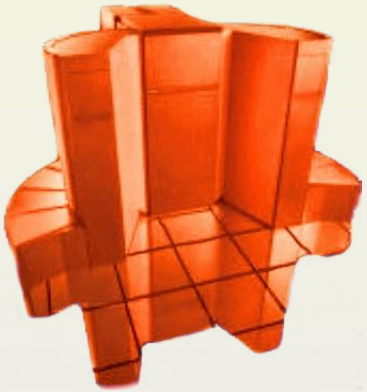
The Future of Virtualization

Felix Erkinger – wuxxin@gmail.com

The "anyOS" paradigm and its implications through virtualization

30 December 2005 – 22c3 Berlin



What is Virtualization ?

Virtualization is a framework or methodology of dividing the resources of a computer into multiple execution environments.

By using this technics its possible to share the resources of a computer to multiple operating systems all running at once.

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Field of Application for Virtualization

- Server consolidation
- Legacy Applications within legacy OS'es
- Secure isolated sandboxes for running untrusted applications
- Application mobility
- Testing and debugging environments
- Clean (single) service design
- Freedom of choice in using multiple os at once
- Soft user migration path
- Virtualization is fun !

Definition of virtualizability

1. The method of executing nonprivileged instructions must be equivalent in both privileged and user mode
2. There must be a protection system or an address translation system to protect the real system and any other VMs from each other
3. There must be a way to automatically signal the VMM when a VM attempts to execute a sensitive instruction
   - Instructions that attempt to change or reference the mode of the VM or the state of the machine
   - Instructions that read or change sensitive registers and/or memory locations
   - Instructions that reference the storage protection system, memory system, or address relocation system
   - All I/O instructions

Definition of virtualizability

1. The method of executing nonprivileged instructions must be equivalent in both privileged and user mode
2. There must be a protection system or an address translation system to protect the real system and any other VMs from each other
3. There must be a way to automatically signal the VMM when a VM attempts to execute a sensitive instruction
   - Instructions that attempt to change or reference the mode of the VM or the state of the machine
   - Instructions that read or change sensitive registers and/or memory locations
   - Instructions that reference the storage protection system, memory system, or address relocation system
   - All I/O instructions

Definition of virtualizability

1. The method of executing nonprivileged instructions must be equivalent in both privileged and user mode
2. There must be a protection system or an address translation system to protect the real system and any other VMs from each other
3. There must be a way to automatically signal the VMM when a VM attempts to execute a sensitive instruction
   - Instructions that attempt to change or reference the mode of the VM or the state of the machine
   - Instructions that read or change sensitive registers and/or memory locations
   - Instructions that reference the storage protection system, memory system, or address relocation system
   - All I/O instructions

Definition of virtualizability

1. The method of executing nonprivileged instructions must be equivalent in both privileged and user mode
2. There must be a protection system or an address translation system to protect the real system and any other VMs from each other
3. There must be a way to automatically signal the VMM when a VM attempts to execute a sensitive instruction
   - Instructions that attempt to change or reference the mode of the VM or the state of the machine
   - Instructions that read or change sensitive registers and/or memory locations
   - Instructions that reference the storage protection system, memory system, or address relocation system
   - All I/O instructions

Definition of virtualizability

1. The method of executing nonprivileged instructions must be equivalent in both privileged and user mode
2. There must be a protection system or an address translation system to protect the real system and any other VMs from each other
3. There must be a way to automatically signal the VMM when a VM attempts to execute a sensitive instruction
   - Instructions that attempt to change or reference the mode of the VM or the state of the machine
   - Instructions that read or change sensitive registers and/or memory locations
   - Instructions that reference the storage protection system, memory system, or address relocation system
   - All I/O instructions

Definition of virtualizability

1. The method of executing nonprivileged instructions must be equivalent in both privileged and user mode
2. There must be a protection system or an address translation system to protect the real system and any other VMs from each other
3. There must be a way to automatically signal the VMM when a VM attempts to execute a sensitive instruction
   - Instructions that attempt to change or reference the mode of the VM or the state of the machine
   - Instructions that read or change sensitive registers and/or memory locations
   - Instructions that reference the storage protection system, memory system, or address relocation system
   - All I/O instructions

Definition of virtualizability

1. The method of executing nonprivileged instructions must be equivalent in both privileged and user mode
2. There must be a protection system or an address translation system to protect the real system and any other VMs from each other
3. There must be a way to automatically signal the VMM when a VM attempts to execute a sensitive instruction
   - Instructions that attempt to change or reference the mode of the VM or the state of the machine
   - Instructions that read or change sensitive registers and/or memory locations
   - Instructions that reference the storage protection system, memory system, or address relocation system
   - All I/O instructions

Status of the X86 Virtualization capabilities

- 17 privileged instructions do not trap in user mode, violating Requirement 3.
- All seventeen instructions violate either part B or part C of Requirement 3
- This makes the Intel x86 processor architecture non-virtualizable.

Status of the X86 Virtualization capabilities

- 17 privileged instructions do not trap in user mode, violating Requirement 3.
- All seventeen instructions violate either part B or part C of Requirement 3
- This makes the Intel x86 processor architecture non-virtualizable.

Status of the X86 Virtualization capabilities

- 17 privileged instructions do not trap in user mode, violating Requirement 3.
- All seventeen instructions violate either part B or part C of Requirement 3
- This makes the Intel x86 processor architecture non-virtualizable.

How to hack around those limitations ?

