UNIVERSITÄT LEIPZIG

Mainframe Internet Integration

Prof. Dr. Martin Bogdan
Prof. Dr.-Ing. Wilhelm G. Spruth

SS2013

Virtualisierung Teil 2

Host/Gast Status

z/VM (VM/370) Betriebssystem

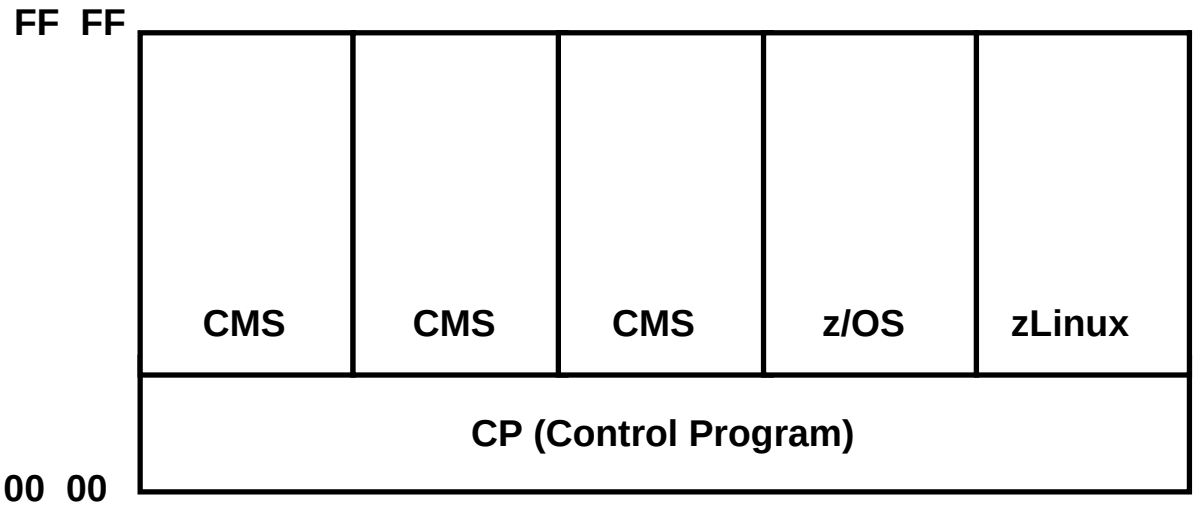
Für Mainframe Rechner existiert seit 1972 ein spezielles Betriebssystem VM/370, welches heute fast ausschließlich für Virtualisierungsaufgaben eingesetzt wird.

Ein Vorläufer wurde 1965 unter der Bezeichnung CP/40 im IBM Scientific Center in Cambridge/Mass. entwickelt. Eine spezielle Version wurde 1967 für das Rechner Modell S/360-67 für den Vertrieb freigegeben.

Seitdem wurden zahlreiche Verbesserungen und Erweiterungen eingeführt. Von besonderer Bedeutung war die Entwicklung der „Interpretive Execution Facility“. (IEF) im Jahre 1981. Die heutige Bezeichnung des Betriebssystems ist z/VM.

Unter CP/67 und VM/370 laufen alle Anwendungen grundsätzlich auf virtuellen Maschinen. Der Host Kernel (Hypervisor) wird als CP (Control Programm) bezeichnet und läuft im Kernelstatus.

