Cloud Infrastructure

- Enabling Technologies
  - Virtualisierung
  - SOA
  - Web Services / RESTful Services

  - Siehe auch weitere pdfs zu den 3 Themen

Virtualisierung

- Definition
  - Virtualisierung ist eine Abstraktion: Logische Systeme werden von der physischen Implementierung abstrahiert. Ressourcen werden dabei nicht dediziert, sondern gemeinsam genutzt.
  - Eine logische Schicht wird zwischen Anwender und Ressource eingeführt, um die physischen Gegebenheiten z.B. der Hardware zu verstecken.
  - Die intelligente Zuordnung und Verwaltung der Ressourcen ist eine wichtige Funktionalität innerhalb der Virtualisierung.
- Virtualisierung spielt auf mehreren Ebenen der IT eine wichtige Rolle:
  - Anwendungen und Geschäftsprozesse
  - Middleware
  - IT-Infrastruktur
  - Bausteine: Server, Storage, Netze, Betriebssysteme

Virtualisierungskonzepte

- Betriebssystemvirtualisierung
- Plattformvirtualisierung
  - Vollständige Virtualisierung
  - Paravirtualisierung
- Speichervirtualisierung
  - SAN Pools
- Netzwerkvirtualisierung
  - Für Load Balancing und Failover
  - Virtuelle lokale Netze (VLANs)
- Anwendungsvirtualisierung
  - Hosted Applications oder Virtual Appliances

Virtualisierungsklassen

- Generell lassen sich drei Klassen der Virtualisierung ableiten:
  - Partitionierung
    - Aufteilung einzelner physischer Systeme in mehrere logische Systeme
  - Aggregation
    - Verbindung mehrerer physischer Systeme zu größeren logischen Systemen
  - Emulation
    - Abbildung unterschiedlicher Systemarchitekturen aufeinander

Server Virtualisierung

- Motivation
  - Flexiblere Zuteilung von Ressourcen
  - Erhöhung der Auslastung, Serverkonsolidierung
  - Bessere Verfügbarkeit, Skalierung
  - Isolation, Crash betrifft nur virtuelle Maschine
  - Erhöhung der Kompatibilität, standardisierte virtuelle Hardware
  - Kosten für die Hardware-Investition, Software-Lizenzen und Wartung optimiert, Senkung der Total Cost of Ownership (TCO)

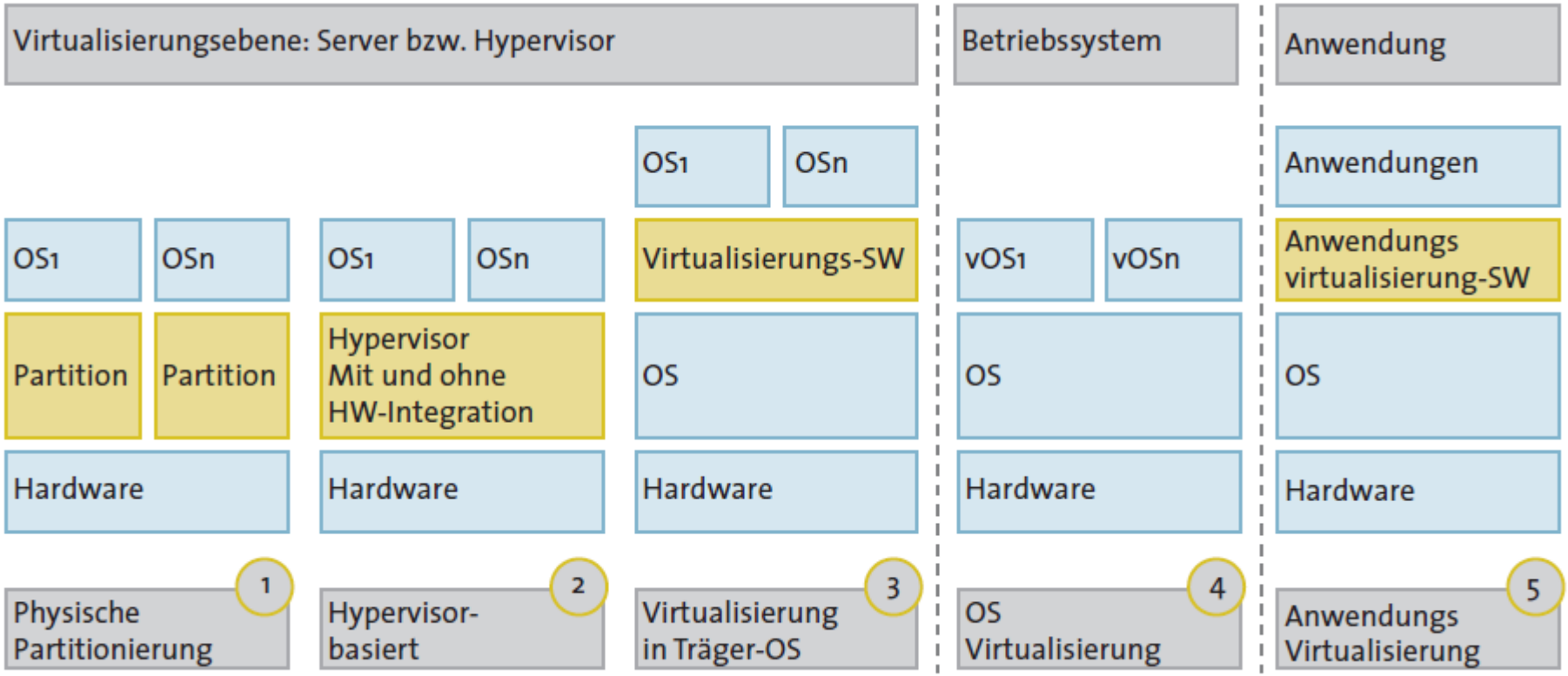
Server Virtualisierung

- Motivation cont.
  - Administration der Umgebungen wird vereinfacht
  - Vereinfachung von Backup & Recovery, ein File für einen Server, Möglichkeit von Snapshots
  - Test und Debuggen von Kernel-Modifikationen ohne ständiges Rebooten
  - Private Server auf einem geteilten Rechner, Verwendung in der Lehre
  - Generell leichtes und sicheres Experimentieren mit Betriebssystemen

Virtuelle Maschinen

- Trennung der Funktion Multi-Programming von den übrigen Funktionen eines Betriebssystems
- Emulation der Hardware zu so genannten virtuellen Maschinen (VM)
- Verschiedene Betriebssysteme können gleichzeitig auf den virtuellen Maschinen laufen
- Die virtuellen Maschinen werden durch ein Monitorprogramm (virtual machine monitor, VMM) bzw. einen Hypervisor voneinander getrennt

Virtualisierungsarchitekturen



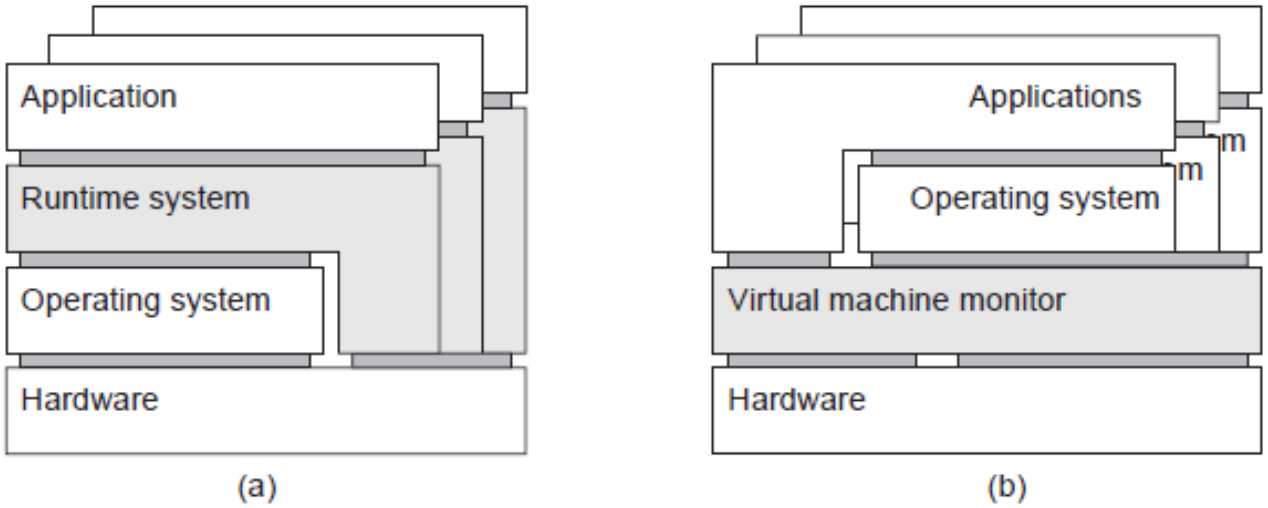
Virtualisierungsarchitekturen Details

Kriterium	Physische Partitio-nierung	Hypervisor-basierende Partitio-nierung	Virtualisie-rung innerhalb eines Träger-Betrieb-systems	Betriebs-system-Virtua-lisierung	Applikations-virtuali-sierung
Varianten	Cluster Computing	Mikropartitionen Paravirtualisie-rung	Emulation	Workload Management Emulation	Java, .NET basierte Virtual Machines Anwendungsum-gebungen
Virtualisie-rungsschicht	Serverebene (pas-siver Hypervisor)	Serverebene (akti-ver Hypervisor)	Serverebene (OS basierend)	OS-Ebene	Anwendungs-ebene
Granularität	Baugruppen	Ressource/ Subressource[1]	Ressource/ Subressource[1]	Ressource/ Subressource[1]	Subressource
Zuordnung Ressourcen	1:1	Pool[1]	Pool[1]	Pool[1]	Pool
Isolation	sehr hoch (HW)	Hoch[1]	Mittel	Gering	Gering
Dynamik	Ja[1]	Ja[1]	Ja[1]	Ja[1]	Nein
Mobilität	Nein	Ja[1]	Ja[1]	Ja[1]	Nein
Verschiedene Gast-OS	Ja	Ja	Ja	Nein	Nein

Virtualisierungsarchitekturen Beispiele

Kriterium	Physische Partitio-nierung	Hypervisor-basierende Partitio-nierung	Virtualisie-rung innerhalb eines Träger-Betrieb-systems	Betriebs-system-Virtua-lisierung	Applikations-virtuali-sierung
Beispiele/ Plattformen	■ Bull NovaS-cale: FAME ■ Fujitsu-Sie-mens Primepo-wer: PPAR ■ HP 9000/Inte-grity Server: nPAR ■ Sun Fire Ser-ver: System Domains ■ Sun Dyna-mic System Domains ■ Unisys Cellular Multiproces-sing	■ Citrix / Open-Source: XEN ■ IBM Systems: LPAR/DLPAR/ Micropartitions ■ IBM: z/VM ■ Microsoft Hyper-V ■ Sun Logical Domains (LDOMs) ■ Sun xVM Server ■ VMware ESX	■ Microsoft Vir-tual Server ■ Sun xVM Virtualbox ■ VMware Server	■ FreeBSD Jails ■ IBM AIX: WPAR ■ IBM PowerVM Lx86 ■ SUN Solaris: Container	■ Citrix XenApp ■ Java Virtual Machines ■ .Net Common Language Runtime ■ Microsoft AppV ■ Symantec Altiris SVS ■ Vmware ThinApp

Process VMs versus VM Monitors



- Process VM: A program is compiled to intermediate (portable) code, which is then executed by a runtime system (Example: Java VM).
- VM Monitor: A separate software layer mimics the instruction set of hardware $\Rightarrow$ a complete operating system and its applications can be supported (Example: VMware, VirtualBox).

Terminology Bootstrap

- Virtual Machine Extensions (VMX)
- Virtual Machine Monitor (VMM)
- VMX Root operation
  - VMM, host VM
- Management VM
  - dom0
- VMX Non-root operation
  - domU, guest VM

Vendor technologies

	Intel	AMD
CPU Flag	Virtual Machine Extensions (VMX)	Secure Virtual Machine (SVM)
Processor emulation	VT-x	AMD-v
Extended page tables	Extended page tables (EPT)	Rapid Virtualization Indexing (RVI)
MMU emulation	VT-d	AMD-Vi
Network emulation	VT-c	
PCI emulation	PCI-SIG I/O Virtualization	PCI-SIG I/O Virtualization

We will be focused on Intel VT-x

VMM Types

Type 1. “bare metal” hypervisors run directly on the host hardware

- guest OS runs at level above the hypervisor

Type 2. “hosted” hypervisors run on top of an OS

- The hypervisor layer exists as distinct second software level
- Guest operating systems run at the third level above the hardware

Popek and Goldberg Virtualization Criterion

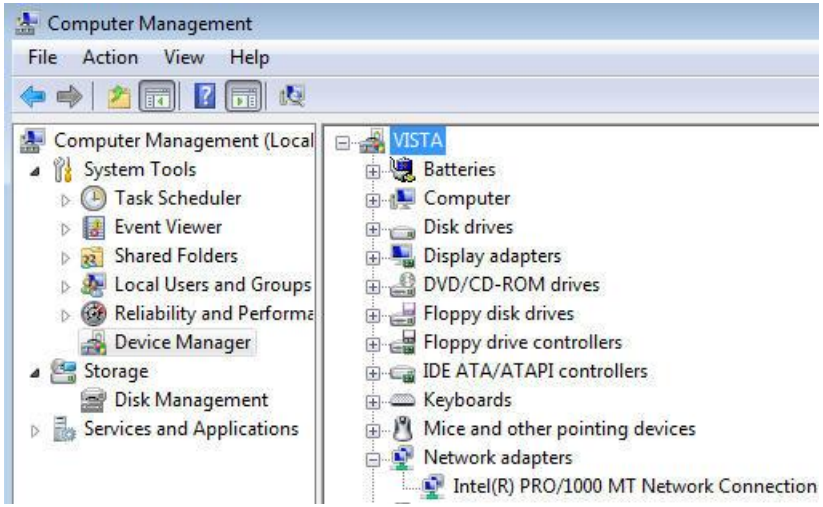
- POPEK, G. J., GOLDBERG, R. P., “Formal requirements for virtualizable third generation architectures,” ACM Communications, July **1974**
- Equivalence / Fidelity
  - A program running under the VMM should exhibit a behavior essentially identical to that demonstrated when running on an equivalent machine directly.
- Resource control / Safety
  - The VMM must be in complete control of the virtualized resources.
- Efficiency / Performance
  - A statistically dominant fraction of machine instructions must be executed without VMM intervention.