Technics used for virtualization

- hardwarechange (Intel VT-x, AMD Pacifica)
- full emulation
- dynamic recompilation
- dynamic scan before execute / binary rewriting
- full kernel porting / OS Emulation
- api Emulation (eg. wine)
- mikro kernel approch
- paravirtualising and fractional kernel porting

what's inside the box ?

- Protection management (Secure isolation)
  - reconciling the virtual and physical architecture
  - preventing vm from interfering with each other or the monitor
- Resource management (Partitioning, Quality of Service)
  - time multiplexed resources: cpu, network, disk bandwidth
  - space multiplexed resources: physical memory including paging support for a guest vm
- Checkpoint/(live)migration/recovery
- Near to native speed (all virtualization guys are quake players ...)
  - fast monitor(hypervisor) calls and switches between vm's
  - fast interrupt handling
  - Idle instruction support
  - chunk writes for i/o (Double ring buffers)

what's inside the box ?

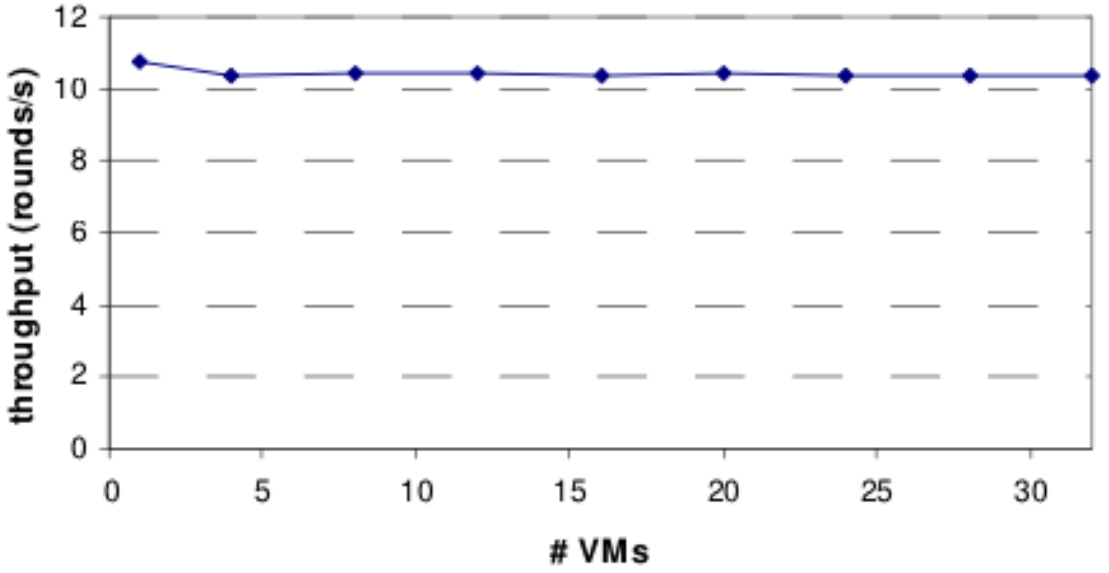
- Protection management (Secure isolation)
  - reconciling the virtual and physical architecture
  - preventing vm from interfering with each other or the monitor
- Resource management (Partitioning, Quality of Service)
  - time multiplexed resources: cpu, network, disk bandwidth
  - space multiplexed resources: physical memory including paging support for a guest vm
- Checkpoint/(live)migration/recovery
- Near to native speed (all virtualization guys are quake players ...)
  - fast monitor(hypervisor) calls and switches between vm's
  - fast interrupt handling
  - Idle instruction support
  - chunk writes for i/o (Double ring buffers)

what's inside the box ?

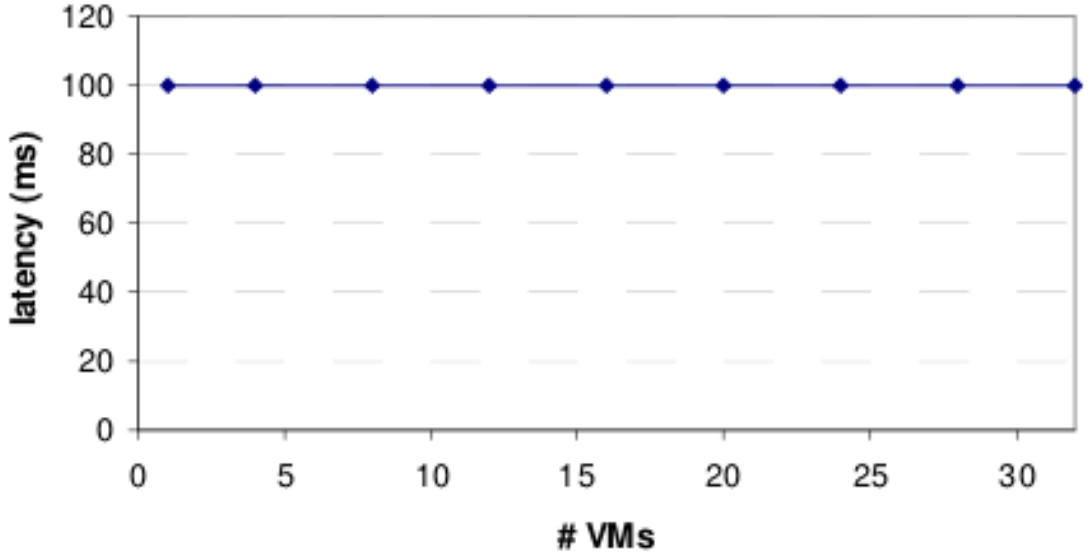
- Protection management (Secure isolation)
  - reconciling the virtual and physical architecture
  - preventing vm from interfering with each other or the monitor
- Resource management (Partitioning, Quality of Service)
  - time multiplexed resources: cpu, network, disk bandwidth
  - space multiplexed resources: physical memory including paging support for a guest vm
- Checkpoint/(live)migration/recovery
- Near to native speed (all virtualization guys are quake players ...)
  - fast monitor(hypervisor) calls and switches between vm's
  - fast interrupt handling
  - Idle instruction support
  - chunk writes for i/o (Double ring buffers)

what's inside the box ?