http://www.multicians.org/thvv/360-67.html



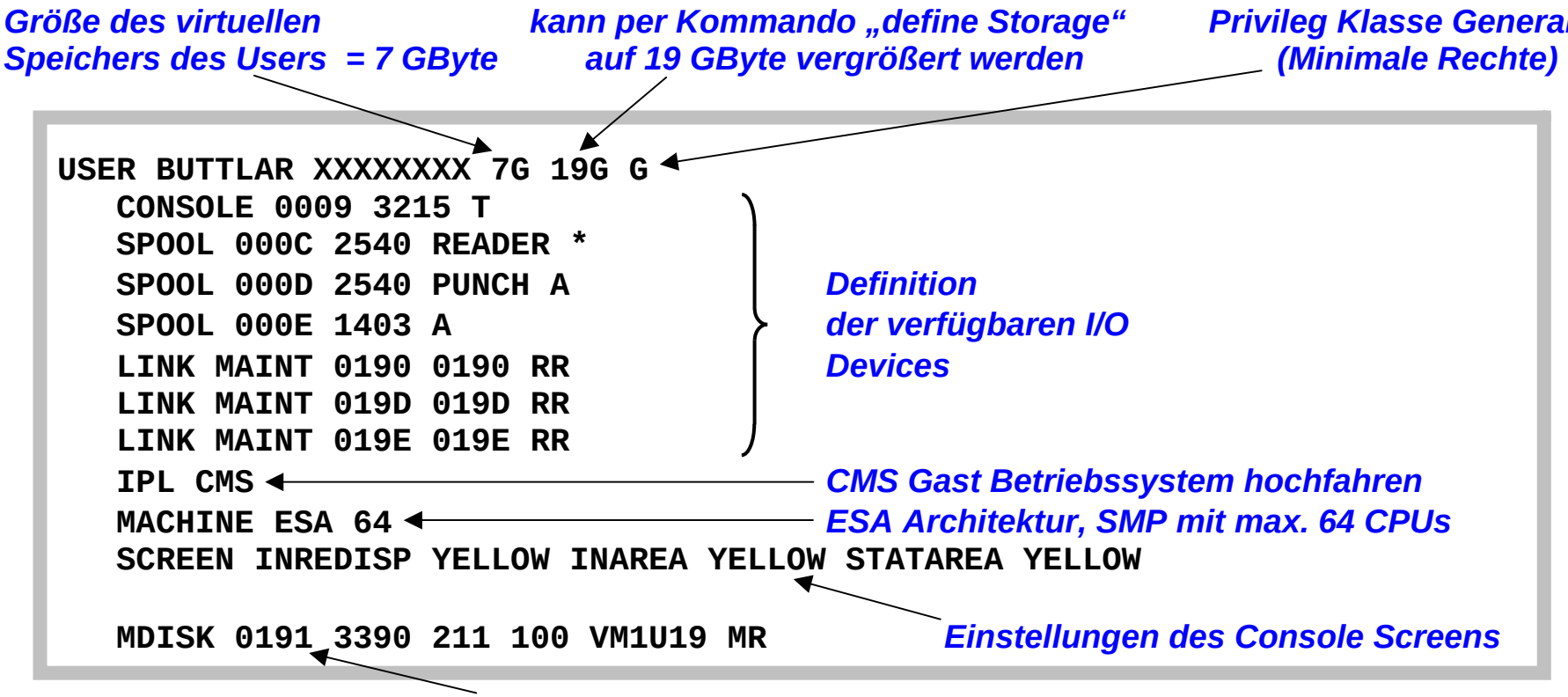
z/VM (VM/370) Betriebssystem

Für CP/67 und VM/370 wurde ein eigenes Gast Betriebssystem CMS (Conversational Monitor System) entwickelt. Dieses und alle anderen Gast Betriebssysteme (einschließlich ihrer Kernel Funktionen) laufen im Problemstatus. Privilegierte Maschinenbefehle (z.B. E/A) werden von CP abgefangen und interpretativ abgearbeitet.

CMS ist ein besonders für die Software Entwicklung ausgelegtes Einzelplatz Betriebssystem. Für 1000 gleichzeitige CMS Benutzer werden 1000 CMS Instanzen angelegt.

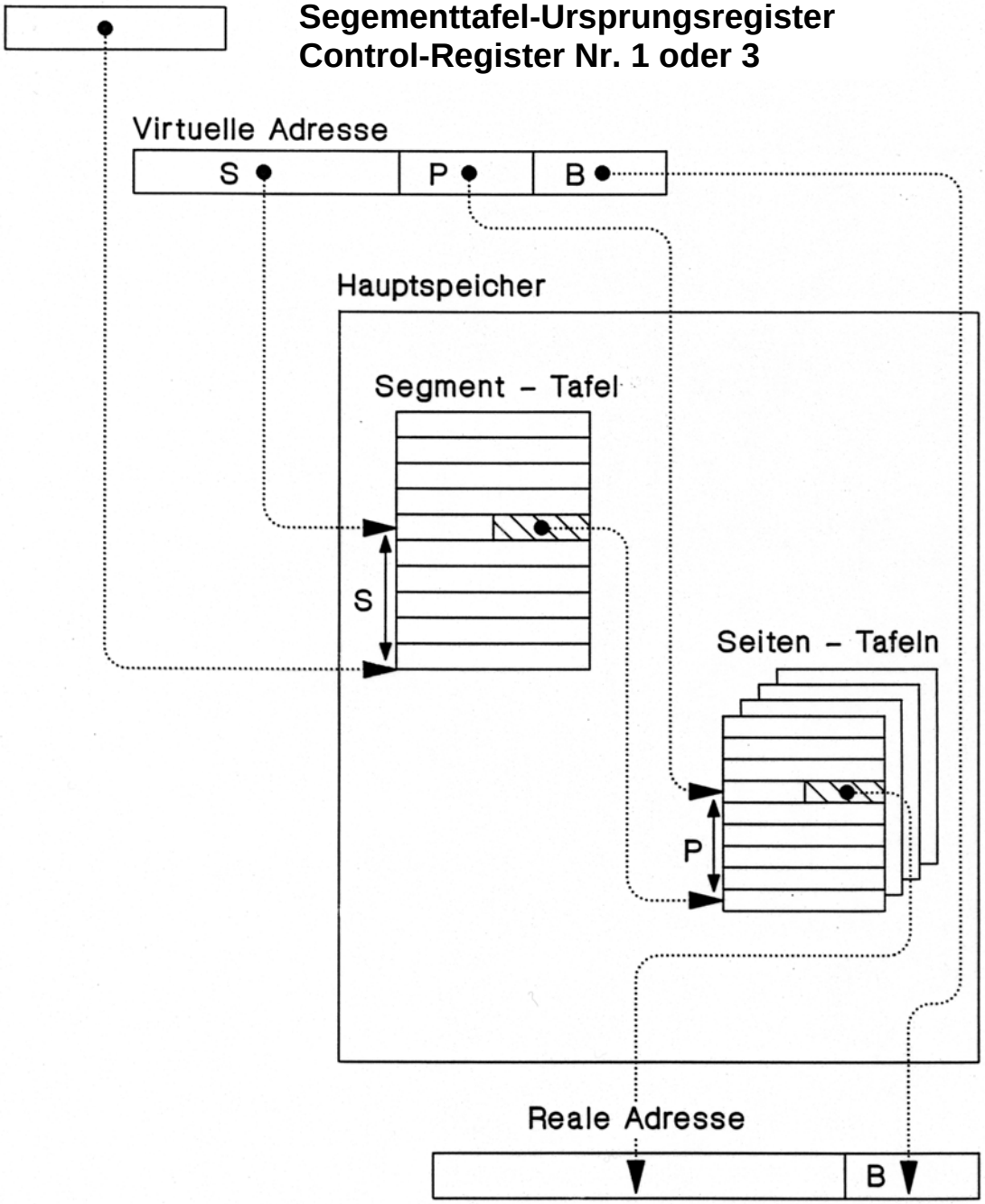
VM/370 und z/VM erreichen eine uneingeschränkte S/390 und System z Kompatibilität für alle Gast Betriebssysteme. Der Performance Verlust ist relativ gering (< 5 %).