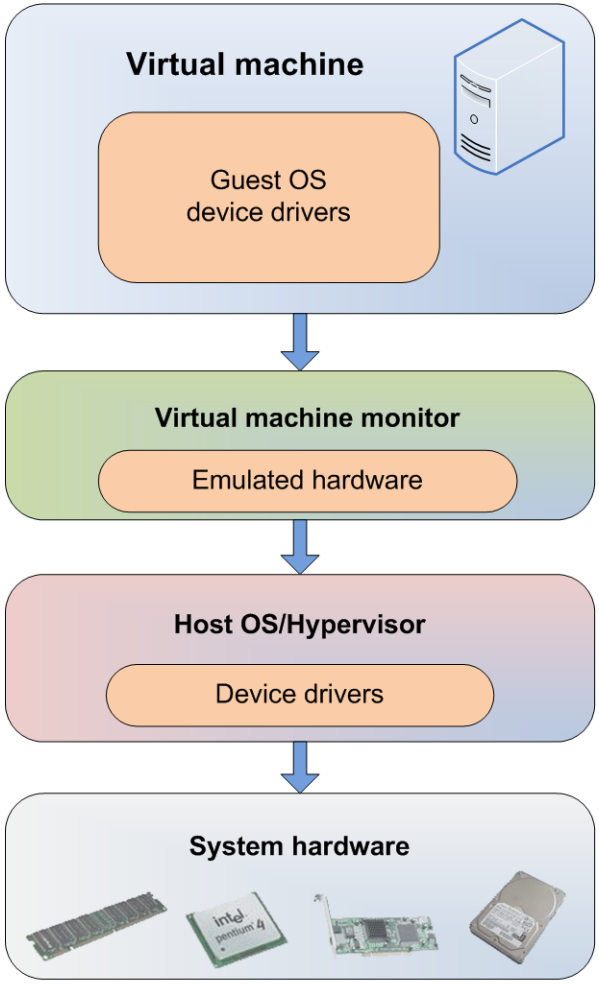
Three Virtualization Approaches

- Full Virtualization
- Paravirtualization
- Hardware-assisted Virtualization

Full Virtualization

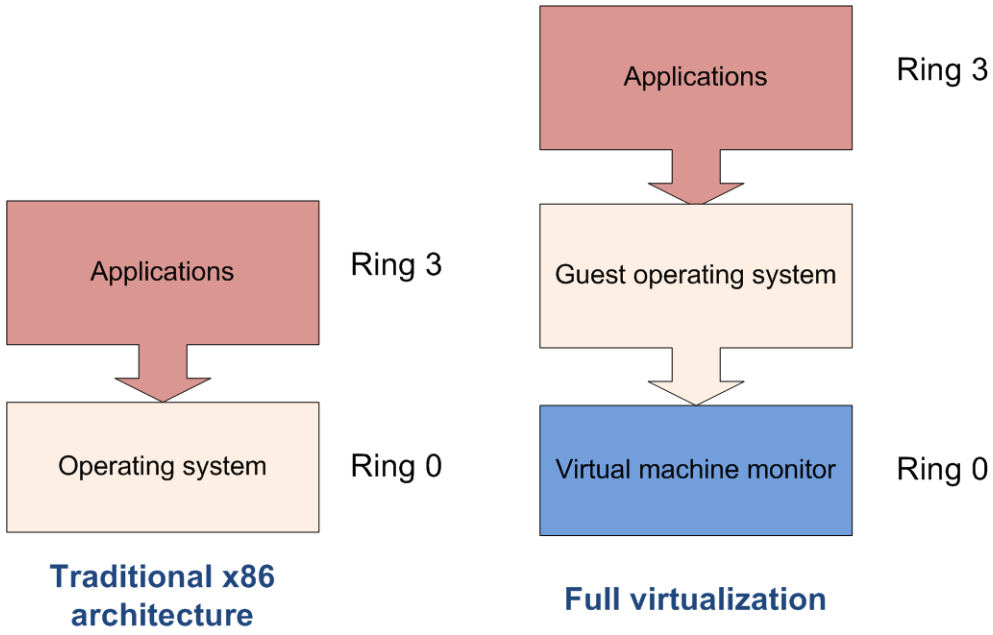
- Everything is virtualized
- Full hardware emulation
- Emulation = latency





Privileged Instructions

- Privileged instructions: OS kernel and device driver access to system hardware
- Trapped and emulated by VMM



Pros and Cons – Full Virtualization

- **Pros**
  - Disaster recovery, failover
  - Virtual appliance deployment
  - Legacy code on non-legacy hardware
- **Cons** – LATENCY of core four resources
  - RAM performance reduced 25% to 75%
  - Disk I/O degraded from 5% to 20%
  - Network performance decreased up to 10%
  - CPU privileged instruction dings nearing 1% to 7%

Paravirtualization

–OS or system devices are virtualization aware

Requirements:

–OS level – recompiled kernel

–Device level – paravirtualized or “enlightened” device drivers

Paravirtualization

- Pro: fast

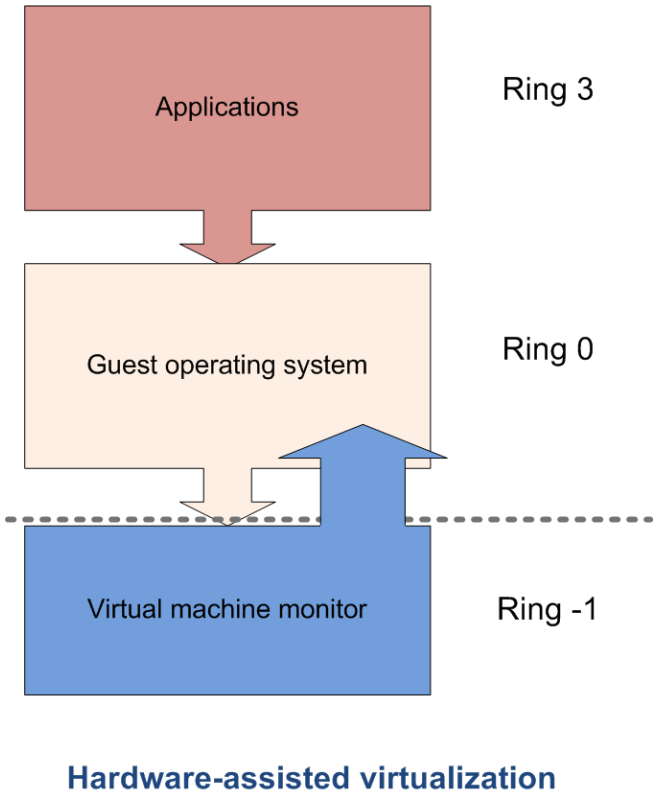
- Con: requires a specially modified guest OS, thus precludes the ability to run off-the-shelf and legacy OS in paravirtual environments

Hardware-assisted Virtualization

- Server hardware is virtualization aware
- Hypervisor and VMM load at privilege Ring -1 (firmware)
- Removes CPU emulation bottleneck
- Memory virtualization coming in quad core AMD and Intel CPUs

Actual CPUs implement virtualization enhancements

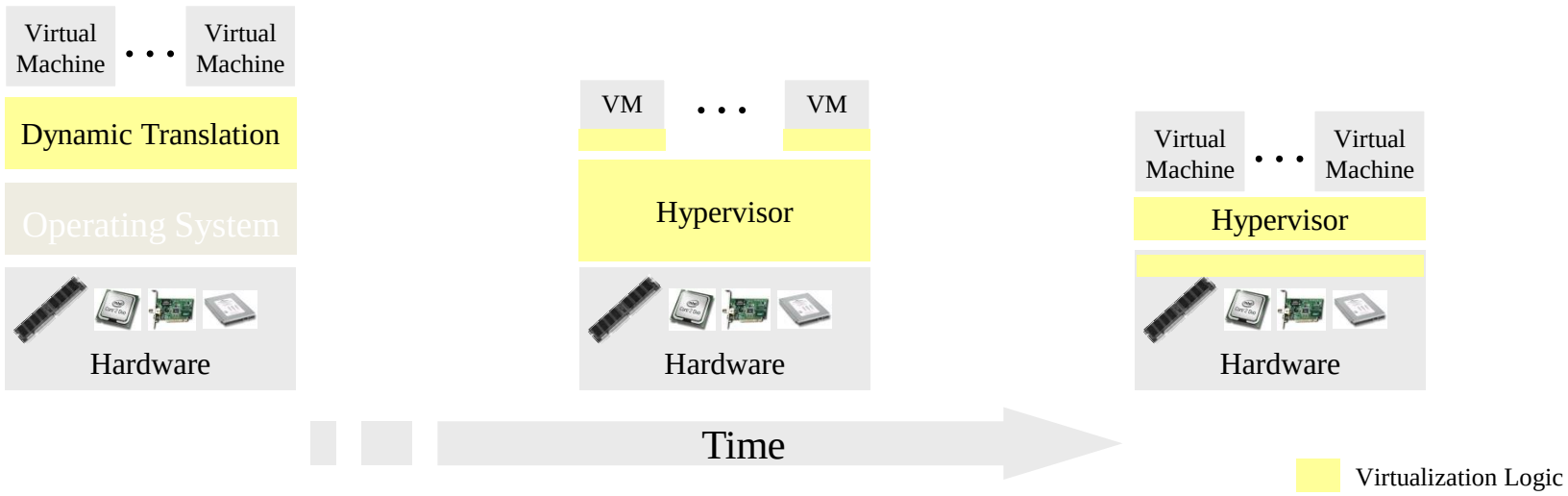
- Intel: Intel Virtualization Technology (Intel-VT)
- AMD: AMD Pacifica or AMD-V



Hardware-assisted virtualization

Evolution of Software solutions*

- 1st Generation: Full virtualization (Binary rewriting)
  - Software Based
  - VMware and Microsoft
- 2nd Generation: Paravirtualization
  - Cooperative virtualization
  - Modified guest
  - VMware, Xen
- 3rd Generation: Silicon-based (Hardware-assisted) virtualization
  - Unmodified guest
  - VMware and Xen on virtualization-aware hardware platforms



*This slide is from Intel® Corporation

Who is what?

- Full virtualization (aka emulation)
  - Bochs and QEMU
- Paravirtualization
  - Xen, VMware
- Binary Translation
  - VMware, VirtualPC, VirtualBox, QEMU
- Hardware Virtualization
  - Xen, VMware, VirtualPC, VirtualBox, KVM, …

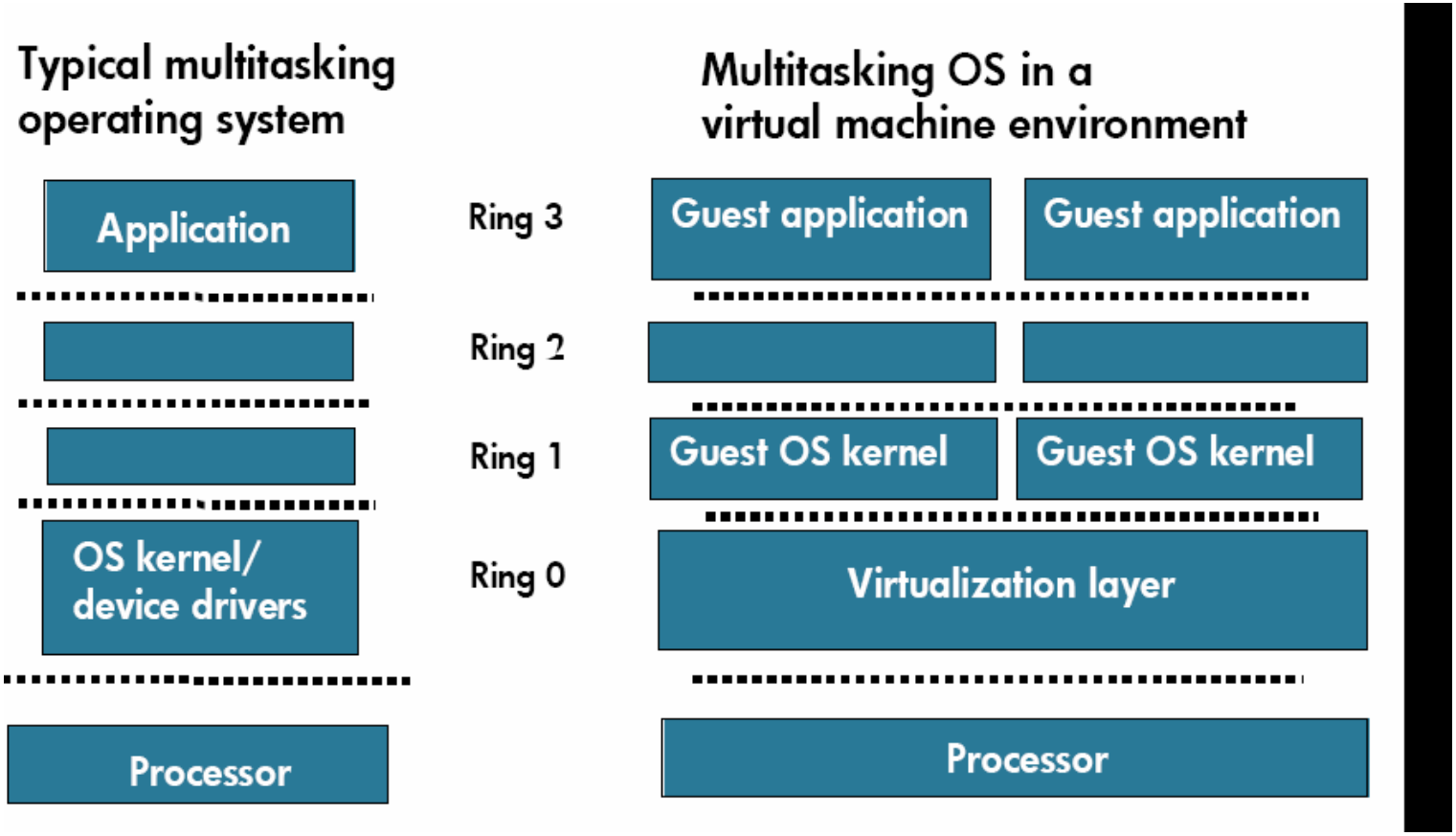
Software Virtualization Challenges

- CPUID instruction
- Ring Aliasing
- Ring Compression
- Memory addressing
- Non-faulting guest access to privileged state
- 17 instructions don't meet Popek and Goldberg criteria [Lawton and Robin] (citation)