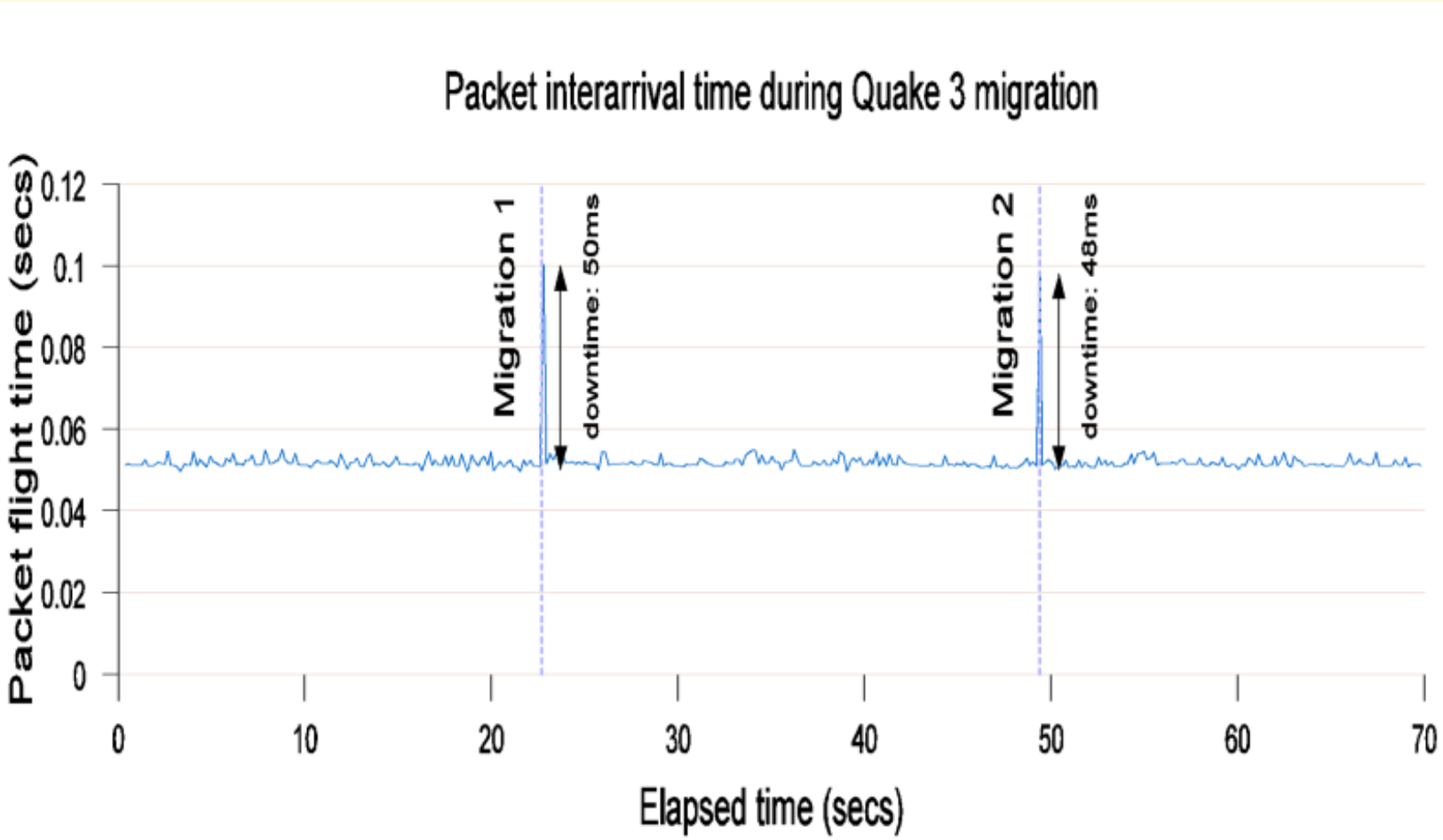
- Protection management (Secure isolation)
  - reconciling the virtual and physical architecture
  - preventing vm from interfering with each other or the monitor
- Resource management (Partitioning, Quality of Service)
  - time multiplexed resources: cpu, network, disk bandwidth
  - space multiplexed resources: physical memory including paging support for a guest vm
- Checkpoint/(live)migration/recovery
- Near to native speed (all virtualization guys are quake players ...)
  - fast monitor(hypervisor) calls and switches between vm's
  - fast interrupt handling
  - Idle instruction support
  - chunk writes for i/o (Double ring buffers)



(a) Quake II server throughput (per VM)



(b) Quake II server latency


Packet interarrival time during Quake 3 migration
Packet flight time (secs)
0.12
0.1
0.08
0.06
0.04
0.02
0
Migration 1
downtime: 50ms
Migration 2
downtime: 48ms
0
10
20
30
40
50
60
70
Elapsed time (secs)

Hardwarechange

Possibilities

Change the 17 privileged instructions to trap in user mode, and change there semantics

- Good: Easy to implement, if used as switchable extension also safe with "legacy" software
- Bad: Easy to implement...