Plattenspeicherplatz wird allen Gastbetriebssystemen in der Form virtueller „Minidisks“ statisch zugeordnet. Hauptspeicherplatz wird dynamisch verwaltet.



z/VM Welcome Screen

Gezeigt ist der Welcome Screen, mit dem sich ein CMS User in z/VM einlogged. Die Größe des virtuellen Speichers wird für jeden Benutzer eingeschränkt, um den Verwaltungsaufwand klein zu halten. Reader und Punch sind eine Erinnerung an die früher gebräuchlichen Lochkartenleser und –stanzer. Mit dem IPL (Initial Program Load) Kommando wird das CMS Gast Betriebssystem in den virtuellen Adressenraum dieses Benutzers geladen. Es steht eine Minidisk mit der Device Nr. 191 zur Verfügung. Sie ist auf einem realen Plattenspeicher vom Typ IBM 3390 abgespeichert, beginnt auf Zylinder 211 und ist 100 Zylinder groß.



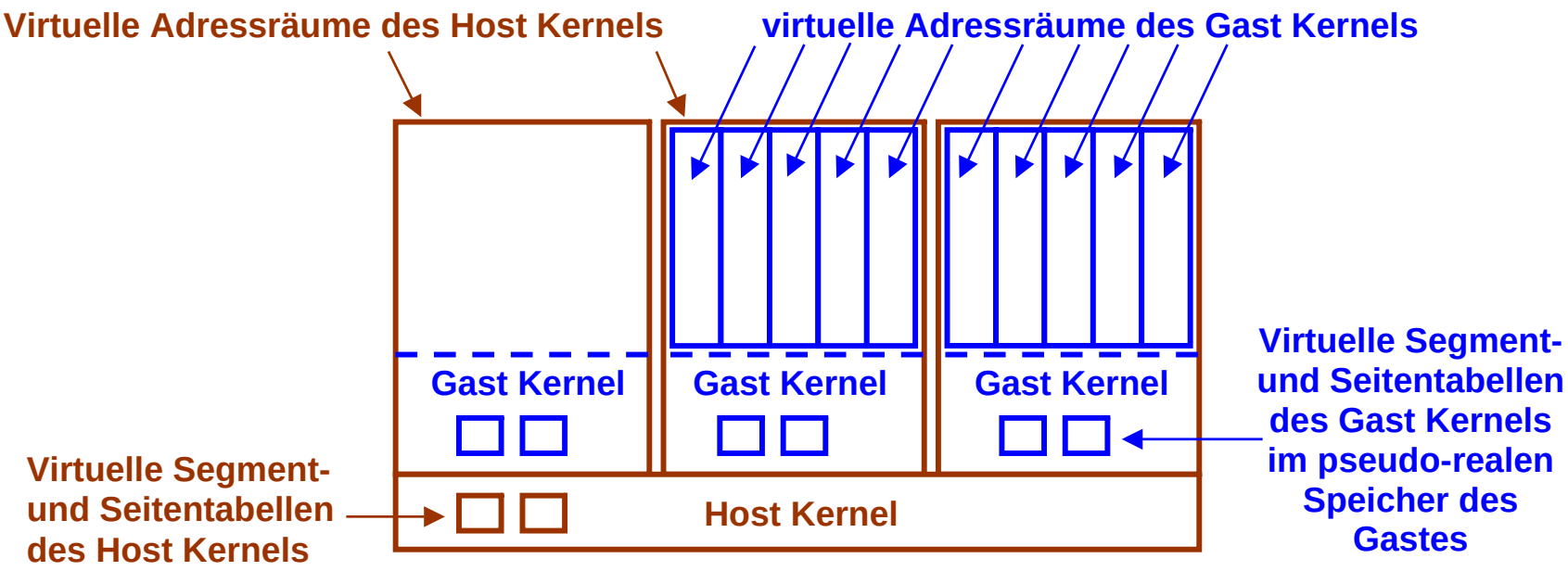
Virtuelle Adressumsetzung Pentium, S/390

Die Adressumsetzung erfolgt durch zwei Arten von Tabellen (Segment- und Seitentabelle), die im Kernel Bereich des Hauptspeichers untergebracht sind. Die Anfangsadresse der Segmenttabelle steht in einem Control Register der Zentraleinheit, z.B. CR Nr. 1 bei der S/390 Architektur.

Die Adressumsetzung erfolgt bei jedem Hauptspeicherzugriff durch Hardware mit Unterstützung durch die Segmenttabelle und eine Seitentabelle im Kernel Bereich . Sie kann durch den Programmierer nicht beeinflusst werden.

(Zur Leistungsverbesserung werden die derzeitig benutzten Adressen in einem Adressumsetzpuffer untergebracht.)

Problem: Eine Gastmaschine, die mit Virtueller Adressumsetzung konfiguriert ist, besteht darauf, ihre eigene Adressumsetzung durchzuführen.