CPUID instruction

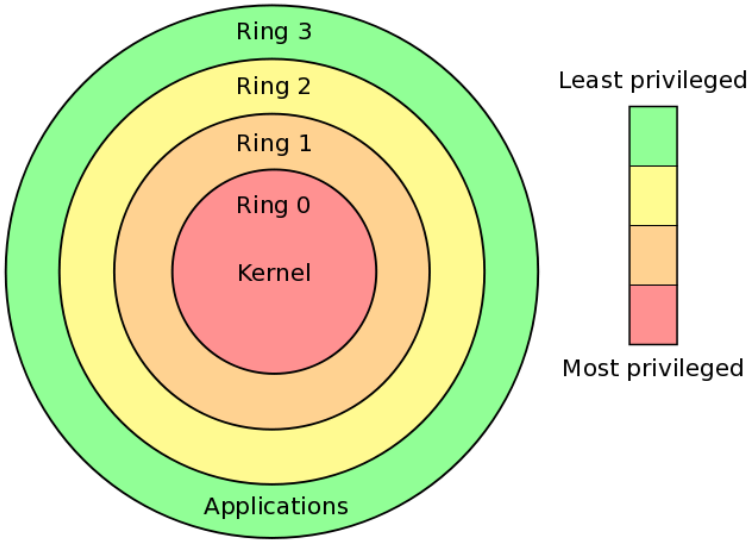
- Returns processor identification and feature information
- Thought: When employing virtualization, are there certain undesirable features not to be exposed to the guest?
  - Some of these features could make the guest believe it can do things it can't
  - Might want to mask off some features from guest (virtualization)

Privileges


Typical multitasking operating system
Application
OS kernel/ device drivers
Processor
Ring 3
Ring 2
Ring 1
Ring 0
Multitasking OS in a virtual machine environment
Guest application
Guest application
Guest OS kernel
Guest OS kernel
Virtualization layer
Processor

Ring Aliasing

- Ring 0 is most privileged
- OS kernels assume to be running at ring 0
  - Our guest VM is no different
- VMM and guest cannot share ring 0
  - If guest isn't in ring 0, could use PUSH CS to figure that out (CPL is in the last two bits of CS register)



See Figure 5-3. Protection Rings for Intel's version

Ring Compression

- IA32 supplies two isolation mechanisms, Segmentation and Paging
- Segmentation isn't available in 64-bit
- So paging is only choice for isolating a guest
- But paging doesn't distinguish between rings 0 – 2
  - See Section 5.11.2 "If the processor is currently operating at a CPL of 0, 1, or 2, it is in supervisor mode; if it is operating at a CPL of 3, it is in user mode."
- And our host kernel is in ring 0 and guest software is in ring 3. No more rings = we're compressed.
  - Therefore, our guest cannot be isolated from user-space applications and cannot be reasonably protected from each other.

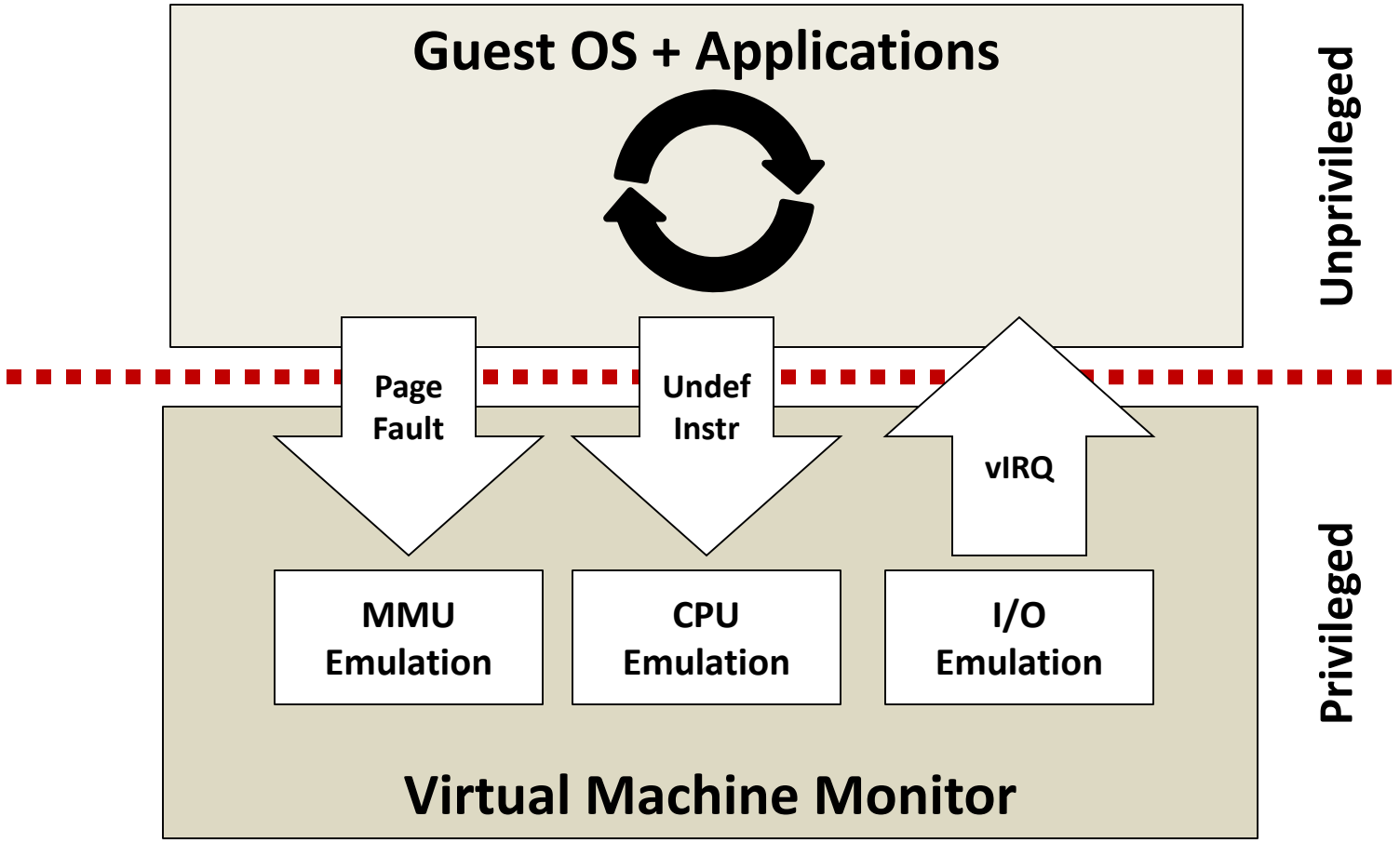
Software based techniques

1. Binary translation
   - Emulation of one instruction set by another for same CPU.
   - When source and target instruction set are the same, it's called instruction set simulation
   - can be done "just in time" (JIT)
     - can do some caching to be more efficient (i.e., hot spot detection)
2. Para-virtualization
   - modification of guest kernel to support being virtualized
   - Can be pretty efficient

Processor Virtualization

- Trap and Emulate
- Binary Translation

Trap and Emulate



“Strictly Virtualizable”

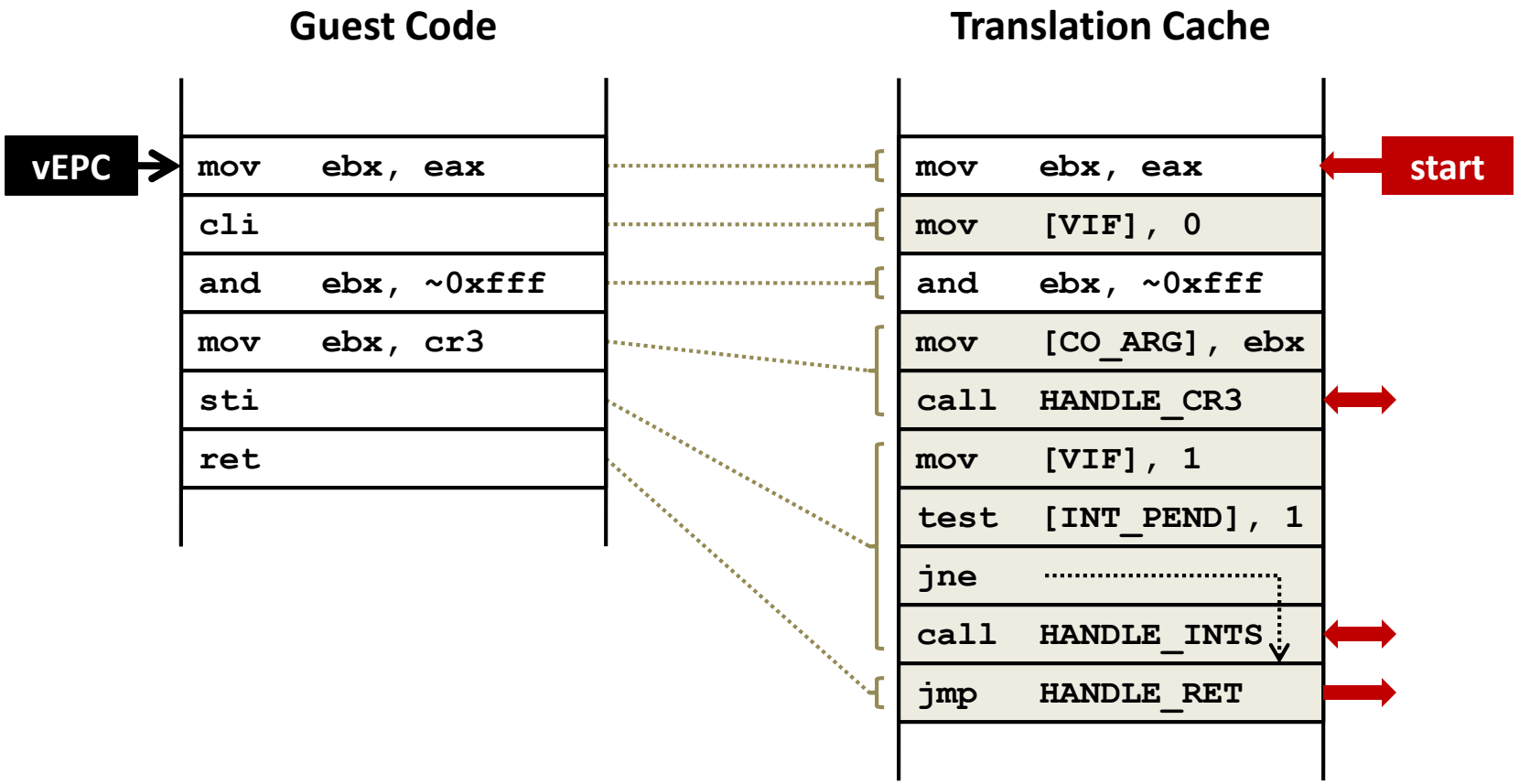
A processor or mode of a processor is *strictly virtualizable* if, when executed in a lesser privileged mode:

- all instructions that access privileged state trap
- all instructions either trap or execute identically

Issues with Trap and Emulate

- Not all architectures support it
- Trap costs may be high
- VMM consumes a privilege level
  - Need to virtualize the protection levels

Binary Translation



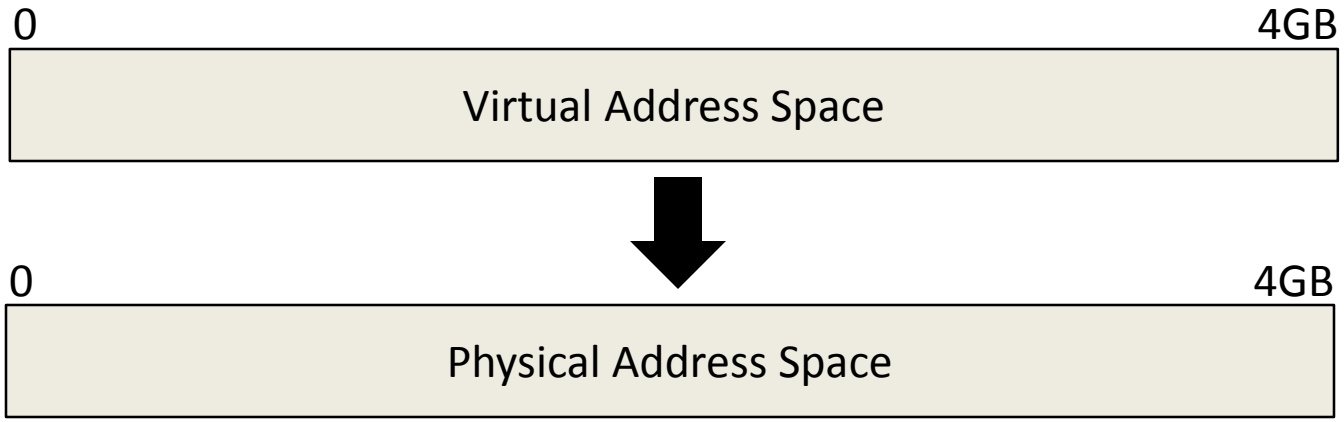
Issues with Binary Translation

- Translation cache management
- PC synchronization on interrupts
- Self-modifying code
  - Notified on writes to translated guest code
- Protecting VMM from guest