Or...

Invent a hole new set of instructions and a new mode.
Intel VT-x, AMD Pacifica

Hardwarechange

Possibilities

Change the 17 privileged instructions to trap in user mode, and change there semantics

- Good: Easy to implement, if used as switchable extension also safe with "legacy" software
- Bad: Easy to implement...

Or...

Invent a hole new set of instructions and a new mode.
Intel VT-x, AMD Pacifica

Hardwarechange

Possibilities

Change the 17 privileged instructions to trap in user mode, and change there semantics

- Good: Easy to implement, if used as switchable extension also safe with "legacy" software
- Bad: Easy to implement...

Or...

Invent a hole new set of instructions and a new mode.
Intel VT-x, AMD Pacifica

Hardwarechange

Possibilities

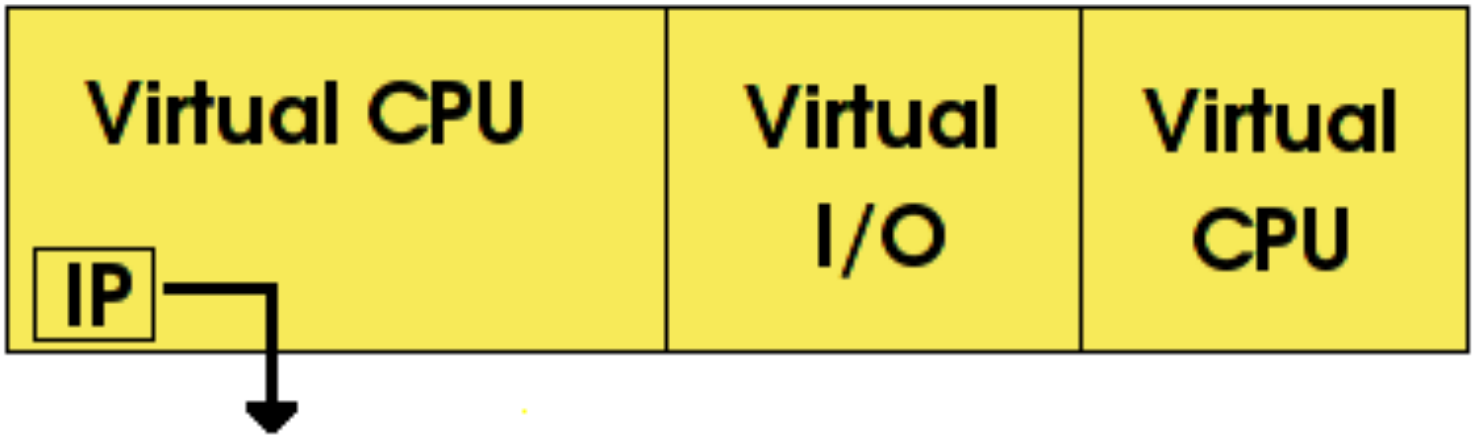
Change the 17 privileged instructions to trap in user mode, and change there semantics

- Good: Easy to implement, if used as switchable extension also safe with "legacy" software
- Bad: Easy to implement...

Or...

Invent a hole new set of instructions and a new mode.
Intel VT-x, AMD Pacifica

Full Emulation



- Fetch & Translate Opcode & Execute Opcode (as function)
- Update Virtual Device States

- Good: Multiple architectures (cross architecture emulation), good debugging capabilities
- Bad: Very slow, need nearly complete architecture emulated (including bios, I/O, ..)

Dynamic scan before execute / binary rewriting

code currently examined

```
jnz done
:bigger
mov ax,[sp-2]
smsw
:done
ret 2
```

- Branch
- sensitive Instruction
- function return

- Good: quite fast
- Bad: very dirty, need to support selfmodifiying code

Dynamic recompilation

```
...0E 7F B2 00 04 ...
```

for all opcodes

- Analyse opcode
- Translate into small virtual opcodes (c instructions)
- keep care of virtualized infrastructure
- Translate virtual opcodes into native machine code

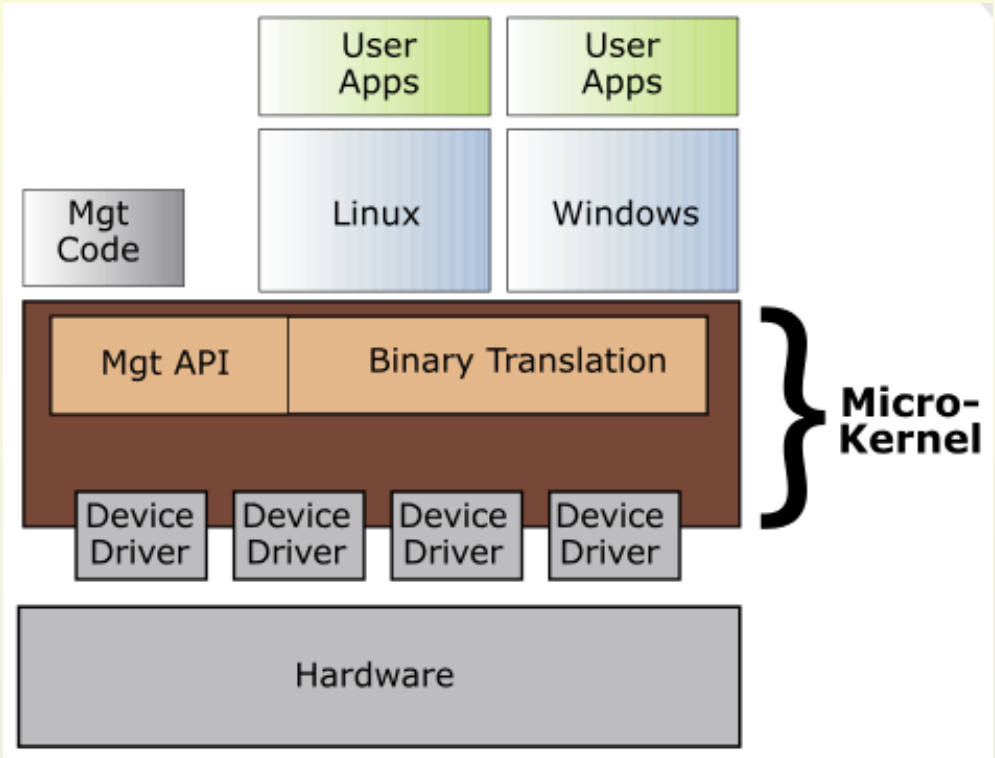
- Good: quite fast (depending on the implemention faster than binary rewriting), Multiple architectures (cross architecture recompliation), only need usermode support
- Bad: delicate, need to support selfmodifying code