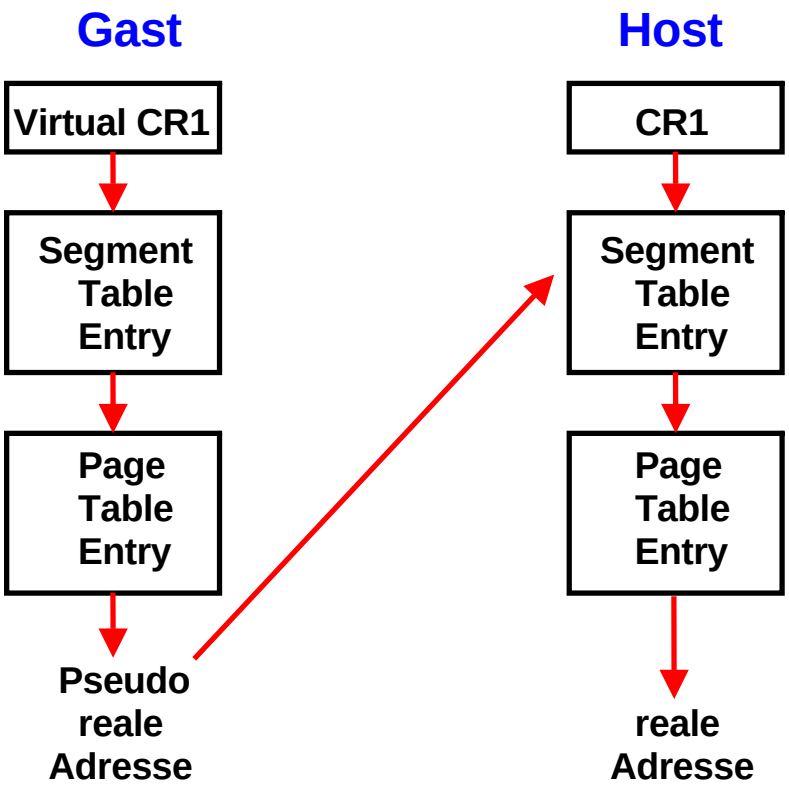


Virtuelle Maschinen mit virtueller Adressumsetzung

Angenommen, wir laden ein Gast Betriebssystem in einen (und nur einen) virtuellen Adressenraum (Address Space) des Host Betriebssystems.

Bei der Einführung von VM/370 war CMS das einzige verfügbare Gast-Betriebssystem. CMS ist ein single User Betriebssystem und verwendet keine virtuelle Adressumsetzung. Das erleichtert die Implementierung.

Praktisch alle modernen Betriebssysteme verwenden eine virtuelle Adressumsetzung. Ein solches Gast Betriebssystem glaubt, der ihm zur Verfügung gestellte virtuellen Adressenraum des Host Betriebssystems ist ein realer Speicher – hier als pseudo-realer Gastspeicher bezeichnet. Es richtet in diesem realen Gastspeicher mehrfache virtuelle Adressenräume (virtuelle Gastspeicher) ein, und will seine eigenen Prozesse in diesen virtuellen Adressräumen laufen lassen. Wir brauchen eine Virtualisierung der virtuellen Adressenräume des Gastbetriebssystems !!! .

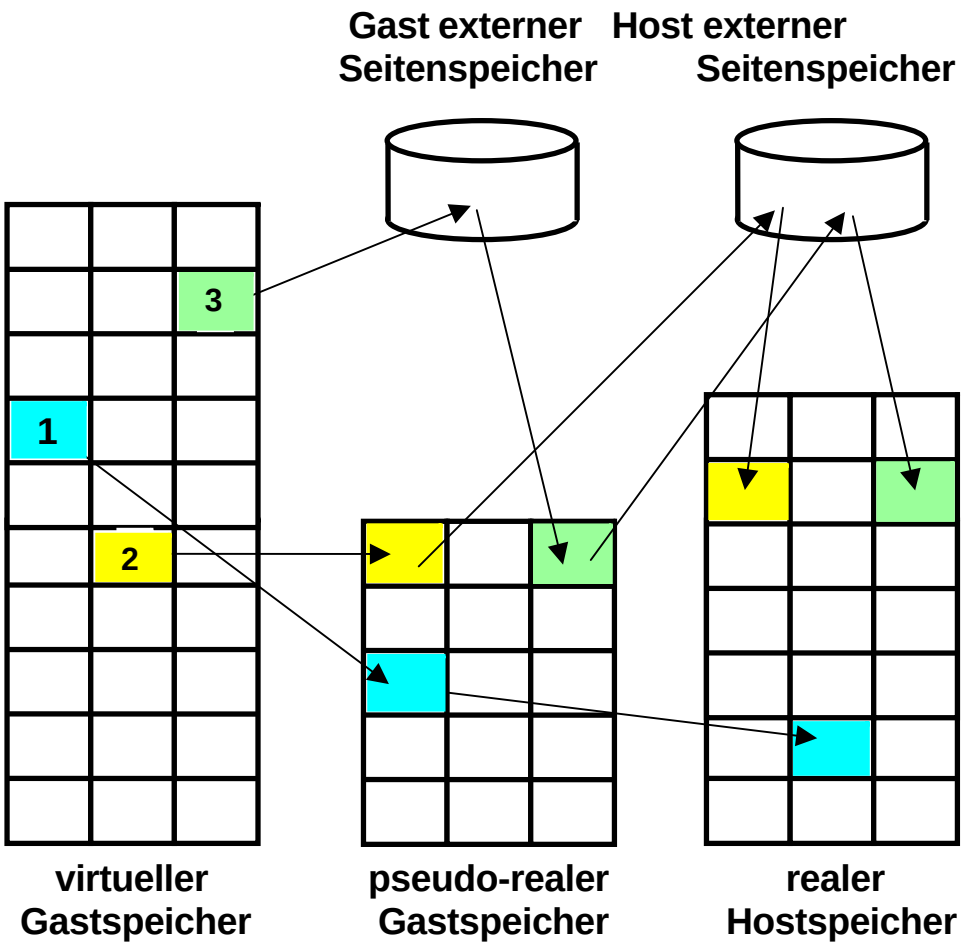


Doppelte Adress-umsetzung

Das Gastbetriebssystem nimmt bei jedem Hauptspeicherzugriff eine virtuelle→ reale Adressumsetzung vor. Es unterhält innerhalb des Gast Kernel Bereiches hierfür seine eigenen Segment- und Seitentabellen. Als Ergebnis der Adressumsetzung entsteht eine Rahmenadresse im pseudo-realen Gastspeicher.

Die Rahmenadresse im realen Gastspeicher ist aus Sicht des Host Kernels aber eine virtuelle Seitenadresse innerhalb des virtuellen Adressenraums, den der Host Kernel der Gast Maschine als pseudo-realen Speicher zur Verfügung stellt. Bei jedem Hauptspeicherzugriff benutzt der Host Kernel das Ergebnis der Adressumsetzung des Gast Kernels, um die Rahmenadresse im pseudo-realen Gastspeicher in eine echte reale Adresse zu übersetzen. Der Host Kernel verwendet für diese Adressumsetzung seine eigenen Segment- und Seitentabellen.