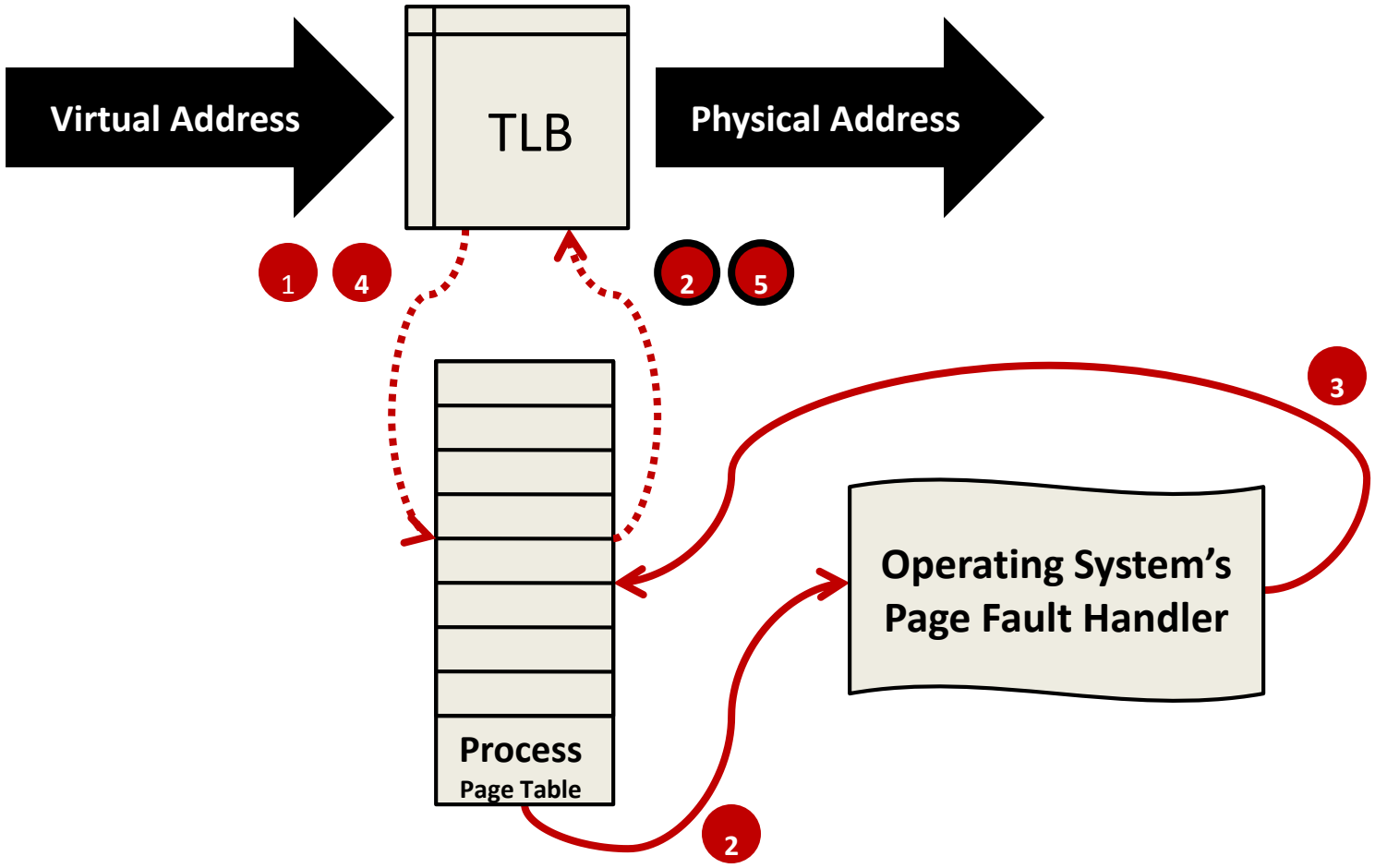
Memory Virtualization

- Shadow Page Tables
- Nested Page Tables

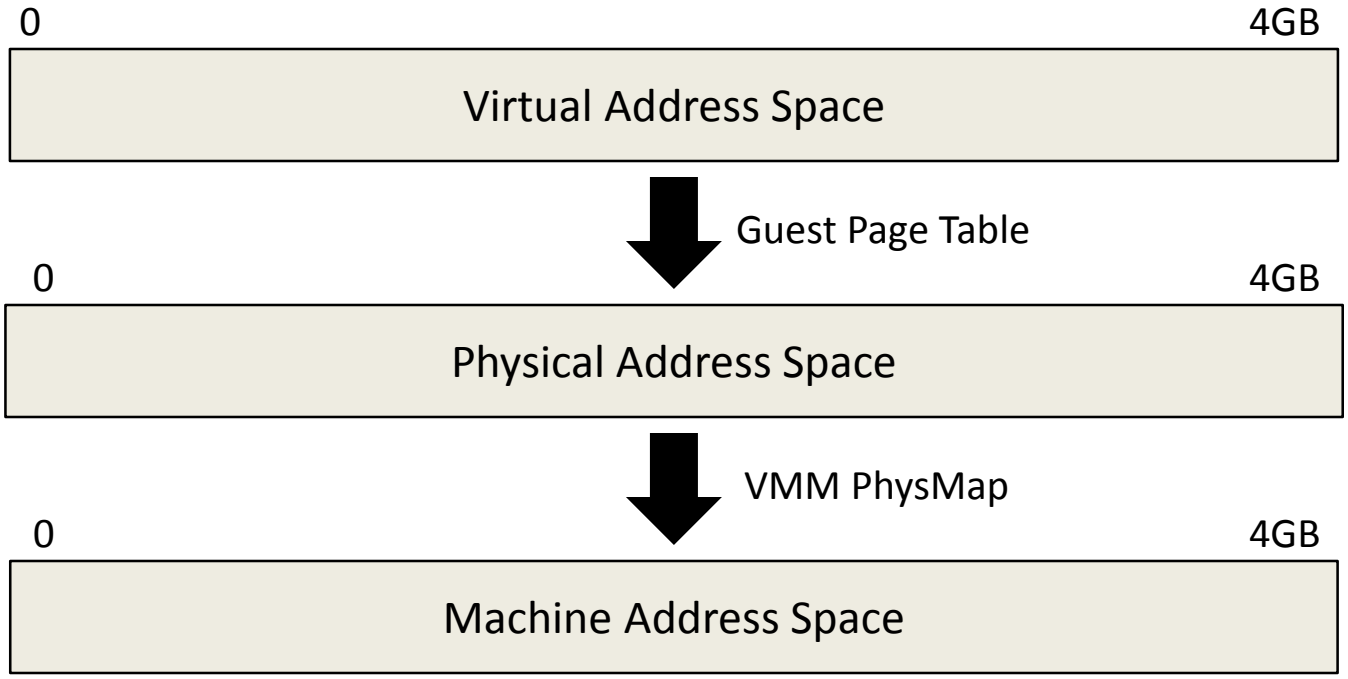
Traditional Address Spaces



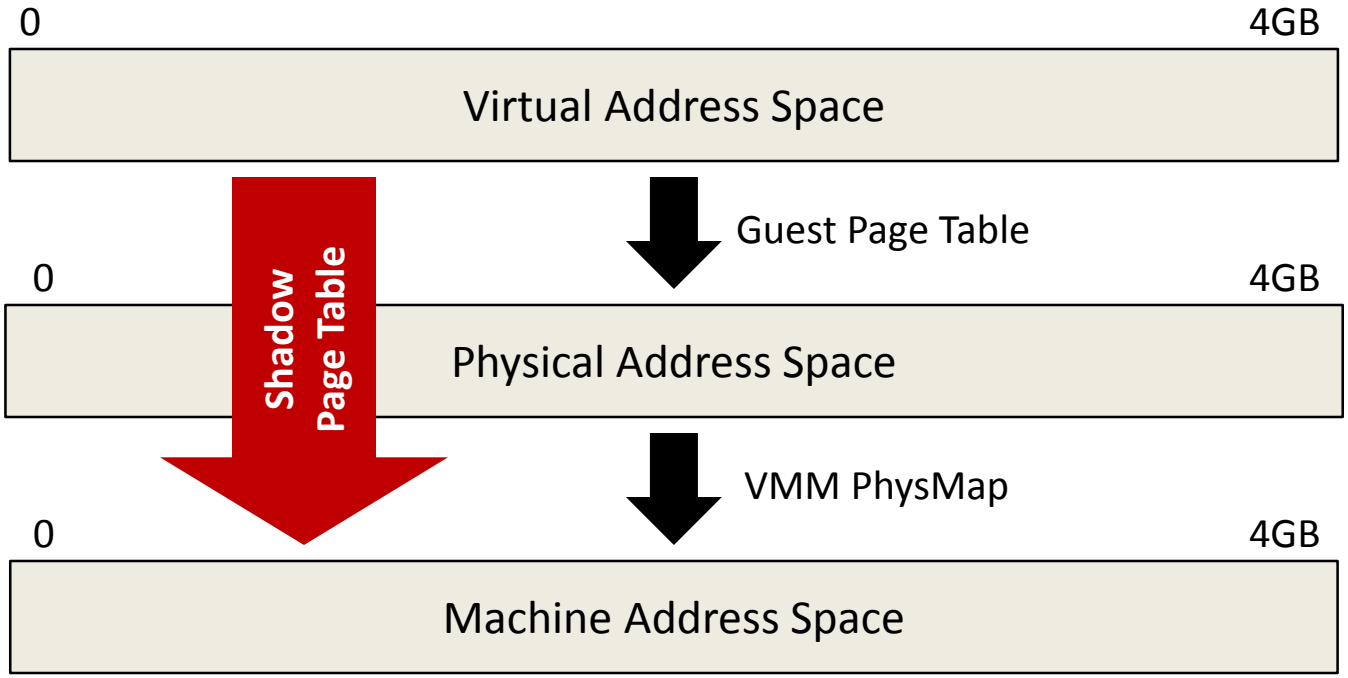
Traditional Address Translation



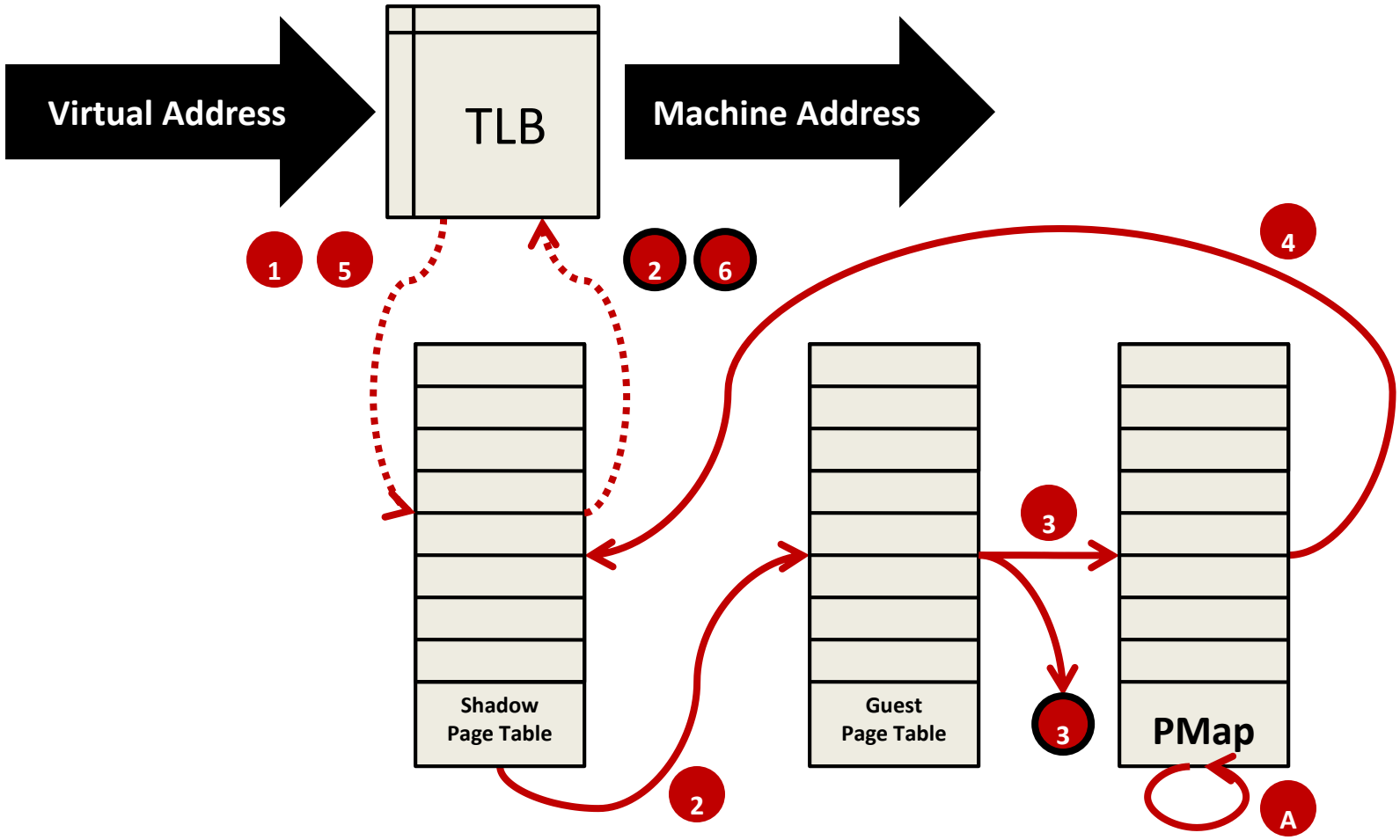
Virtualized Address Spaces



Virtualized Address Spaces w/ Shadow Page Tables



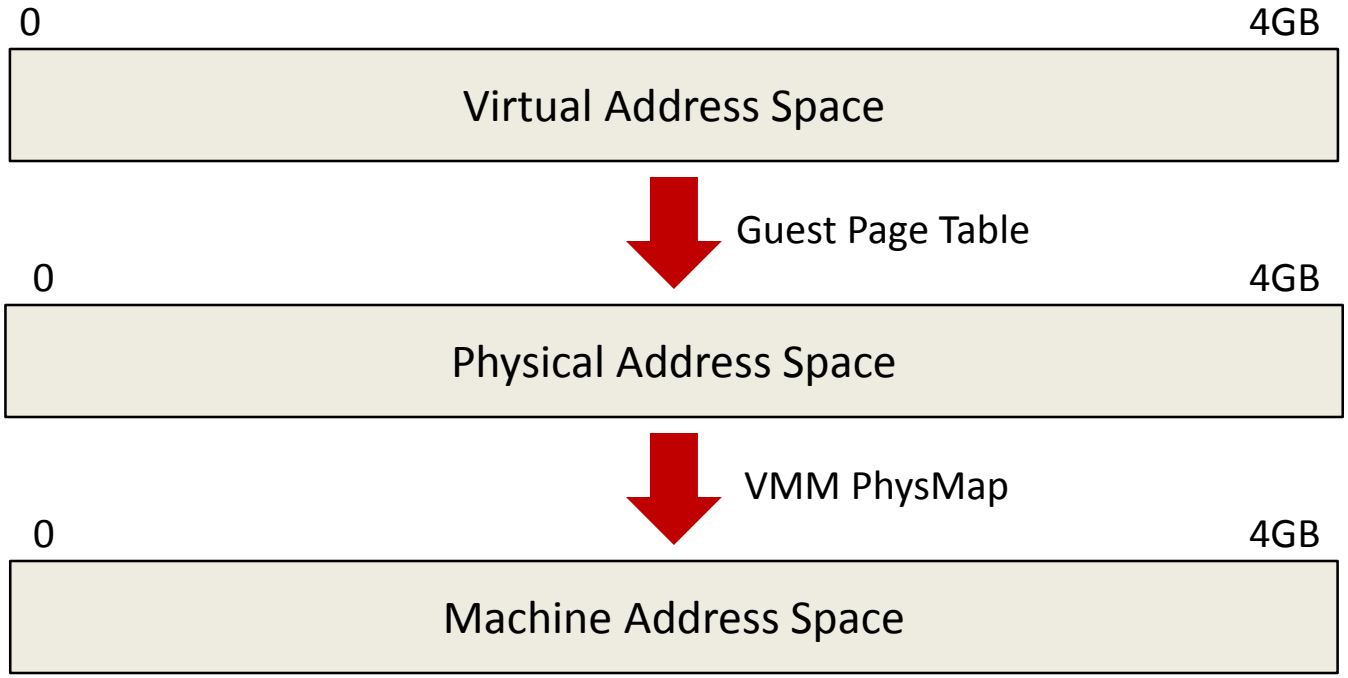
Virtualized Address Translation w/ Shadow Page Tables



