, auf die wir hier kurz eingehen wollen. Lassen Sie uns zuvor etwas ausholen: Warum geht VMware diesen Weg, und welche Vorteile bringen diese Technologien?

Der Layer, der Hardwarevirtualisierung, ist eine Softwarekomponente, die die Hülle für die VM simuliert. Das bedeutet aber im Gegenzug, dass alle Aktionen, die über diese Schicht laufen, Last in diesem Layer erzeugen, bevor die Daten an die eigentliche Hardwarekomponenten gelangen, wie z. B. die Netzwerkkarte. Das erzeugt Rechenzeit auf der CPU und bremst die Performance. Der Ansatz, dem nun gefolgt wird, ist, Teilkomponenten zu paravirtualisieren. Der Vorteil dabei ist, dass nicht der gesamte Kernel angepasst werden muss, sondern dass es reicht, passende »Hardwaretreiber« für das Gastbetriebssystem zur Verfügung zu stellen.

An zwei Stellen gibt es bereits entsprechende Ansätze: bei der neuen Netzwerkkarte vmxnet3 und bei dem paravirtualisierten SCSI-Adapter (PVSCSI).

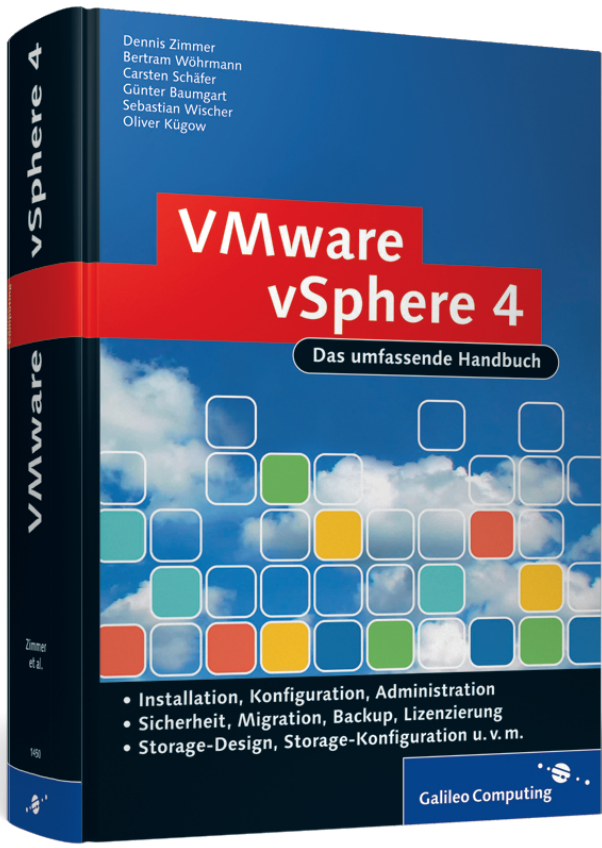
Auf die Funktionen der beiden Adapter gehen wir an entsprechender Stelle im Kapitel 13, »Virtuelle Maschinen«, im Buch ein.

Lassen Sie uns jetzt die allgemeinen Beschreibungen verlassen und direkt in die Materie vSphere 4 einsteigen.

Dennis Zimmer, Bertram Wöhrmann, Carsten Schäfer,
Günter Baumgart, Sebastian Wischer, Oliver Kügow

Dieses Bonus-Kapitel stammt aus dem Buch

VMware vSphere 4



Galileo Press