Da die Adressumsetzung per Hardware erfolgt, kann die doppelte Adressumsetzung nicht funktionieren !



Adressumsetzung zwischen Gast-Kernel und Host-Kernel

Drei alternative Möglichkeiten

Wie erfogt die Adressumsätzung der Seiten des virtuellen Gastspeichers in Rahmen des realen Hauptspeichers des Host System. Je nach dem Zustand der Seiten treten 3 Fälle auf:

1. Der gewünschte Rahmen befindet sich im (pseudo-)realen Speicher des Gastes und auch im realen Speicher des Rechners selbst (realer Hostspeicher).

2. Der gewünschte Rahmen befindet sich im pseudo-realen Speicher des Gastes aber nicht im realen Hostspeicher. Der Host-Kernel löst eine Fehlseitenunterbrechung aus und lädt den Rahmen in den realen Hauptspeicher.

3. Der gewünschte Rahmen befindet sich nicht im realen Speicher des Gastes. Der Gast-Kernel löst eine Fehlseiten-unterbrechung aus, die vom Host-Kernel abgefangen wird. Dieser lädt den Rahmen in den realen Hauptspeicher und gleichzeitig auch in den realen Gastspeicher.

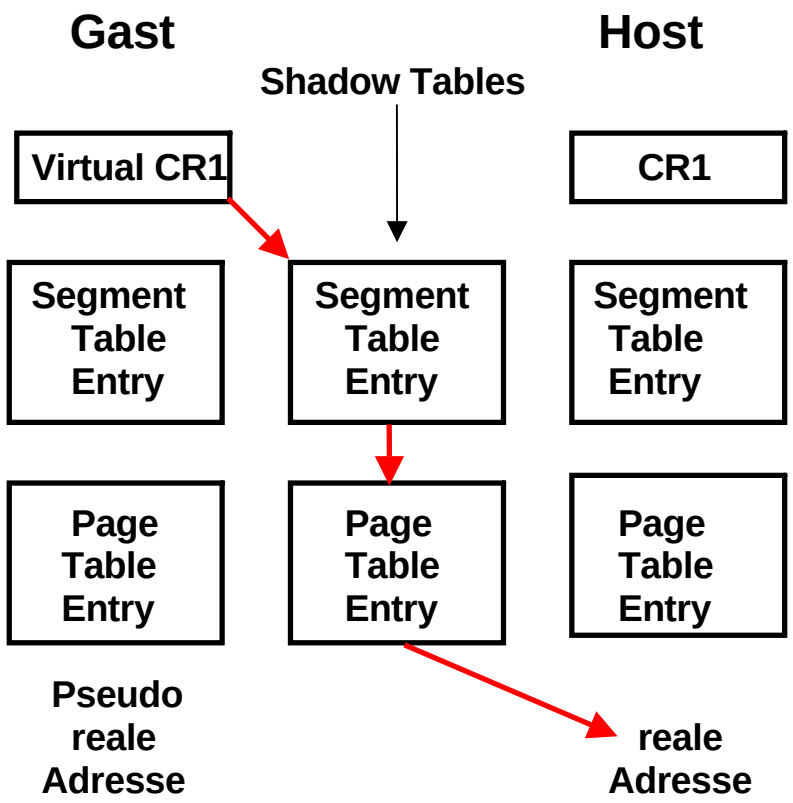
Problem der Adressübersetzung für die Virtuelle Maschine

Die virtuelle Adressumsetzung mit Hilfe von Segment- und Seitentafeln erfolgt durch Hardware und kann durch Software nicht beeinflusst werden. Das hier geschilderte Verfahren kann deshalb nicht funktionieren.

Die Anfangsadresse der Segmenttafel steht im Control Register 1 (Mainframe) oder Control Register 3 (x86 Architektur). Für die Gast Maschine ist dies ein virtuelles Control Register, welches den Pointer für die Lokation der Segmenttabelle im Gast-Kernel Bereich enthält. Die Information in dem virtuellen Control Register steht aber nicht in dem physischen Control Register, sondern wird vom Host Kernel irgendwo gespeichert. Mit diesem virtuellen Control Register kann somit die Adressumsetzungs-Hardware nichts anfangen.

Erforderlich wäre eine Änderung der Hardware Architektur, welche die hier beschrieben doppelte Adressumsetzung ermöglicht. Dies war jedoch in der ursprünglichen x86 (Pentium) Architektur nicht vorgesehen.

Zur Lösung des Problems führte man die sogenannten „Shadow Tables“ ein. Shadow Tables duplizieren teilweise die Einträge in den Segment- und Seitentabellen des Gasts und des Hosts, und müssen mit diesen im laufenden Betrieb synchron gehalten werden.



Shadow Page Tables unter VM/370 und VMware

VM/370 und VMware erstellen anhand der Gast- und der eigenen Segment- und Seitentabellen sogenannte *Shadow Tables*, die direkt die virtuellen Adressen des Gastes auf reale Adressen des Hosts abbilden. Diese Tabellen liegen im Speicherbereich des Host-Kernels.

Wenn immer eine Gast Maschine (in Wirklichkeit ein Prozess des Host Kernels) die Verfügungsgewalt über die CPU bekommt, muss der Host Kernel die Shadow Tables mit neu berechneten korrekten Werten füllen. Bei jeder Änderung in den Segment- und Seitentabellen einträgen des Gastes werden diese Änderungen in den Shadow Tables nachvollzogen. Der Vorteil des Verfahrens ist, es funktioniert. Nachteilig ist, es kostet viel Performance.

Sensitive Maschinenbefehle

Es muss verhindert werden, dass bei der Ausführung eines Maschinenbefehls innerhalb einer virtuellen Maschine das Verhalten einer anderen virtuellen Maschine beeinflusst wird.

Die Ausführung von nicht-sensitiven Maschinenbefehlen beeinflusst nicht das Verhalten einer anderen virtuellen Maschine.

Die Ausführung von sensitiven Maschinenbefehlen kann das Verhalten einer anderen virtuellen Maschine beeinflussen.