API-Emulation, OS-Emulation

- OS-Emulation
  - All hardware access are mapped to functions, all os calls are remapped to real os calls
- API-Emulation
  - All function calls are mapped to the corresponding emulation functions

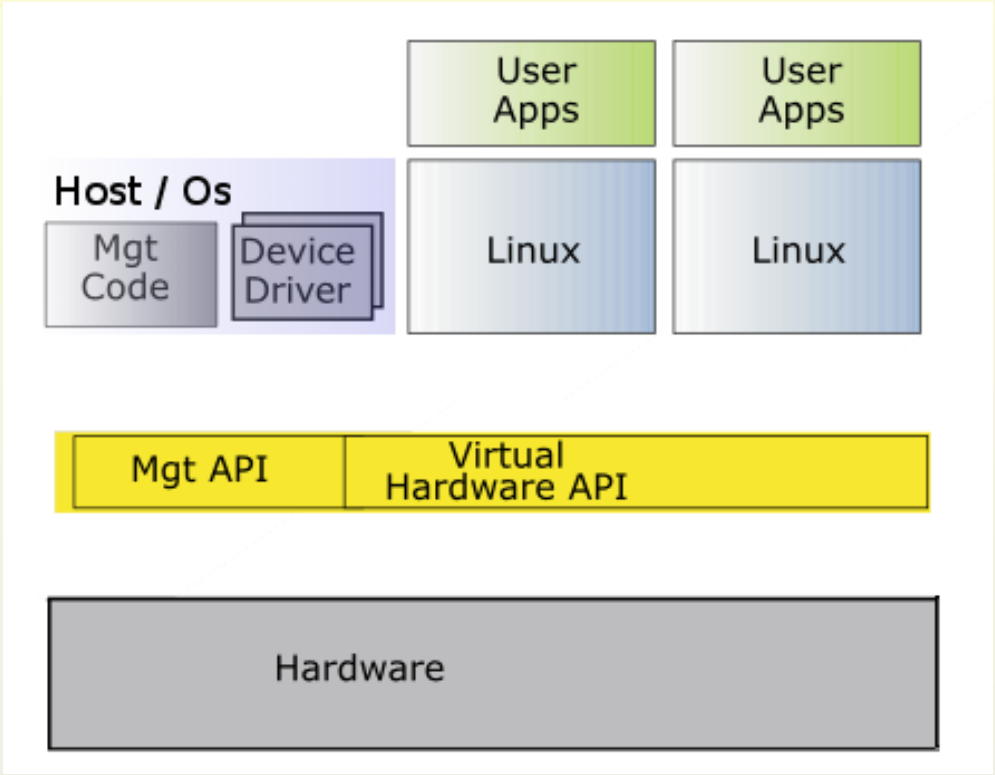
- Good: quite fast, userspace task in general
- Bad: hard to secure, needs to be "bug compatible" (depending source availability this is either easy or very hard), need to emulate a lot of libraries, need kernel module for speed (OS-emulation)