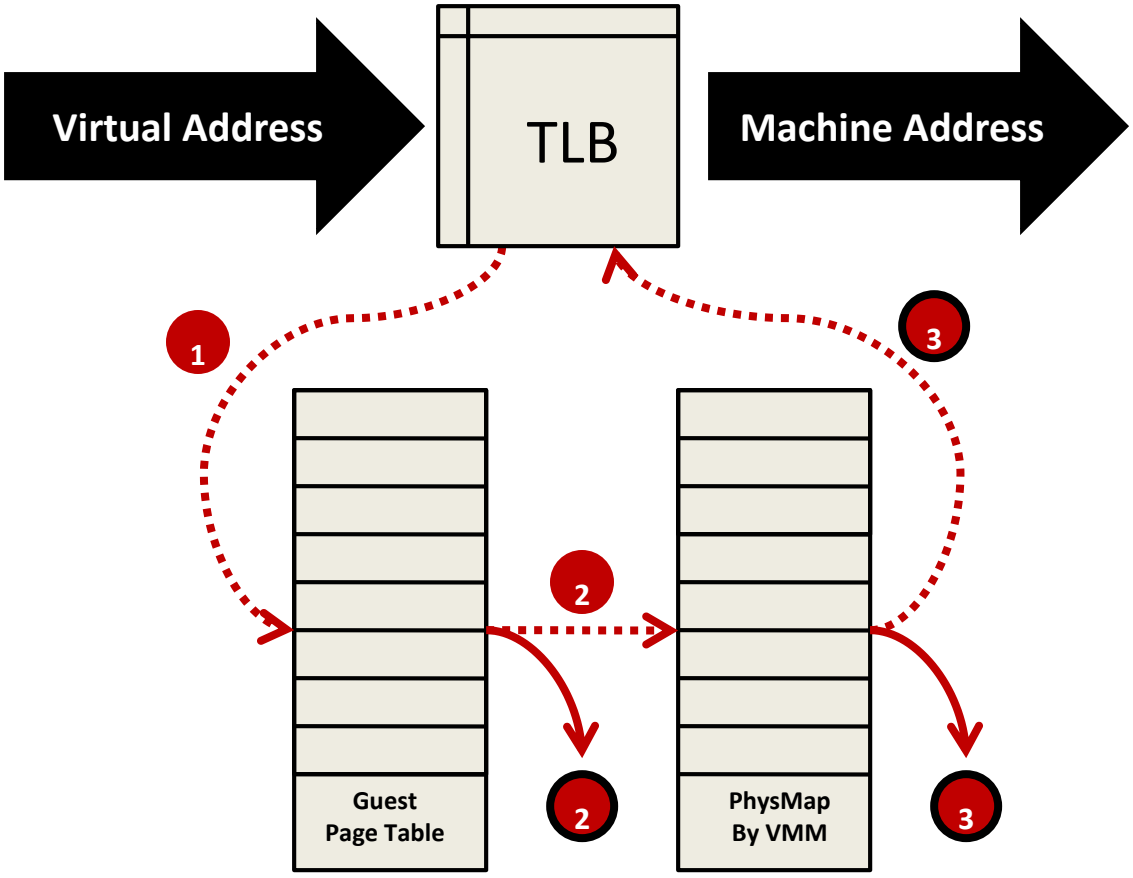
Issues with Shadow Page Tables

- Guest page table consistency
  - Rely on guest's need to invalidate TLB
- Performance considerations
  - Aggressive shadow page table caching necessary
  - Need to trace writes to cached page tables

Virtualized Address Spaces w/ Nested Page Tables



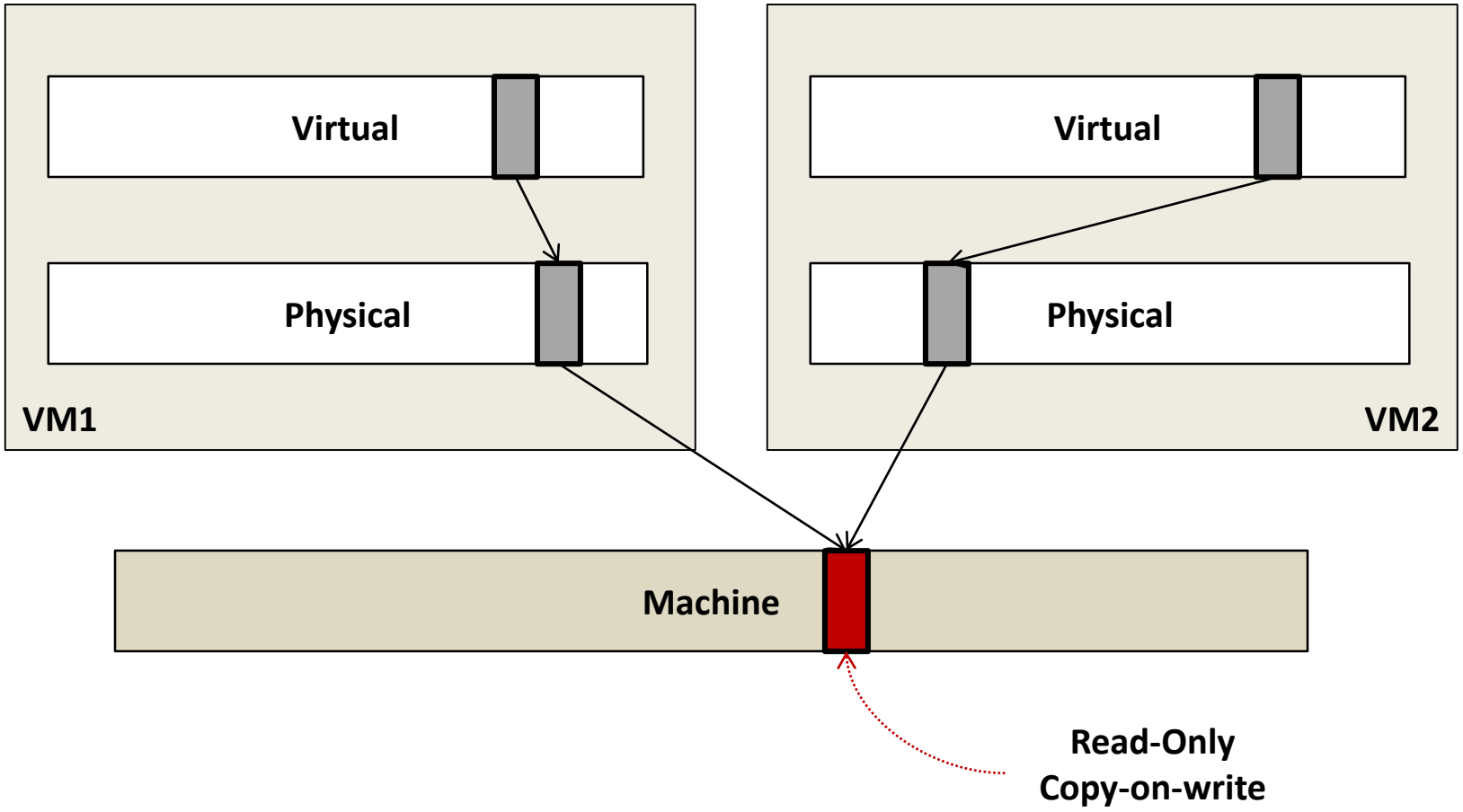
Virtualized Address Translation w/ Nested Page Tables



Issues with Nested Page Tables

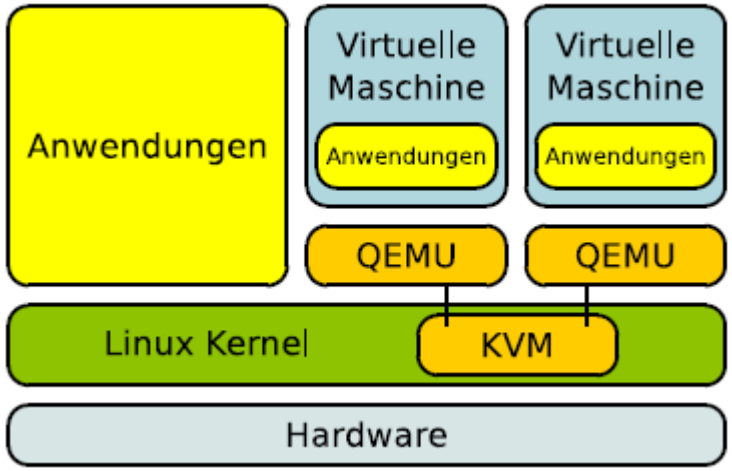
- Positives
  - Simplifies monitor design
  - No need for page protection calculus
- Negatives
  - Guest page table is in physical address space
  - Need to walk PhysMap multiple times
    - Need physical-to-machine mapping to walk guest page table
    - Need physical-to-machine mapping for original virtual address
- Other Memory Virtualization Hardware Assists
  - Monitor Mode has its own address space
    - No need to hide the VMM

Interposition with Memory Virtualization Page Sharing



Beispiel KVM

- KVM ist als Modul direkt im Linux-Kernel integriert
  - KVM-Basismodul: kvm.ko
  - Hardwarespezische Module: kvm-intel.ko und kvm-amd.ko



- Nach dem Laden der Module arbeitet der Kernel selbst als Hypervisor
  - KVM kann nur mit CPUs mit Hardwarevirtualisierung arbeiten
  - Dadurch braucht KVM weniger Quellcode als z.B. Xen
- Neben den Kernelmodulen enthält KVM den Emulator QEMU
  - KVM stellt keine virtuelle Hardware zur Verfügung. Das macht QEMU
  - CPU-Virtualisierung stellt der Prozessor bereit (Intel VT oder AMD-V)
  - Der Speicher wird durch KVM virtualisiert
  - E/A wird durch einen QEMU-Prozess pro Gastsystem virtualisiert

QEMU

- **QEMU is a fast processor emulator**
  - Uses a portable dynamic translator
  - Can emulate devices needed to run an OS
- **Two operating modes:**
- **User space only**
  - Launch Linux processes compiled for one CPU on another CPU
  - Cross-compilation
  - Cross-debugging
- **Full system Virtualization**
  - Emulate a full system that includes
    - A processor
    - Several peripheral devices

From Kernel to Userspace (QEMU)

- **Kernel provides /dev/kvm device**
- **through this device, the following functions are done:**
  - own address-space for virtual machines
  - simulated I/O (can be mapped to host but doesn't have to)
  - video is being forwarded to the host
- **QEMU does all of his work via this device.**
- **limited paravirtualization is available (using the VirtIO framework)**
  - paravirtual Ethernet card(s)
  - paravirtual Disk controller(s)
  - baloon device (for adjusting memory usage on-the-y)
  - VGA graphics interface (SPICE or VMWare drivers)

Libvirt: The virtualization API

- **Libvirt is:**
  - A toolkit to interact with the virtualization capabilities of recent versions of Linux (and other OSes)
  - Free software available under the GNU LGPL
  - A long term stable C API
  - A set of bindings for common languages
- **Libvirt provides:**
  - Remote management using TLS encryption and x509 certificates
  - Remote management authenticating with Kerberos and SASL
  - Local access control using PolicyKit
  - Zero-conf discovery using Avahi multicast-DNS
  - Management of virtual machines, virtual networks and storage
  - Portable client API for Linux, Solaris and Windows

Libvirt

- **Libvirt supports:**
  - KVM/QEMU Linux hypervisor
  - Xen hypervisor on Linux and Solaris hosts
  - LXC and OpenVZ Linux container systems
  - VirtualBox hypervisor
  - VMware ESX and GSX hypervisors
  - VMware Workstation and Player hypervisors
  - Microsoft Hyper-V hypervisor
  - Virtual networks using bridging, NAT, VEPA and VN-LINK
  - Storage on IDE/SCSI/USB disks, FibreChannel,
  - LVM, iSCSI, NFS and filesystems

Linux Virtualization Tools

- **The following tools utilize libvirt**
  - **virsh** Interactive shell, and batch scriptable tool for performing management tasks
  - **virt-install** Provision new virtual machines from a OS distribution install tree. (ISO, NFS, HTTP and FTP)
  - **virt-manager** GUI application
  - **virt-df** Peeks into the guest disks and determines how much space is used. (Linux FS, LVM)
  - **virt-top** Watch CPU, memory, network and disk utilization
  - **virt-p2v, virt-viewer, virt-image, virt-clone, ...**

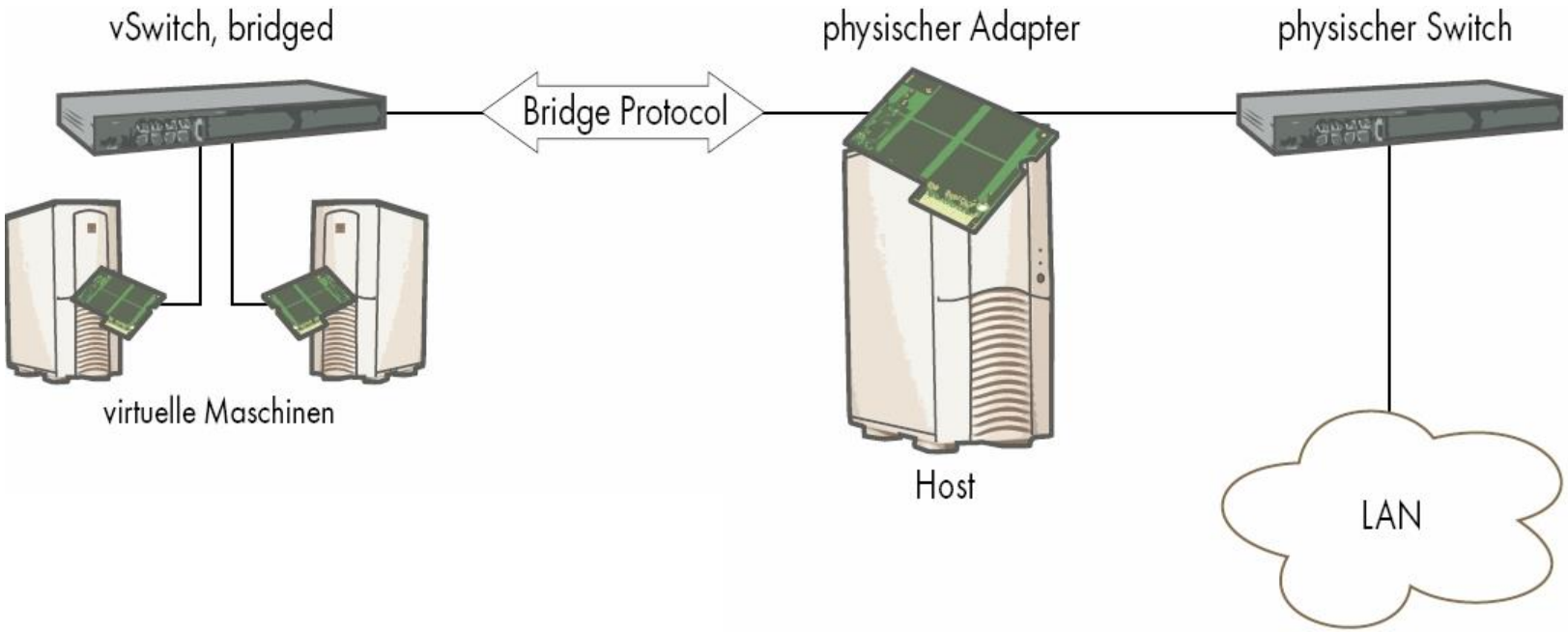
Cloud Infrastructure

- Netzwerkvirtualisierung

- Motivation
  - Netzwerktopologien ohne Verkabelung aufbauen
  - Sicherheit durch logische Trennung
- Technologien
  - Virtuelle Netzwerke
  - VLANs
- Trend
  - SDN und NFV