Einfachste Implementierung: Alle sensitiven Maschinenbefehle sind gleichzeitig auch privilegierte Maschinenbefehle und können nur vom Host-Kernel ausgeführt werden. So wurde VM/370 in 1971 implementiert.

Die Firma VMWare portierte VM/370 auf die x86 Plattform und lieferte ihr erstes Produkt in 1999 aus. Unglücklicher weise war die x86 Architektur hierfür wesentlich schlechter als die damalige S/370 Architektur geeignet:

Many models of Intel's machines allow user code to read registers and get the value that the privileged code put there instead of the value that the privileged code wishes the user code to see.

G.J. Popek, R.P. Goldberg: Formal Requirements for Virtualizable Third Generation Architectures. Comm. ACM, Vol. 17, Nr. 7, Juli 1974, S. 412-421.
http://www.cap-lore.com/CP.html

Probleme der x86 Architektur

Im Vergleich zu VM/370 sind die VMware ESX und GSX Server benachteiligt, weil einige kritische Eigenschaften in der x86 Architektur fehlen. Für den Betrieb von Gast-Maschinen ist es erforderlich, dass alle Maschinenbefehle, welche den privilegierten Maschinenstatus abändern oder auch nur lesen, nur im Kernel Status ausgeführt werden können.

Wenn ein Gast in ein Control Register schreibt, muss der Host Kernel diesen Maschinenbefehl abfangen, damit nicht das reale Kontrollregister des Hosts verändert wird. Der Host Kernel wird jetzt nur die Effekte der Instruktion für diesen Gast simulieren. Liest der Gast anschließend diese Kontrollregister wieder aus, so muss diese Instruktion ebenfalls abgefangen werden, damit der Gast wieder den Wert sieht, den er vorher in das Register geschrieben hat (und nicht etwa den realen Wert des Kontrollregisters, der nur für den Host sichtbar ist).

Da die x86 Architektur diese Bedingung ursprünglich nicht erfüllte, war es nicht möglich, wie unter VM/370, alle Maschinenbefehle einfach im User Mode auszuführen, und auf Programmunterbrechungen zu vertrauen wenn auf privilegierten Maschinenstatus Information zugegriffen wird.

VMware's ESX Server überschreibt hierzu dynamisch Teile des Gast-Kernels und schiebt Unterbrechungs-bedingungen dort ein, wo eine Intervention des Host-Kernels erforderlich ist. Als Folge hiervon tritt ein deutlicher Leistungsverlust auf, besonders bei Ein-/Ausgabe-Operationen. Manche Funktionen sind nicht vorhanden oder können nicht genutzt werden.

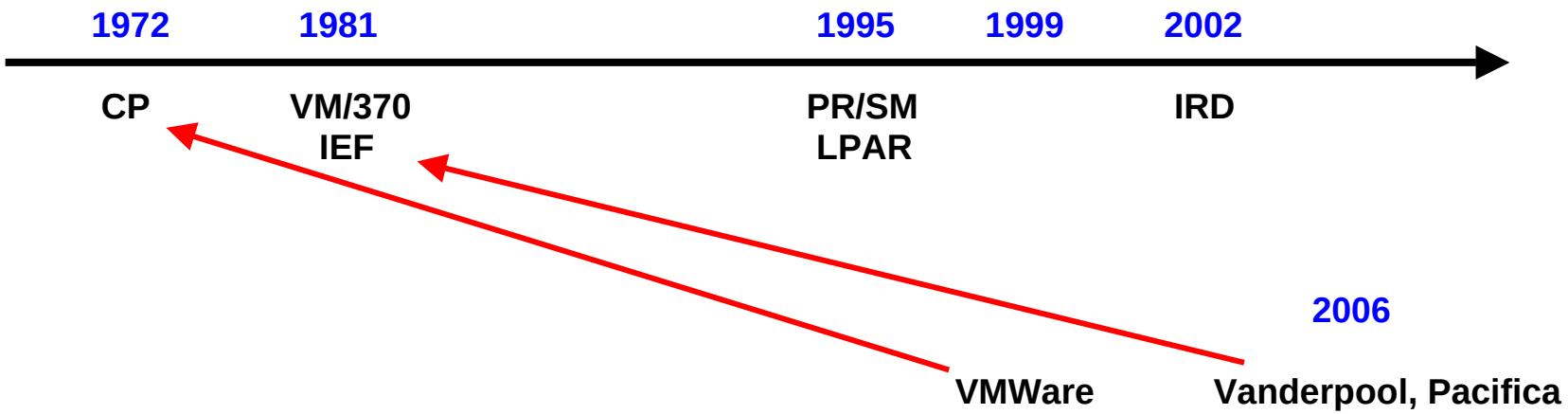
Kompatibilitätsprobleme treten auf; es kann sein, dass bestimmte Anwendungen nicht lauffähig sind.

Paravirtualization.

Hierbei ist die Betriebssystem-Architektur der virtuellen Maschine nicht vollständig mit der Host Betriebssystem-Architektur identisch. Die Benutzerschnittstelle (Application Binary Interface, ABI) ist die gleiche. Der Gast-Kernel unterstellt jedoch Abweichungen zu der x86 Architektur. Dies verbessert das Leistungsverhalten, erfordert aber Änderungen des Gast-Kernels. Hiervon ist nur ein sehr kleiner Teil des Kernel-Codes betroffen. Beispielsweise existiert ein funktionsfähiger Linux-Port (XenoLinux), dessen ABI mit dem eines nicht-virtualisierten Linux identisch ist.

Ein ähnlicher Ansatz wird von *Denali verfolgt*. Als Gast-Betriebssystem dient *Ilwaco*, eine speziell an den Denali Hypervisor angepasste BSD Version. Denali unterstützt nicht das vollständige x86 ABI. Es wurde für Netzwerk-Anwendungen entwickelt und unterstellt Einzelbenutzer-Anwendungen. Mehrfache virtuelle Adressräume sind unter Ilwaco nicht möglich.