Mikro-kernel approch



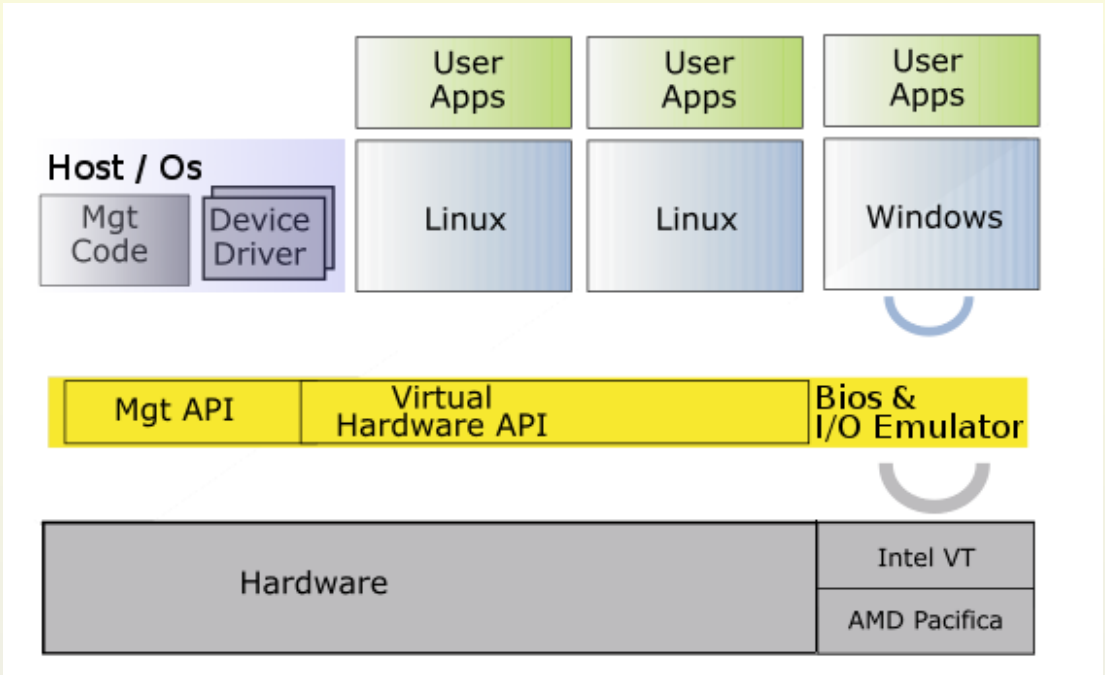
- Good: Clean design, fast
- Bad: Tons of code, very hard to maintain driver availability

Paravirtualization



- Good: very fast (near native), scaleable (linear overhead), easy hardware support
- Bad: complicated, needs modified kernel, hardware domain dependency

Paravirtualization & Fullvirtualization together



- Good: very fast (near native), scaleable (linear overhead), easy hardware support, able to run unmodified kernel
- Bad: complicated, need bios support, hardware domain dependency