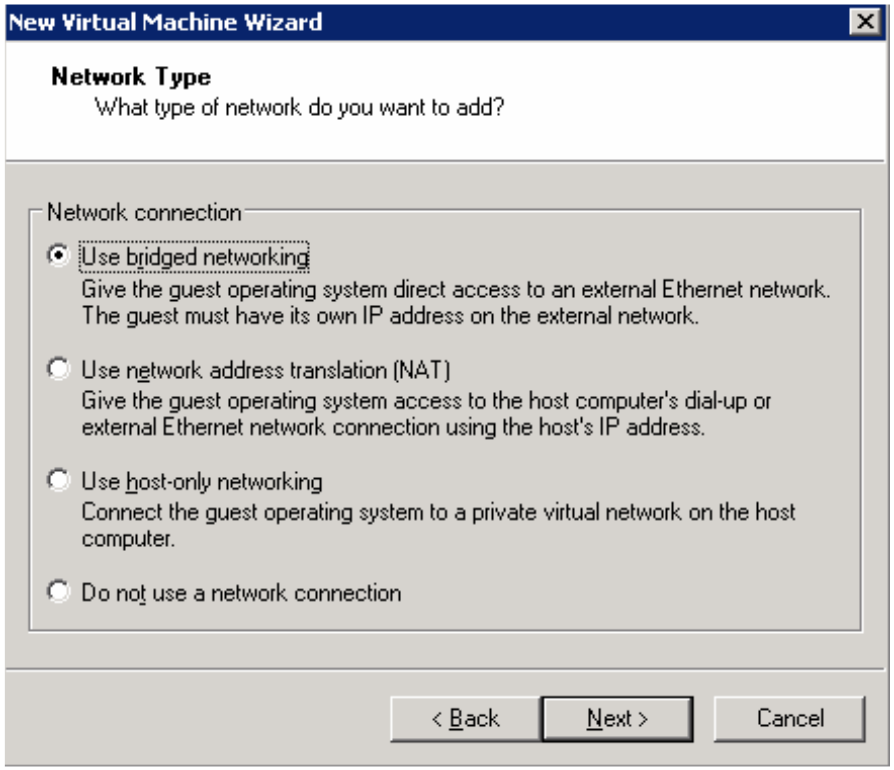
Virtuelle Netzwerke

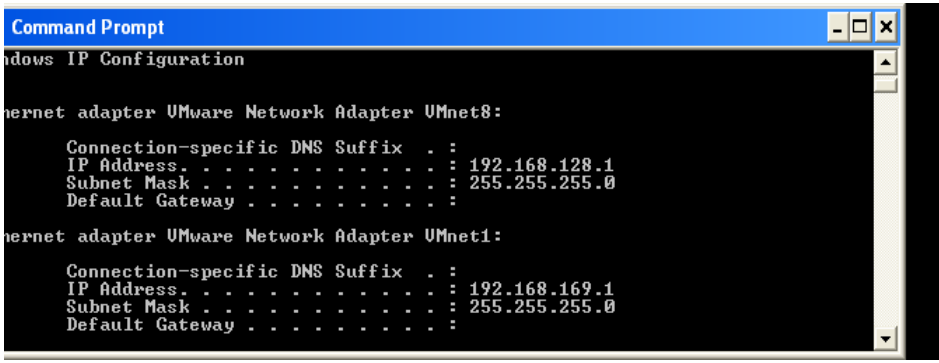
- Der Hypervisor stellt eine virtuelle Netzwerkumgebung für die Gäste bereit
  - Verschiedene Optionen konfigurierbar
  - z.B. NAT, DHCP, etc.



Virtuelle Netzwerke

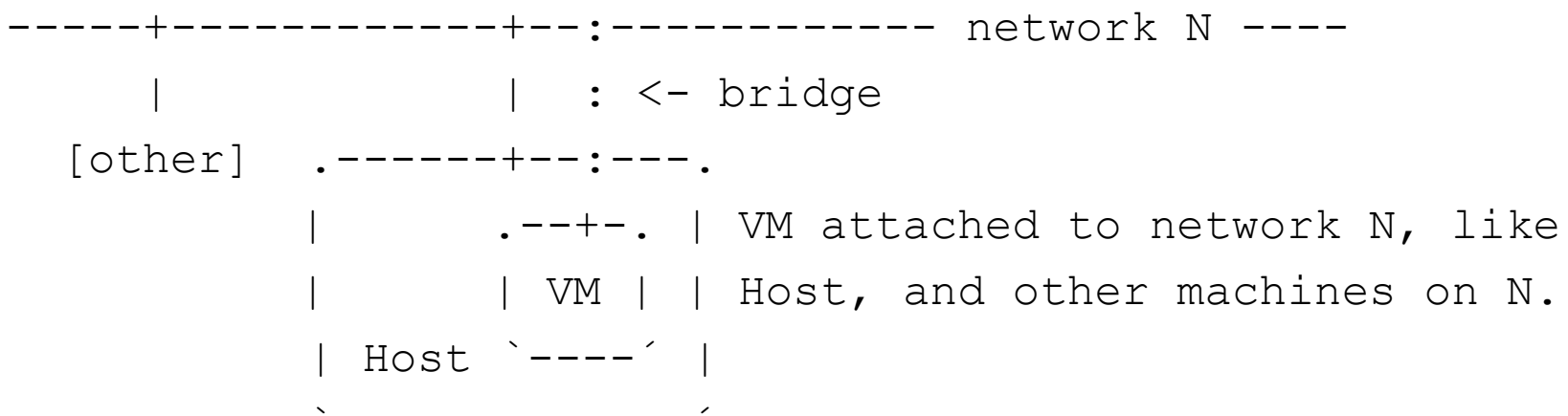
- Optionen für das virtuelle Netzwerk





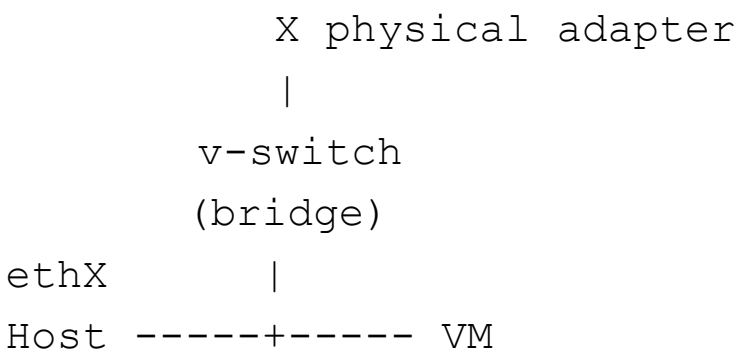
Bridged Mode

- This is the most straightforward way to attach a Virtual Machine to the network.
- The VM is bridged, or attached, to the same network that the host OS is physically connected to, using the Host's interface
- If DHCP is present on this network, then the VM will receive an IP from it.
- To other hosts on the network, the VM will look like just any other machine, with its own MAC address and IP address.



Bridged Mode

- Inside the Host OS, a virtual switch is used to achieve this.



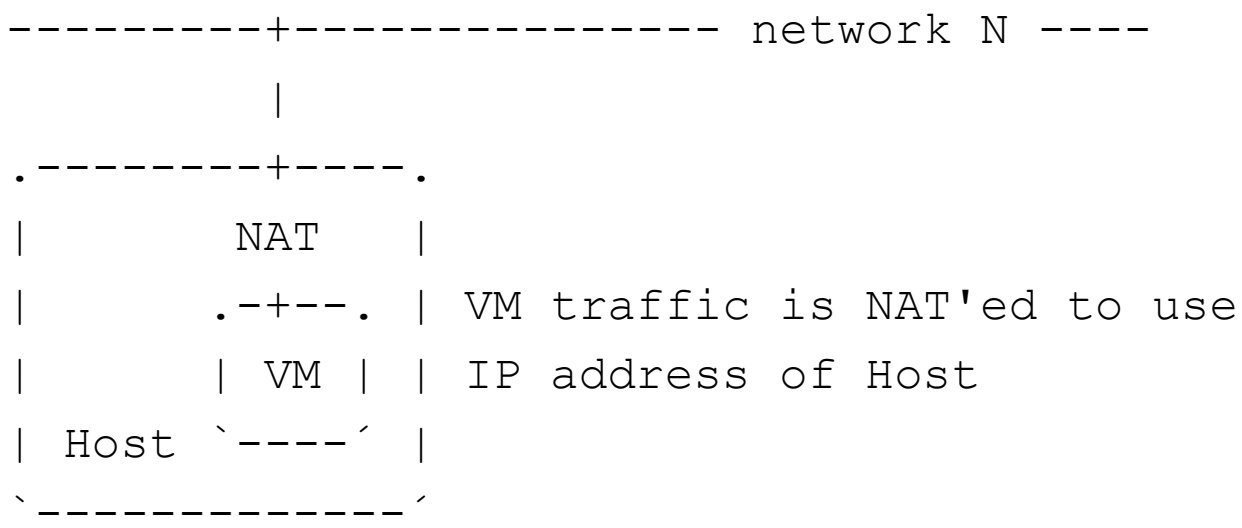
- Multiple VMs can be attached to the network in this fashion.
  - Pros: VM participates in the network like any other machine
  - Cons: VMs are visible to other machines (whether one wants to or not, and there has to be enough IPs available on the network

Host only / internal

- These modes do not place the VM directly on the network where the Host is connected to the outside world.
- Instead, a private, internal network is created, and the VM is placed on it. Each mode works differently:
  - host-only: the Host and the Guest VM can communicate together, but the VM cannot talk to the outside world – only with the Host For example, it's possible to SSH directly to the Guest from the Host (and vice-versa)
  - internal: the Guest VM is isolated and cannot talk to the Host or the outside world, but can talk to other VMs, if they are connected to the same internal network

NAT

- This is the default way for VirtualBox and VMware Desktop editions to connect the VM. NAT (Network Address Translation) is used to allow the Guest VM to access the outside networks.
- NAT uses the IP address configured on the Host's physical interface, but the Host and Guest still cannot talk together.



NAT Service (VirtualBox)

- The Network Address Translation (NAT) service works in a similar way to a home router, grouping the systems using it into a network and preventing systems outside of this network from directly accessing systems inside it, but letting systems inside communicate with each other and with systems outside using TCP and UDP over IPv4 and IPv6.

- A NAT service is attached to an internal network. An example command to create a NAT network is:
  - `VBoxManage natnetwork add --netname natnet1 --network "192.168.15.0/24" --enable`
  - Here, "natnet1" is the name of the internal network to be used. To attach a DHCP server to the internal network, we modify the example as follows:
  - `VBoxManage natnetwork add --netname natnet1 --network "192.168.15.0/24" --enable --dhcp on`

Übersichtstabelle Zugriffsmöglichkeiten

Netzwerktyp	Zugriff Gast -> andere Gäste	Zugriff Host -> Gast	Zugriff Gast -> externes Netzwerk
Not attached	-	-	-
Network Address Translation (NAT)	-	-	✓
Network Address Translation Service	✓	-	✓
Bridged networking	✓	✓	✓
Internal networking	✓	-	-
Host-only networking	✓	✓	-

NAT refresher

- Network Address Translation
  - RFC 1631

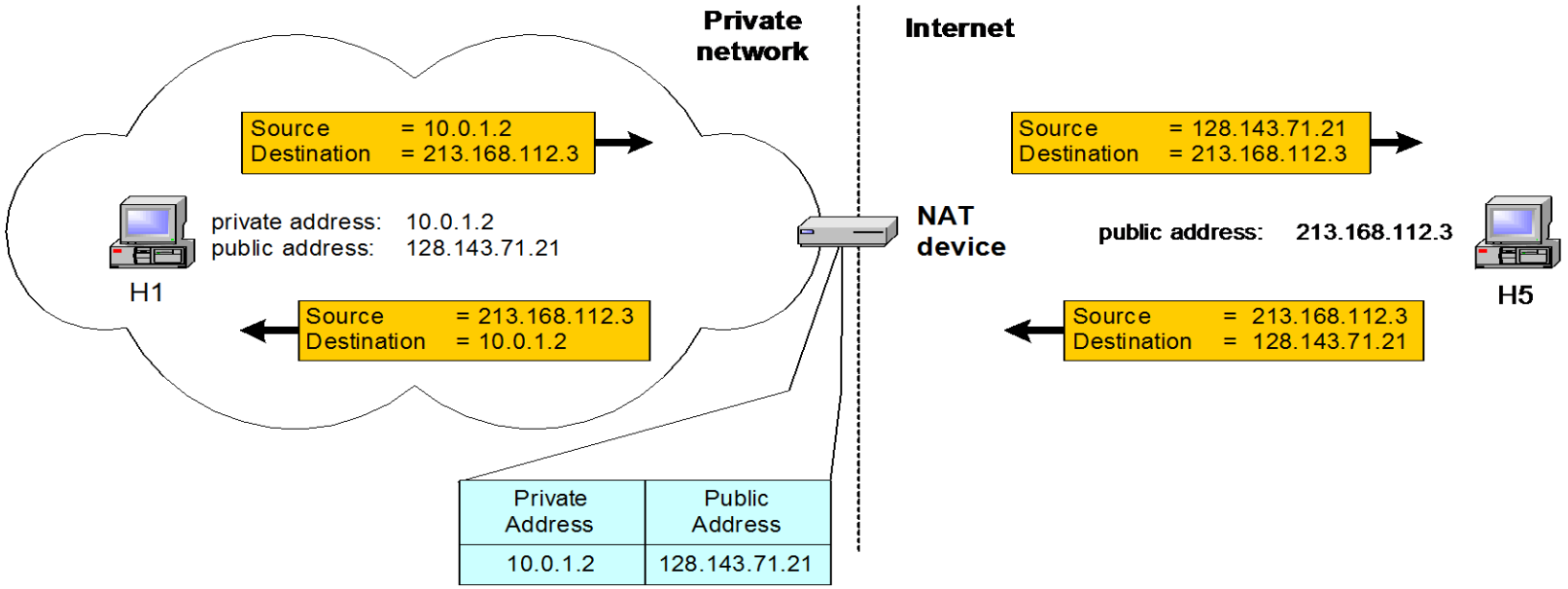
- A short term solution to the problem of the depletion of IP addresses
  - Long term solution is IP v6

- NAT is a way to conserve IP addresses
  - Can be used to hide a number of hosts behind a single IP address
  - Uses private addresses:
    - 10.0.0.0-10.255.255.255,
    - 172.16.0.0-172.32.255.255 or
    - 192.168.0.0-192.168.255.255

NAT refresher

- NAT is a router function where IP addresses (and possibly port numbers) of IP datagrams are replaced at the boundary of a private network
- NAT is a method that enables hosts on private networks to communicate with hosts on the Internet
- NAT is run on routers that connect private networks to the public Internet, to replace the IP address-port pair of an IP packet with another IP address-port pair.