Die Entwicklung der Virtualisierung



Das VM/370 Betriebssystem mit dem CP Hypervisor und dem CMS Gast-Betriebssystem ist seit 1972 verfügbar. 1995 gründeten einige VM/370 Entwickler die Firma VMware. Seit 1999 ist VMWare für die x86 Plattform verfügbar.

Bereits 1981 hatte IBM für alle S/370 Rechner eine Erweiterung der Hardware Architektur eingeführt, die als „Interpretive Execution Facility“ (IEF) bezeichnet wurde, und zu einer erheblichen Leistungsverbesserung der virtuellen Maschinen führte. Eine vergleichbare Architekturerweiterung wurde von Intel und AMD im Jahre 2006 für die x86 (Pentium) Architektur unter dem Namen Vanderpool bzw. Pacifica eingeführt.

VT-x (Virtualisation Technology for x86) ist eine alternative Bezeichnung für Vanderpool. Eine ähnliche Einrichtung für Itanium hat den Namen VT-i . AMD Virtualization, abgekürzt AMD-V ist eine alternative Bezeichnung für Pacifica.

Bereits vorher im Jahre 1995 hatte die S/390 Architektur eine weitere grundlegende Verbesserung unter dem Namen „Logische Partition“ (LPAR) eingeführt. Dies wurde 2002 durch eine nochmalige Verbesserung, den „Intelligent Resource Director“ (IRD) ergänzt. Der PowerPC führte LPARs 2002 ein.

Experten gehen davon aus, dass x86 weitere 10 Jahre benötigen wird, um eine mit LPAR und IRD vergleichbare Funktionalität zu erreichen.

Host- und Gast Modus

Jeder Rechner unterscheidet zwischen dem Kernel Modus und dem User Modus. Bestimmte sog. „Privilegierte Maschinenbefehle“ können im Kernel Modus aber nicht im User Modus ausgeführt werden.

Zur Lösung von Virtualisierungs-Schwierigkeiten führte man zusätzlich einen Host- bzw. Gast Modus als Architekturerweiterung ein. Mainframes bezeichnen diese Erweiterung als „Interpretive Execution Facility“ (IEF). Die entsprechenden Bezeichnungen sind VT-x (Vanderpool) bei Intel bzw. AMD-V (Pacifica) bei AMD. Obwohl die drei Ansätze sehr ähnlich sind, unterscheiden sie sich in einigen Details. Besonders VT-x und AMD-V sind nicht miteinander kompatibel.

	Kernel Modus	User Modus
Host Modus	+	–
Gast Modus	+	+

Mögliche Zustände beim Einsatz des Host/Gast Modus

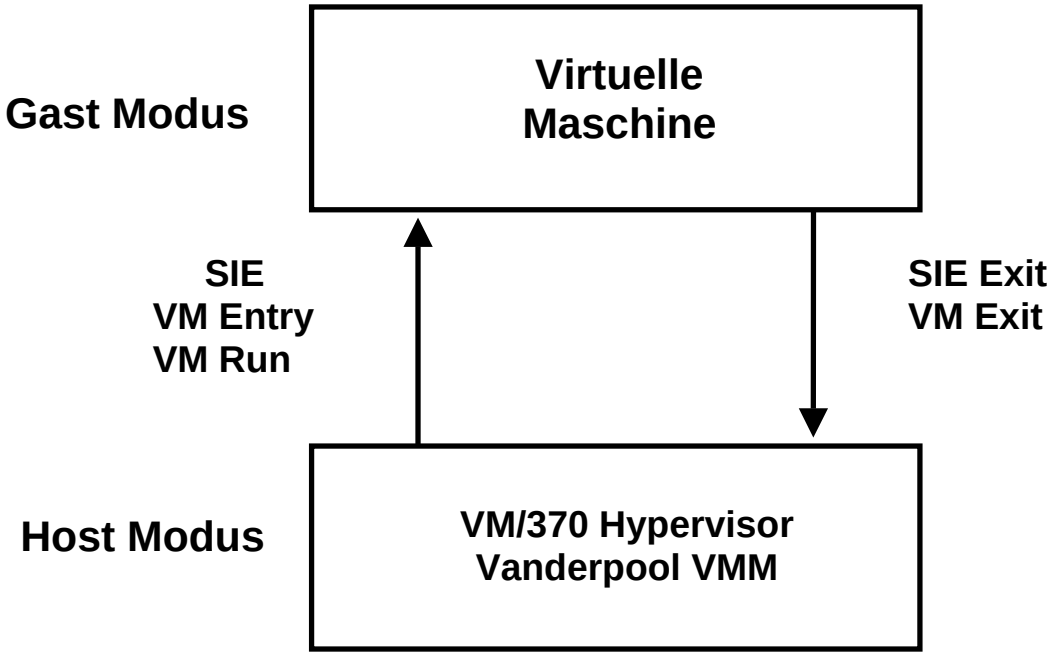
Jeder Rechner unterscheidet zwischen Kernel Modus (supervisor state) und User Modus (problem state). Das unterschiedliche Verhalten wird durch ein Bit im Status Register der CPU (Program Status Word, PSW) definiert.

Interpretive Execution Facility, Vanderpool und Pacifica führten zusätzlich noch eine Unterscheidung Host Modus/Gast/Modus mit Hilfe eines zweiten Bits im Statusregister ein. Damit sind die Kombinationen Host Modus/Kernel Modus. Gast Modus /Kernel Modus und Gast Modus / User Modus möglich. Der Hypervisor läuft grundsätzlich im Host Modus; die Kombination Host Modus / User Modus hat in Bezug auf virtuelle Maschinen keine Bedeutung.