Xen 2.0 Architektur Design

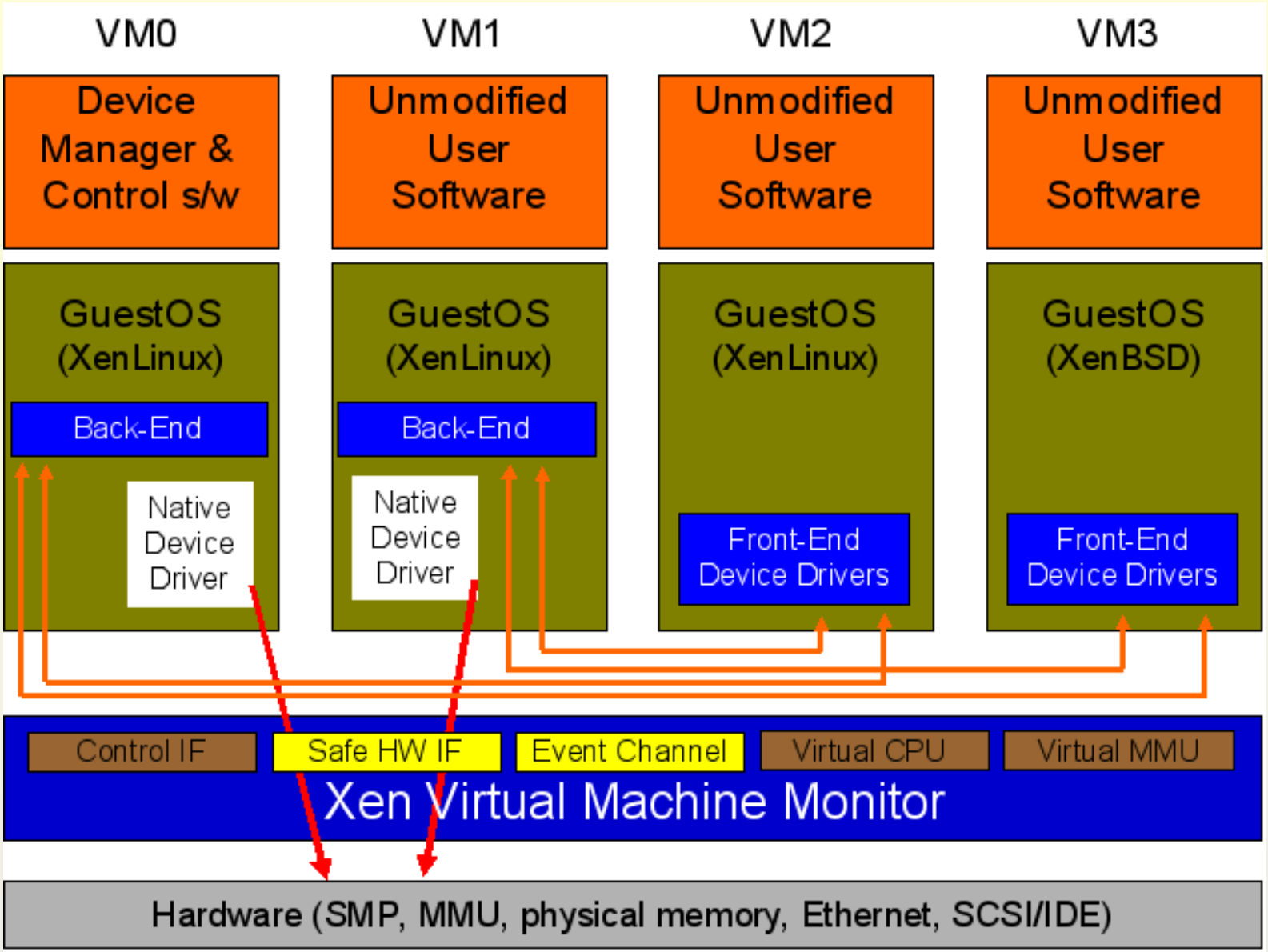


Figure: http://www.cl.cam.ac.uk/Research/SRG/netos/xen/architecture.html

Xen Performance

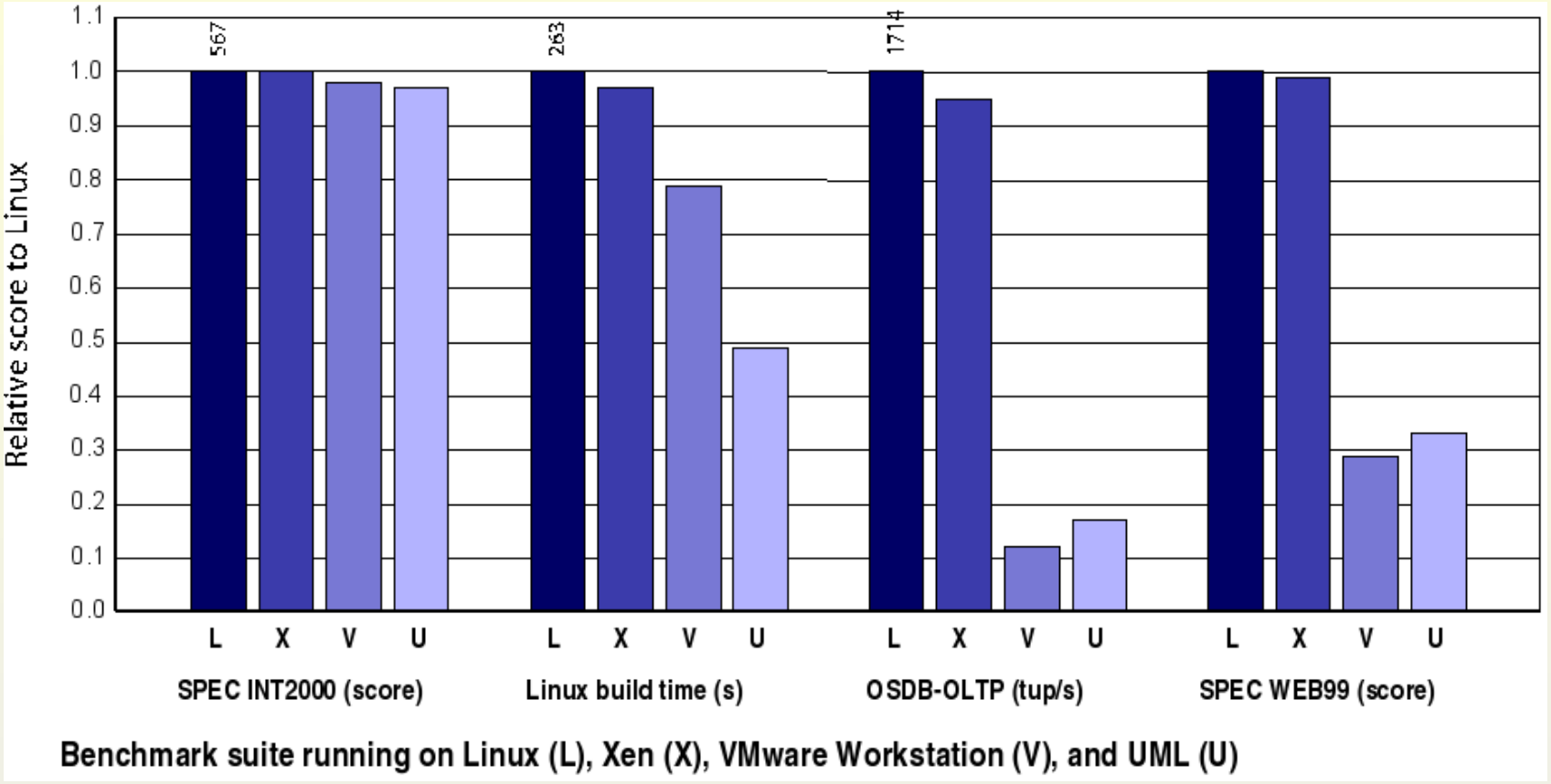


Figure: http://www.cl.cam.ac.uk/Research/SRG/netos/xen/architecture.html

Xen Performance

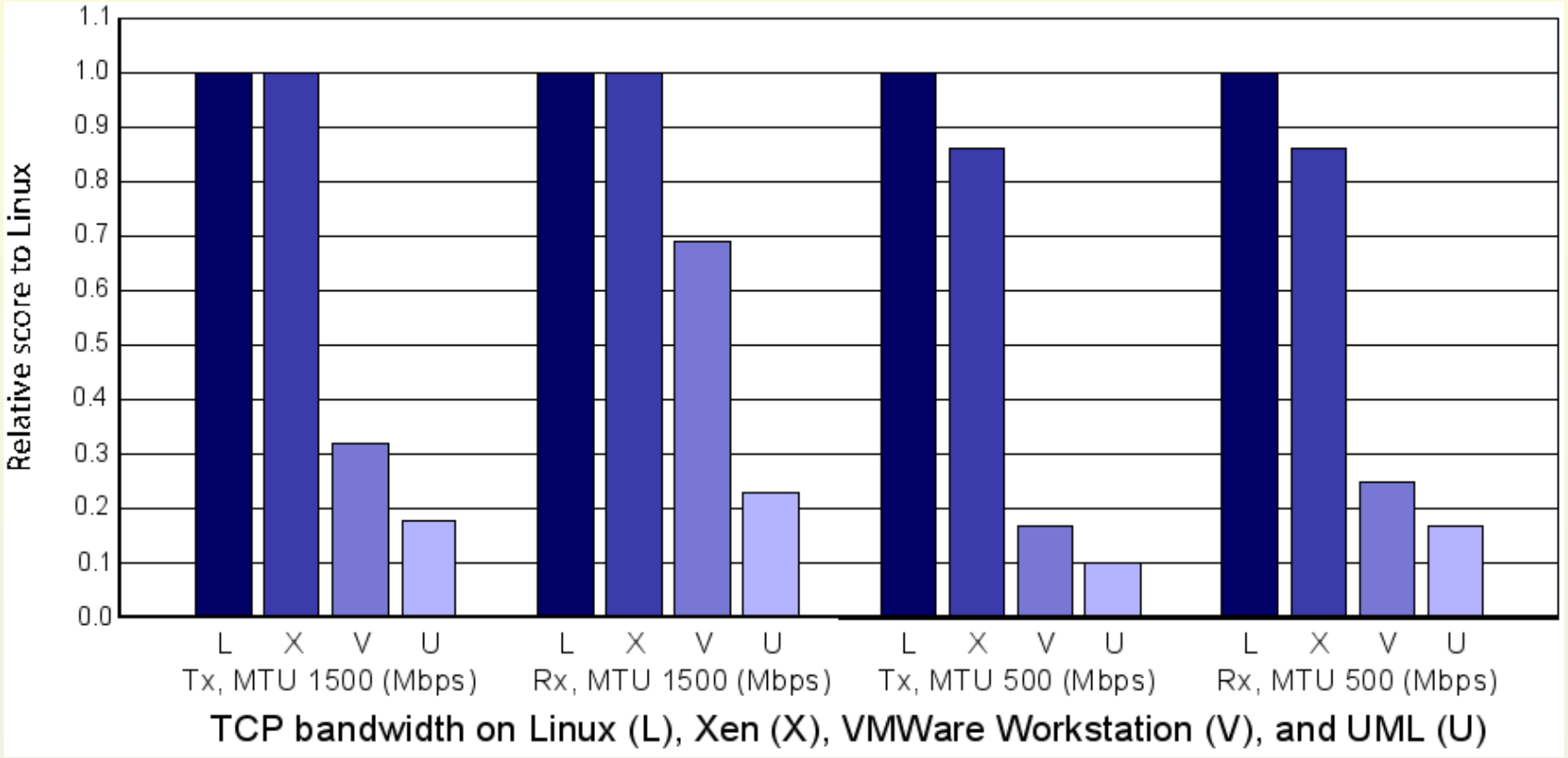


Figure: http://www.cl.cam.ac.uk/Research/SRG/netos/xen/architecture.html

Today's Virtualizationsoftware for PC's (x86 Platform)

- Emulation
  - Bochs
- Dynamic Recompilation
  - Qemu 0.7.x
- Kernel porting / OS Emulation
  - Usermodelinux V0.6x
- API-Emulation
  - Wine
- Paravirtualising and porting
  - Xen V2.0
- Fullvirtualization & Hybridtechnics
  - Microsoft Virtual PC 2004 / Virtual Server 2005
  - Vmware 5.x
  - Xen V3.0

The Future

- **soon:** Virtualization Monitor will become part of a standard "bootloader"
- **in scope:** Linux/*BSD & Reactos will be the "Any Os" with best possible hardware support.
- **whenever needed:** Virtualization will become a key technologie in privacy enforcement
  No more hassels with 'Copyright Enforcement' Technologies

The Future

- **soon:** Virtualization Monitor will become part of a standard "bootloader"
- **in scope:** Linux/*BSD & Reactos will be the "Any Os" with best possible hardware support.
- **whenever needed:** Virtualization will become a key technologie in privacy enforcement
  No more hassels with 'Copyright Enforcement' Technologies