NAT Overview

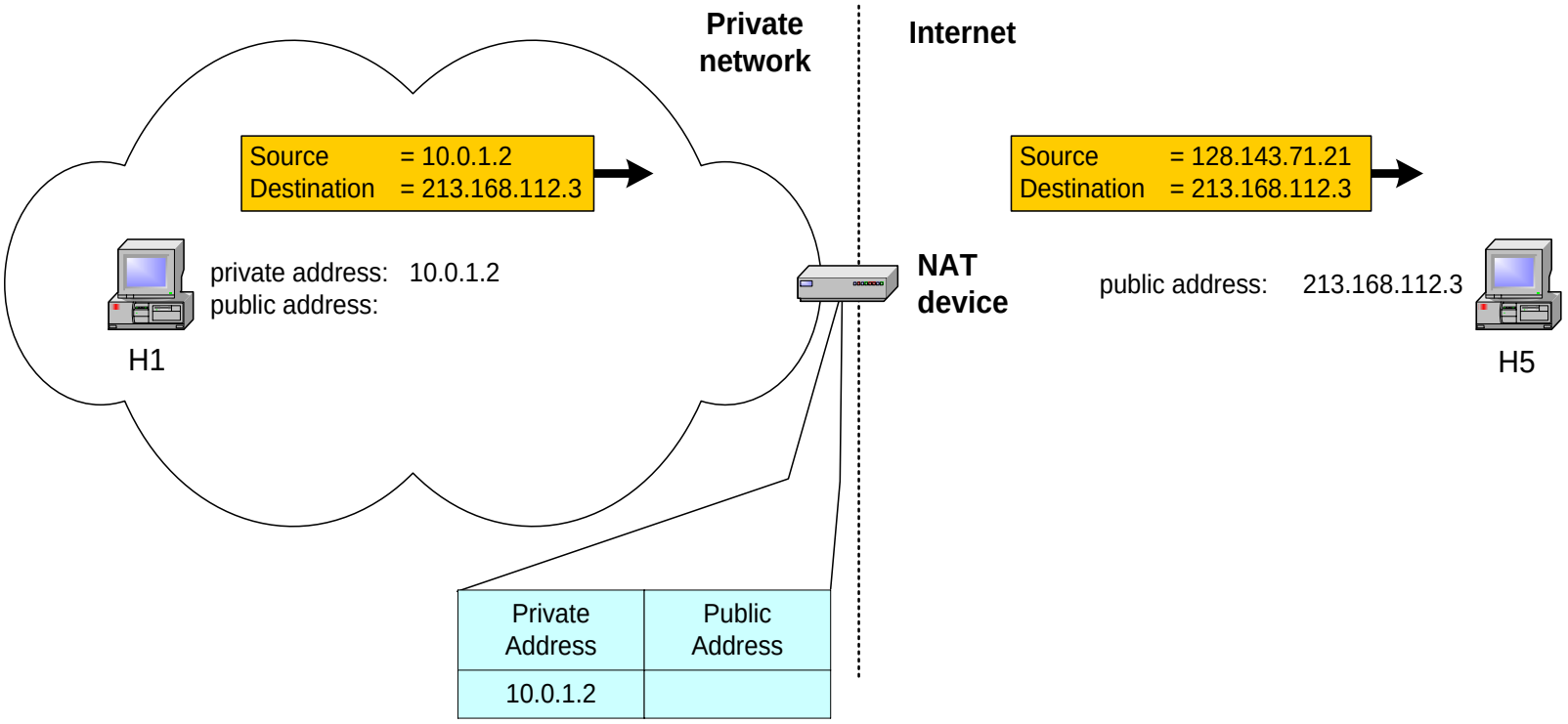


- NAT device has address translation table
- One to one address translation

NAT Pooling of IP Addresses

- **Scenario:** Corporate network has many hosts but only a small number of public IP addresses
- **NAT solution:**
  - Corporate network is managed with a private address space
  - NAT device, located at the boundary between the corporate network and the public Internet, manages a pool of public IP addresses
  - When a host from the corporate network sends an IP datagram to a host in the public Internet, the NAT device picks a public IP address from the address pool, and binds this address to the private address of the host

NAT Pooling of IP Addresses

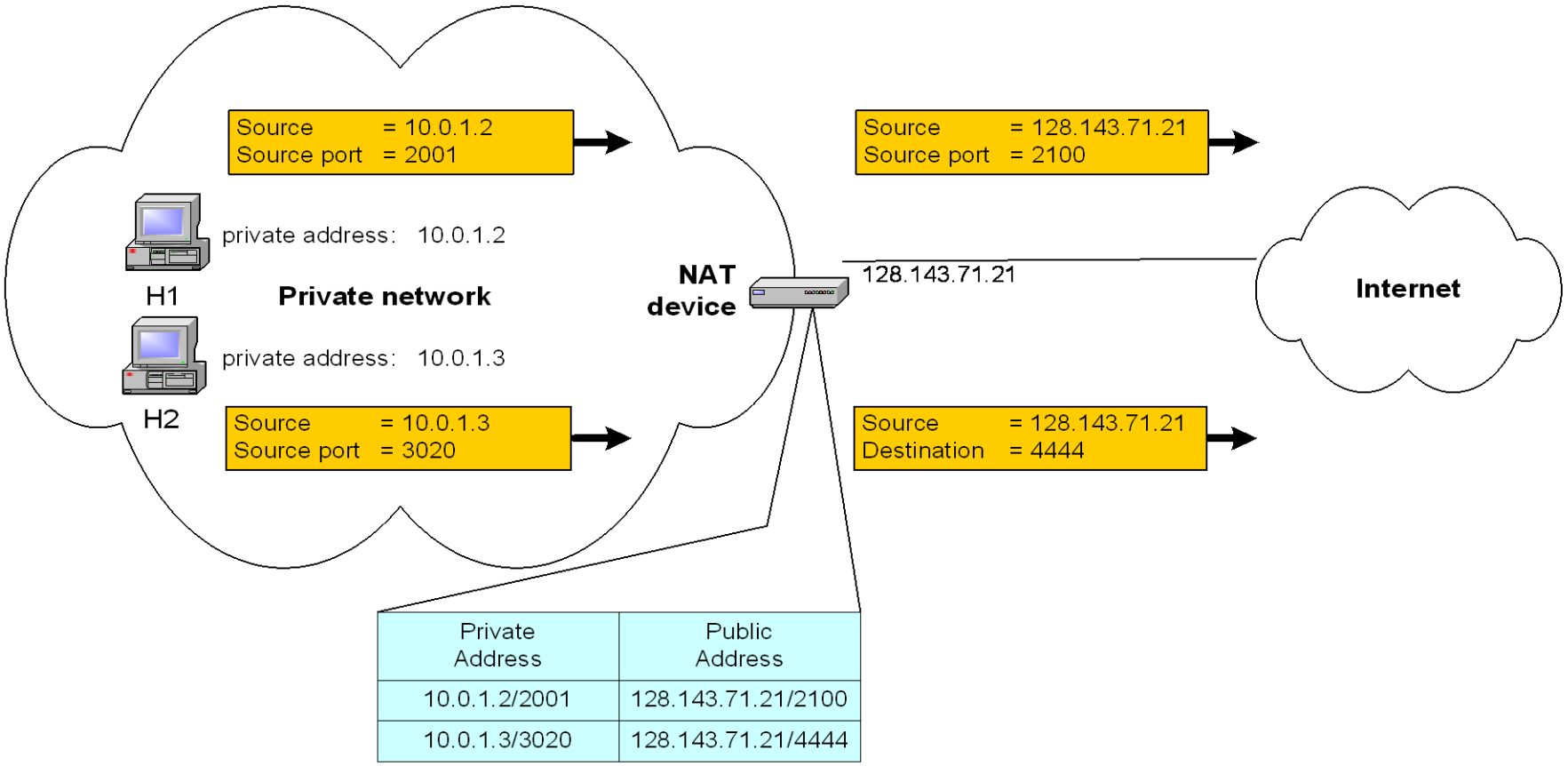


Pool of addresses: 128.143.71.0-128.143.71.30

IP Masquerading

- **Also called: Network address and port translation (NAPT), port address translation (PAT).**
- **Scenario:** Single public IP address is mapped to multiple hosts in a private network.
- **NAT solution:**
  - Assign private addresses to the hosts of the corporate network
  - NAT device modifies the port numbers for outgoing traffic

IP Masquerading



NAT issues

- **Performance:**
  - Modifying the IP header by changing the IP address requires that NAT boxes recalculate the IP header checksum
  - Modifying port number requires that NAT boxes recalculate TCP checksum
- **Fragmentation**
  - Care must be taken that a datagram that is fragmented before it reaches the NAT device, is not assigned a different IP address or different port numbers for each of the fragments.

NAT issues

- **End-to-end connectivity:**
  - NAT destroys universal end-to-end reachability of hosts on the Internet.
  - A host in the public Internet often cannot initiate communication to a host in a private network.
  - The problem is worse, when two hosts that are in a private network need to communicate with each other.

NAT issues

- **IP address in application data:**
  - Applications that carry IP addresses in the payload of the application data generally do not work across a private-public network boundary.
  - Some NAT devices inspect the payload of widely used application layer protocols and, if an IP address is detected in the application-layer header or the application payload, translate the address according to the address translation table.

Links and References Networking

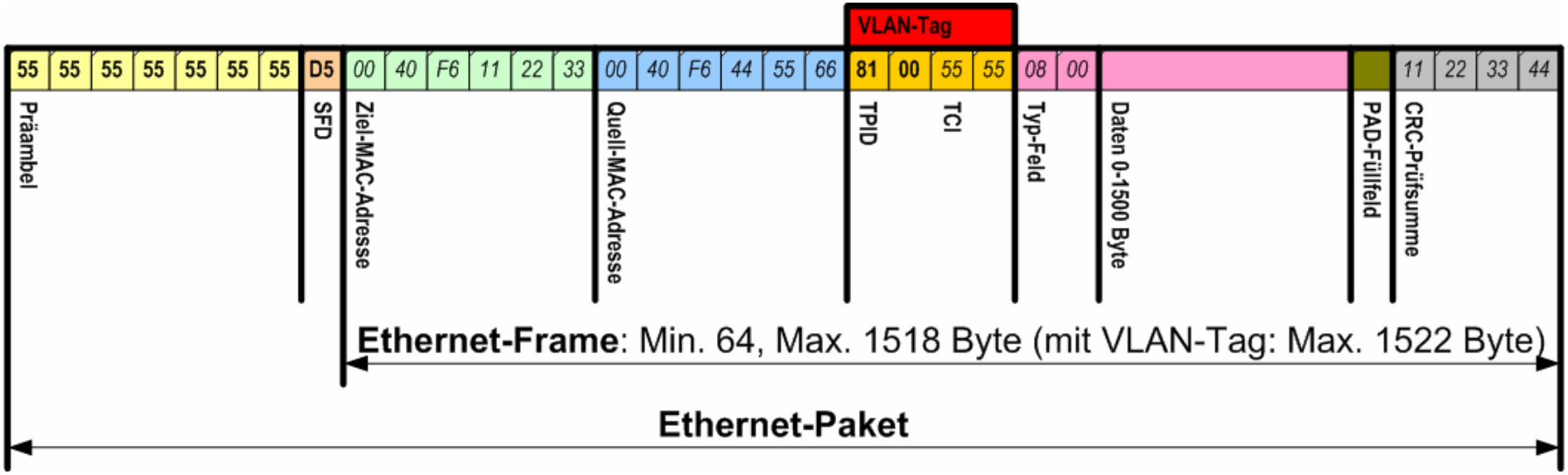
- http://unix.stackexchange.com/questions/118891/wireless-bridged-networking-in-kvm-why-is-it-so-complicated
- https://forum.proxmox.com/threads/proxmox-under-virtualbox-no-outbound-networking.20054/
- http://www.vmaschinen.de/cgi-bin/vmware.cgi?netzwerk
- Networking course
  - http://www.ics.uci.edu/~keldefra/teaching/winter2014/uci_netsys202_ics233/outlineG.htm

Virtual Local Area Network (VLAN)

- Trennung von physikalischer Netzwerktopologie
  - mit Hubs und Switches von der logischen VLAN Topologie
- Motivation
  - Reduktion der Broadcast Last
  - Sicherheit
  - Flexibilität: Virtuelle Netze können ohne Änderung der Verkabelung erstellt werden

VLAN Funktionsweise

- VLAN bekommt Nummer
  - Wird in Ethernet Paketen eingebaut
  - Unterstützung durch VLAN-taugliche Switches (entsprechend IEEE 802.1 Q)
  - VLAN Tags werden auf dem Weg von und zu Endgeräten entfernt / angebracht

