

# Paravirtualisierung (2)

Dr.-Ing. Volkmar Sieh

Department Informatik 4  
Verteilte Systeme und Betriebssysteme  
Friedrich-Alexander-Universität Erlangen-Nürnberg

WS 2019/2020



- User-Mode-Gast-Kernels brauchen
  - nur wenige System-Calls
  - spezielle System-Callsvom Gastgeber-OS
- Neben den virtuellen Rechnern laufen meist keine normalen Applikationen auf einem Gastgeber  
=> Gastgeber-OS abspecken/spezialisieren



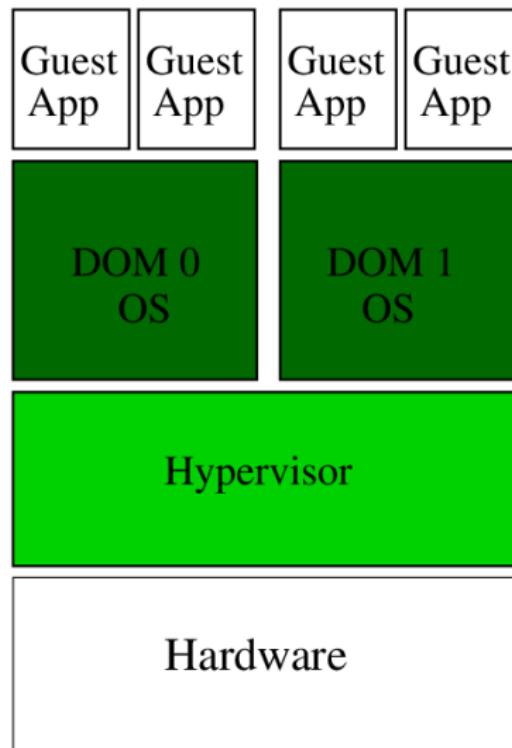
Definitionen:

Abgespecktes Gastgeber-OS: „**Hypervisor**“

Modifizierte Gast-OS mit ihren Gast-Applikationen: „**Domains**“



# Paravirtualisierung (2) – Hypervisor



Gebraucht werden i.a. nur folgende Hypervisor-Calls:

- Ersatz für privilegierte CPU-Instruktionen:
  - Page-Tabellen verwalten
  - Segment-Tabellen verwalten
  - Exception-/Interrupt-/System-Call-Verwaltung
  - Ein-/Ausgabe
- zusätzlich:
  - VMs verwalten (konfigurieren, starten, stoppen)

=> Hypervisor (abgesehen von Geräte-Treibern) nur einige KByte groß.



Hypervisor-Calls Beispiel Xen:

```
set_trap_table mmu_update set_gdt stack_switch
set_callbacks fpu_taskswitch platform_op set_debugreg
    get_debugreg update_descriptor memory_op multicall
update_va_mapping set_timer_op xen_version console_io
grant_table_op vm_assist update_va_mapping_otherdomain
    iret vcpu_op mmuext_op xsm_op nmi_op sched_op
callback_op xenoprof_op event_channel_op physdev_op
    hvm_op sysctl domctl kexec_op tmem_op
```



Beobachtung:

Häufig kommen mehrere privilegierte CPU-Instruktionen kurz hintereinander.

=> viele Hypervisor-Calls in kurzen Abständen

Idee:

mehrere Hypervisor-Calls zu einem zusammenfassen („multicall“)



## Paravirtualisierung (2) – Memory

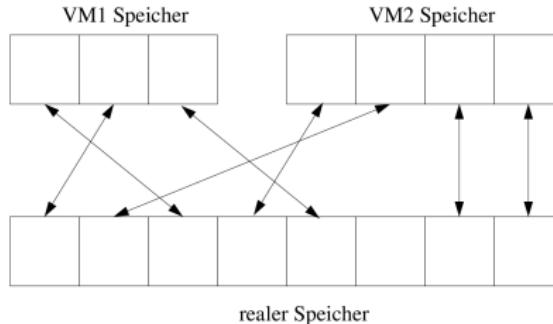
Virtuelle Maschinen sollen nur soviel Speicher reservieren, wie sie gerade wirklich benötigen („**balloon memory**“).