Das Problem der sensitiven Maschinenbefehle kann nun dadurch gelöst werden, dass die Hardware sich im Gast Modus anders verhält als im Host Modus. Auch kann die Gast Maschine in ihrem Kernel Modus solche priviligierten Maschinenbefehle direkt selbst ausführen, die nicht sensitiv sind – sie braucht hierfür nicht mehr die Unterstützung des Host Kernels.

Vanderpool bezeichnet den Host Kernel Modus, in dem der Host Kernel (Virtual Machine Monitor, VMM) ausgeführt wird, als *VMX Root Operation*. Jeder Gast läuft in *VMX non-Root Operation.*

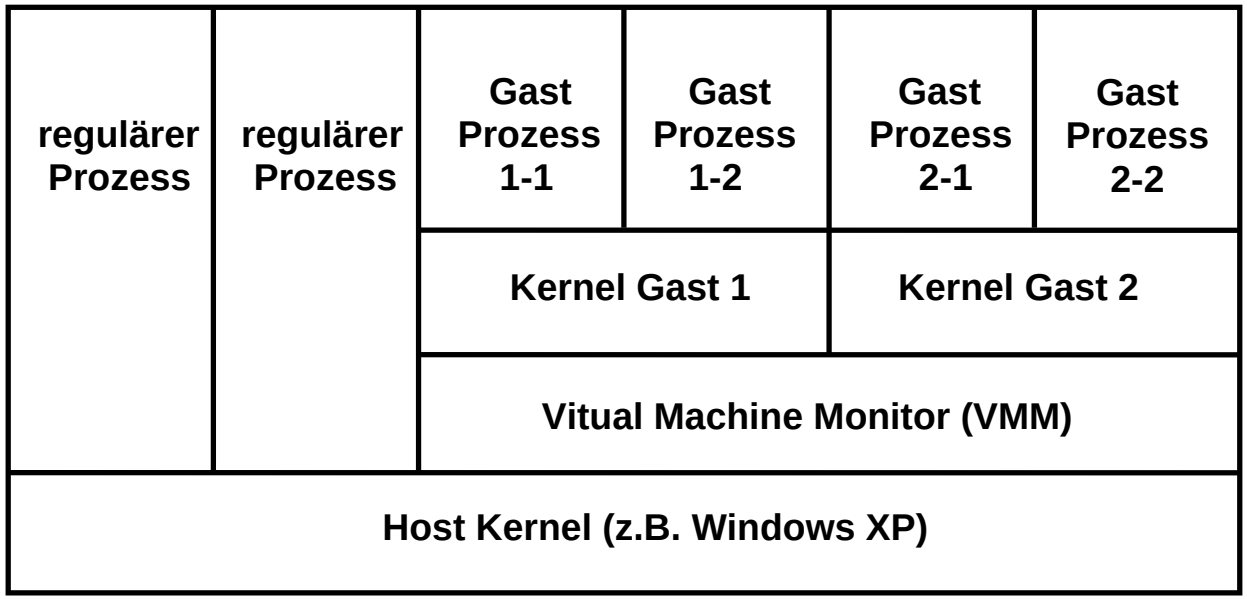
Der Aufruf eines Gastes erfolgt durch einen VM Entry Befehl (VM Run bei AMD VT, SIE bei VM/370). Die Rückkehr zum VMM erfolgt durch einen VM Exit Befehl.



Im Gastmodus können privilegierte, aber nicht sensitive Maschininenbefehle vom Kernel des Gast Betriebssystems abgearbeitet werden. Das Problem der sensitiven nicht-privilegierten x86 Architektur der x86 Architektur wird durch zusätzliche Vanderpool bzw. Pacifica Hardware gelöst, die diese Befehle bei ihrer Dekodierung identifiziert und zwecks Ausführung an den VMM übergibt.

Der Kernel des VM/370 Betriebssystems ist gleichzeitig der Hypervisor. Der Rechner läuft immer im Virtualisierungsmodus.

Der Vanderpool Virtual Maschine Monitor (VMM) ist eine Erweiterung zu einem vorhandenen Betriebssystem Kernel, z.B. dem Windows XP Kernel. Beide zusammen bilden den Hypervisor. Anwendungen können deshalb wahlweise im Virtualisierungs-modus unter einem Gastbetriebssystem oder nicht virtualsiert unmittelbar unter dem Host Betriebssystem laufen



Mögliche Vanderpool Konfiguration

Der Rechner befindet sich entweder im Virtualisierungsmodus oder auch nicht. Im Virtualisierungsmodus ist er in der Lage „Virtual Machine Execution“ (VMX) Operationen durchzuführen und mit Gast Maschinen zu arbeiten. Ein VMXON Befehl versetzt den Rechner in den Virtualisierungsmodus und aktiviert den Virtual Machine Monitor; mit VMXOFF wird der Virtualisierungsmodus wieder verlassen.

Existierende Virtual Machine Monitore wie VMware, KVM, XEN, Virtual PC, Virtual Box usw. nutzen die Vanderpool bzw. Pacifica Technologie. Damit entfallen die meisten bisherigen x86 Probleme.

Shadow Page tables

Mit der Einführung der Interpretive Execution Facility, Vanderpool bzw. Pacifica kann die Recher Hardware für die Bedürfnisse der Virtuellen Maschinen angepasst und erweitert werden. Spezifisch ist damit die Einführung einer doppelten Adressumsetzung und die Eliminierung von Shadow Page Tables möglich.

Hiermit werden die häufigen Updates der Shadow Page Tables eliminiert. Nachteilig sind die zusätzlichen Zugriffe auf Segment- und Pagetables im Hauptspeicher. Der Aufwand hierfür ist in der Regel aber nicht so groß wie der Aufwand für die Shadow Page Table Updates. Der Grund ist, die am häufigsten benutzten einträge in den Segment- und Seitentabellen werden in einem als DLAT (Directory Look Aside Table) oder TLB (Translation Lookaside Buffer) bezeichneten Cache Speicher abgelegt. Wenn man diesen ausreichend dimensioniert (Tausende von Einträgen), ist der Leistungsabfall vernachlässigbar.