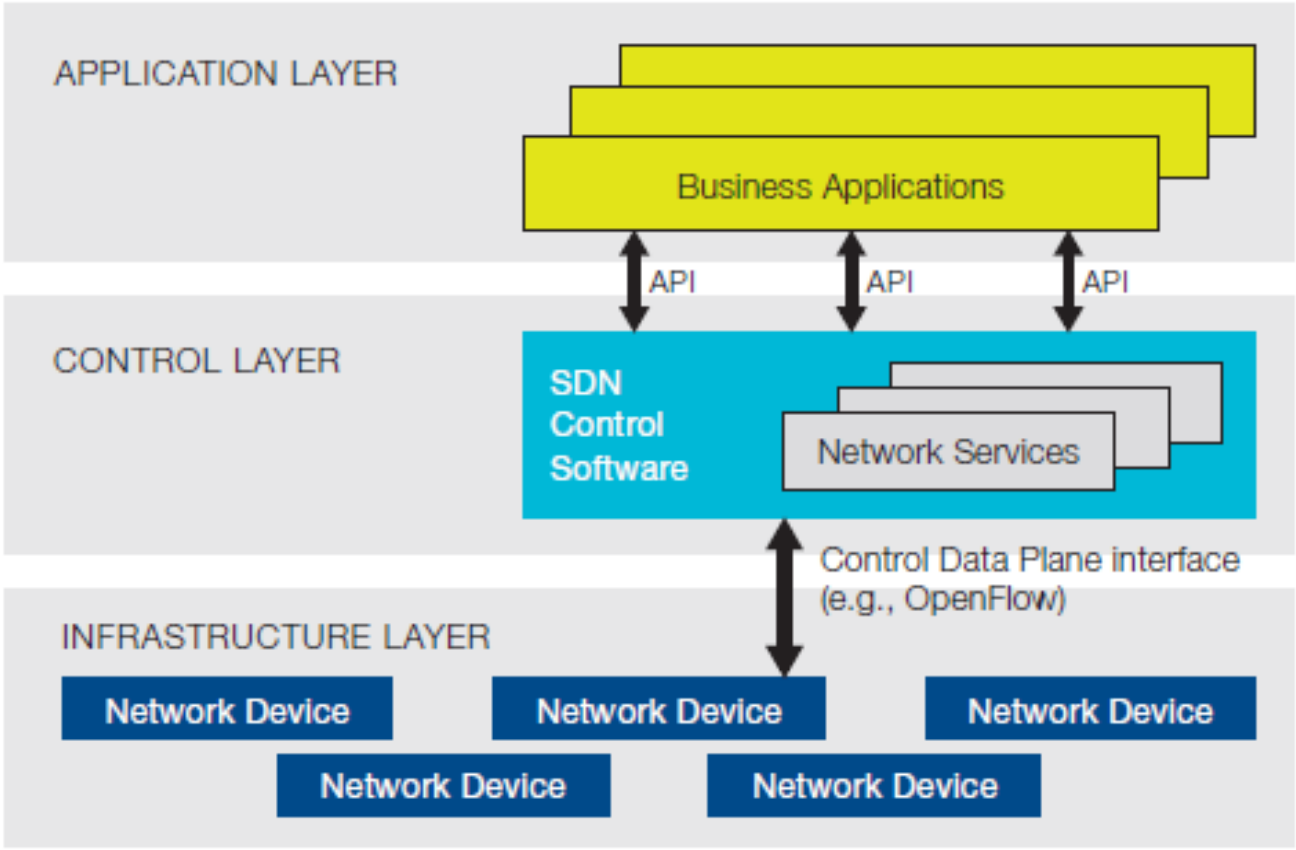


Software Defined Networking
SDN

- SDN allows network administrators to manage network services through abstraction of lower level functionality. This is done by decoupling the system that makes decisions about where traffic is sent (the **control plane**) from the underlying systems that forwards traffic to the selected destination (the **data plane**).
- SDN requires some method for the control plane to communicate with the data plane.
  - One such mechanism, **OpenFlow**, is often misunderstood to be equivalent to SDN, but other mechanisms could also fit into the concept.
  - The Open Networking Foundation (ONF) was founded to promote SDN and OpenFlow

[http://en.wikipedia.org/wiki/Software-defined_networking]

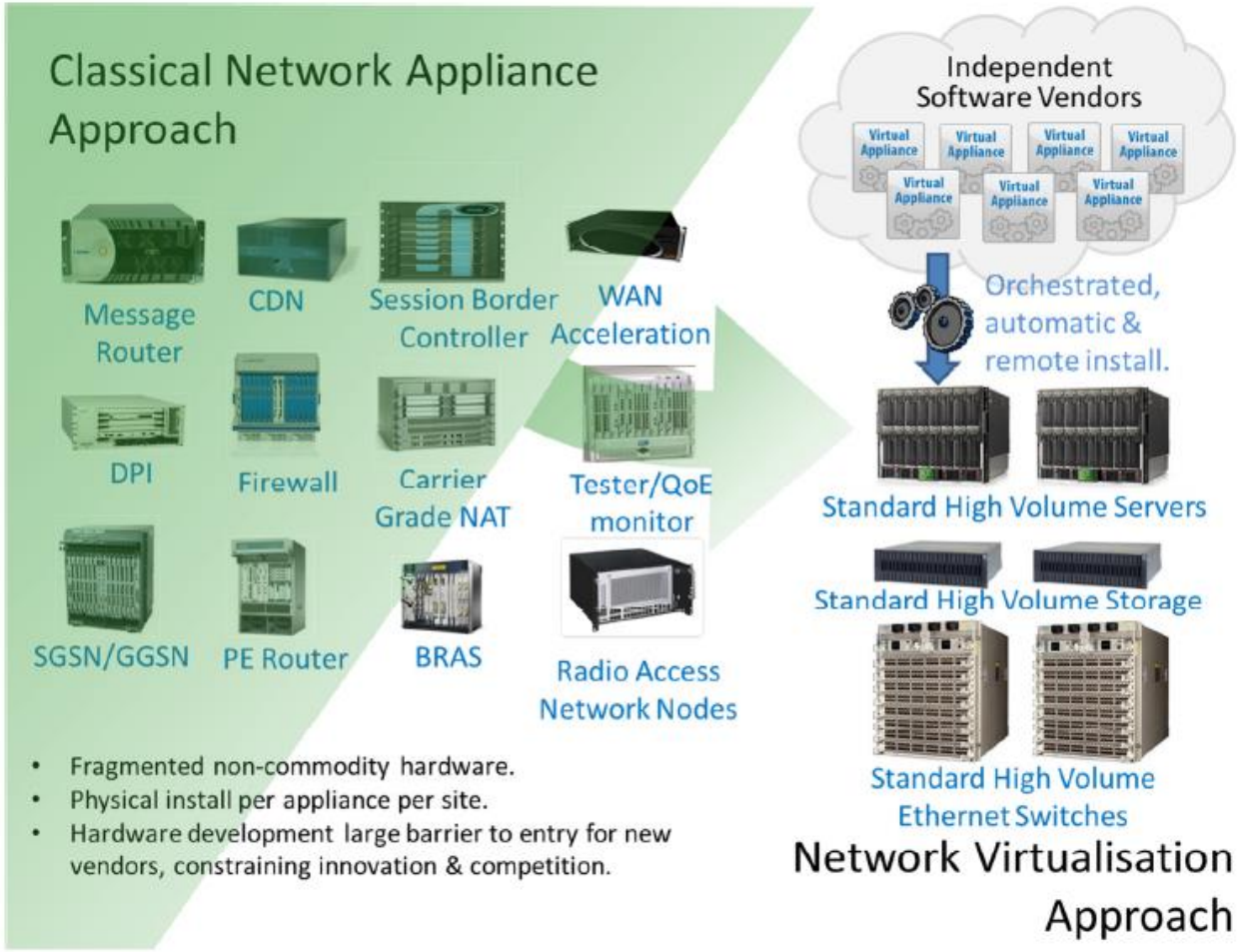
SDN
Architektur



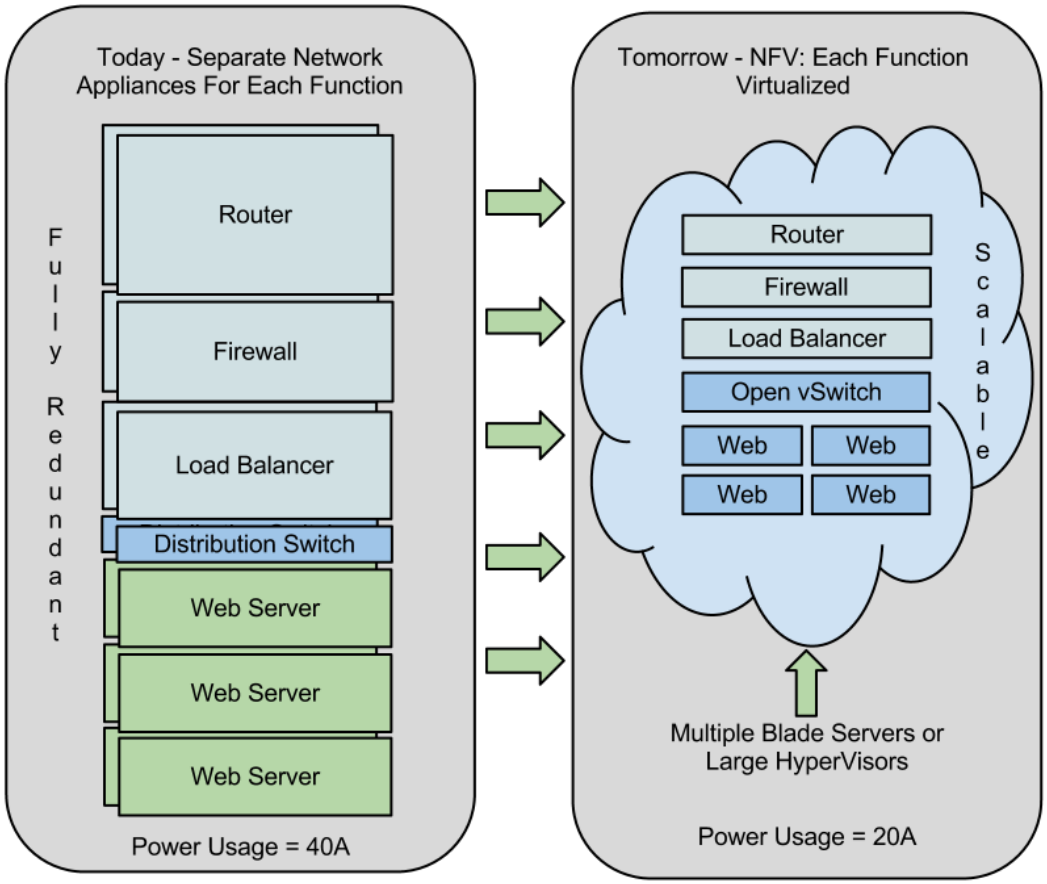
NFV
Definition

- Network Functions Virtualisation aims to transform the way that network operators architect networks by evolving standard IT virtualisation technology to consolidate many network equipment types onto industry standard high volume servers, switches and storage, which could be located in Datacentres, Network Nodes and in the end user premises.
- It involves the implementation of network functions in software that can run on a range of industry standard server hardware, and that can be moved to, or instantiated in, various locations in the network as required, without the need for installation of new equipment.

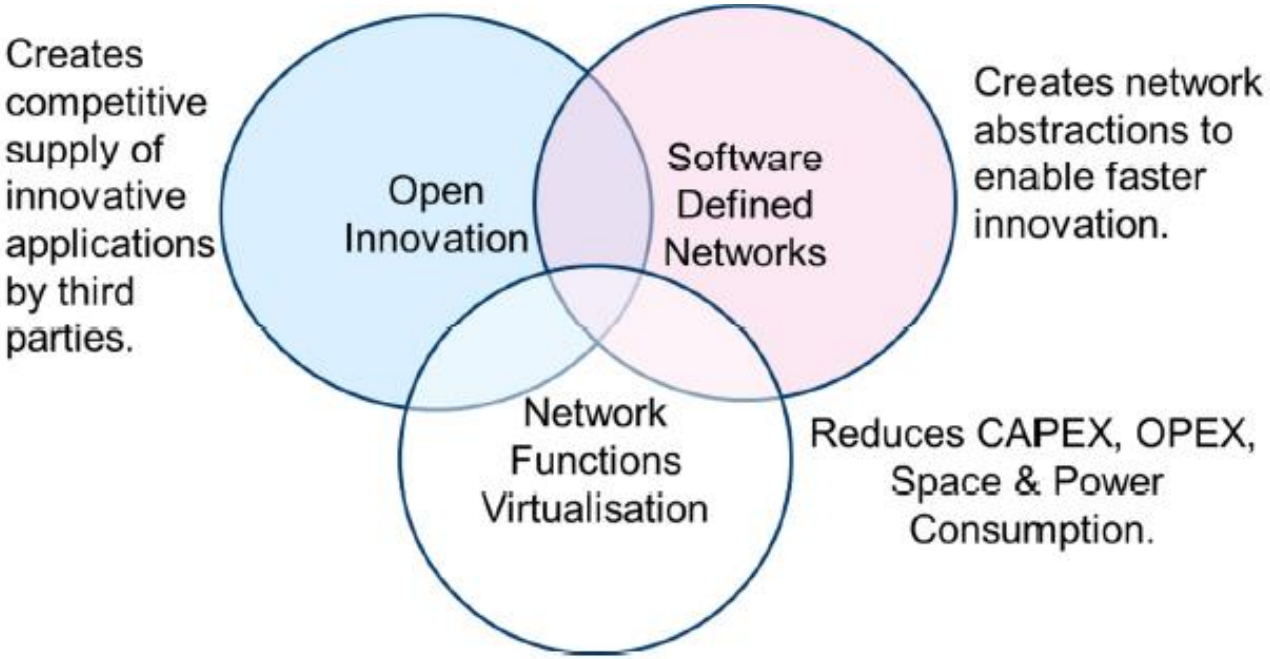
Network Functions Virtualisation NFV



NFV


Today - Separate Network Appliances For Each Function
Fully Redundant
Router
Firewall
Load Balancer
Distribution Switch
Web Server
Web Server
Web Server
Power Usage = 40A
Tomorrow - NFV: Each Function Virtualized
Router
Firewall
Load Balancer
Open vSwitch
Web
Web
Web
Web
Scalable
Multiple Blade Servers or Large HyperVisors
Power Usage = 20A

NFV vs. SDN


Creates competitive supply of innovative applications by third parties.
Open Innovation
Software Defined Networks
Creates network abstractions to enable faster innovation.
Network Functions Virtualisation
Reduces CAPEX, OPEX, Space & Power Consumption.

NFV
Beispiele

- Switching elements: BNG, CG-NAT, routers.
- Mobile network nodes: HLR/HSS, MME, SGSN, GGSN/PDN-GW, RNC, Node B, eNode B.
- Tunnelling gateway elements: IPSec/SSL VPN gateways.
- Traffic analysis: DPI, QoE measurement.
- Service Assurance, SLA monitoring, Test and Diagnostics.
- Converged and network-wide functions: AAA servers, policy control and charging platforms.
- Application-level optimisation: CDNs, Cache Servers, Load Balancers, Application Accelerators.
- Security functions: Firewalls, virus scanners, intrusion detection systems, spam protection.

References

- Advanced x86: Virtualization with VT-x Part 1, David Weinstein
- Introduction to Virtual Machines, Carl Waldspurger (SB SM ’89, PhD ’95), VMware R&D
- Virtualisierung als Grundlage Adaptiver IT, Vasu Chandrasekhara, Andreas Eberhart, Hewlett-Packard
- Cloud Infrastruktur Enabling Technologies: Virtualisierung, Martin Köhler