=> fragmentierter Speicher

Betriebssysteme gehen i.A. von kontinuierlichem Speicher aus.

=> Umrechnungstabellen notwendig

Umrechnung erfolgt beim Eintragen der Page-Nummern in die Page-Tabellen.



Hypervisor muss sicherstellen, dass Page-Tabellen korrekt sind. Andernfalls wäre Zugriff einer VM auf den Speicher einer anderen möglich.

Möglichkeiten:

- Ändern der Page-Tabellen über Hypervisor-Calls
- Überprüfung der Tabelle bevor das Page-Tabellen-Basis-Register (%cr3) darauf umgesetzt wird. Page-Tabelle ab da read-only. Beim Schreiben auf die Tabelle wird Tabellen-Unterbaum abgeschnitten und wieder read-write. Anhängen des Unterbaumes beim nächsten Page-Fault bzw. beim Schreiben weiterer Page-Tabellen-Einträge (nach Kontrolle).
- Guest ändert nur „Schatten-Tabellen“. Die echten Tabellen passt der Host an
  - bei Page-Faults (Eintragen neuer Pages),
  - bei Hypervisor-Calls (Austragen alter Pages).



Drei Möglichkeiten für I/O:

- Hypervisor bietet System-Calls für I/O; VMs nutzen diese
- Hypervisor übergibt Kontrolle von Geräten an VMs
- Hypervisor übergibt Kontrolle von Geräten an *eine* VM; alle anderen VMs machen I/O über diese



„Hypervisor bietet System-Calls für I/O“:

- Vorteile:**
  - Gast-OS unabhängig von I/O-Hardware
  - Geräte können von mehreren Gast-OS geshared werden.
- Nachteile:**
  - Hypervisor u.U. sehr groß.
  - Performance-Verluste beim Übertragen großer Datenmengen Hypervisor <=> Gast-OS.

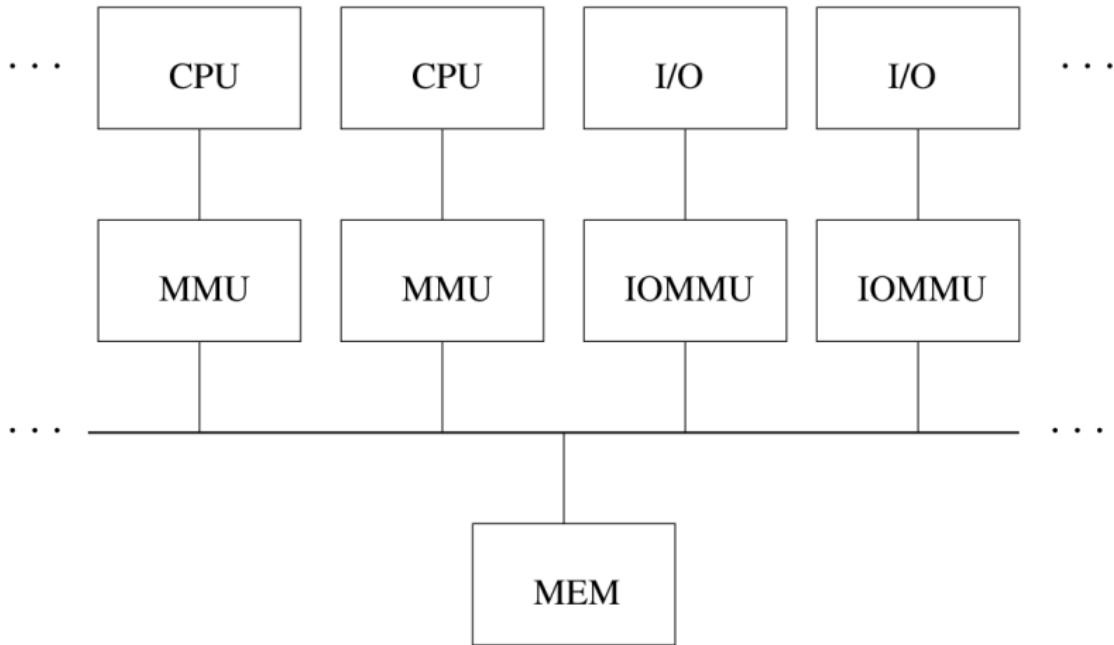


„Hypervisor übergibt Kontrolle von Geräten an VMs“:

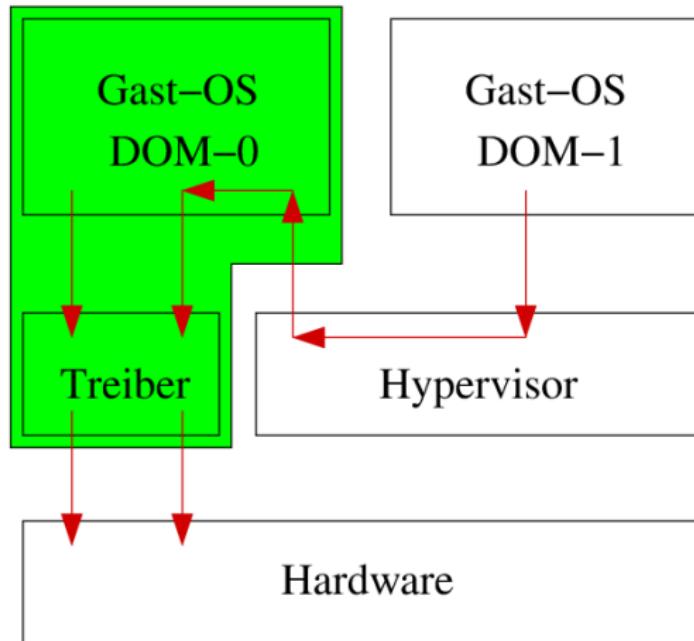
- Vorteile:**
  - Optimale Performance
  - Hypervisor muss keine Gerätetreiber beinhalten; Hypervisor sehr portabel und sehr klein.
  - Native-OS hat sowieso entsprechende Treiber.
- Nachteile:**
  - Bei falscher DMA-Programmierung können Gast-OS sich gegenseitig stören (=> IOMMU).
  - Geräte können nicht geshared werden.



## Paravirtualisierung (2) – IO MMU



„Hypervisor übergibt Kontrolle von Geräten an *eine* VM“:



„Hypervisor übergibt Kontrolle von Geräten an *eine* VM“:

- Vorteile:**
- Hypervisor muss keine Gerätetreiber beinhalten; Hypervisor sehr portabel und sehr klein.
  - Native-OS hat sowieso entsprechende Treiber.
  - Geräte können geshared werden.

- Nachteile:**
- Performance-Verluste beim Übertragen der Daten zur/von der ausgezeichneten VM.



Domain 0: Domain mit I/O-Permissions

- muss die zuerst gestartete Domain sein
- kann weitere Domains starten

Domain U: alle anderen Domains



Problem: Transfer der Daten muss *sehr* schnell geschehen.

Lösung: Daten werden nicht kopiert:

- Schreiben:
  - Domain-U schreibt Daten in einen Puffer (eine Page).
  - Physikalische Adresse der Page wird mit dem Schreibauftrag über den Hypervisor an Domain-0 weitergereicht.
  - Domain-0 nutzt physikalische Page zum Schreiben per DMA zum Gerät.
  - Domain-0 schickt „fertig“-Meldung zurück.
- Lesen:
  - Domain-U schickt physikalische Adresse einer freien Page mit einem Leseauftrag über den Hypervisor zur Domain-0.
  - Domain-0 liest per DMA Daten vom Gerät in die Page ein und schickt „fertig“-Meldung zurück.
  - Domain-U kann Daten aus Page verwenden.



## Paravirtualisierung (2) – I/O