The Future

- **soon:** Virtualization Monitor will become part of a standard "bootloader"
- **in scope:** Linux/*BSD & Reactos will be the "Any Os" with best possible hardware support.
- **whenever needed:** Virtualization will become a key technologie in privacy enforcement
  No more hassels with 'Copyright Enforcement' Technologies

further reading I

IA-32 Intel Architecture Software – Developer's Manuals
http://www.intel.com/design/pentium4/manuals/index_new.htm

Analysis of the Intel Pentium's Ability to Support a Secure Virtual Machine Monitor
http://denali.cs.washington.edu/relwork/papers/pentium.pdf

Xen and the Art of Virtualization
http://www.cl.cam.ac.uk/Research/SRG/netos/papers/2003-xensosp.pdf

further reading II

Keir Fraser, Steven Hand, Rolf Neugebauer, Ian Pratt, Andrew War eld, Mark Williamson.
Safe Hardware Access with the Xen Virtual Machine Monitor.
University of Cambridge Computer Laboratory, 2004
http://www.cl.cam.ac.uk/Research/SRG/netos/papers/2004-oasis-ngio.pdf

Intel Virtualization Technology
http://www.intel.com/technology/computing/vptech/

AMD "Pacifica" Virtualization Technology
http://enterprise.amd.com/Enterprise/serverVirtualization.aspx

Xen V2.0 & V3.0 http://www.xensource.com/

further reading III

- Usermodelinux V0.6x
  http://user-mode-linux.sourceforge.net/
- Denali http://denali.cs.washington.edu/
- Vmware Workstation 5.x http://www.vmware.com/
- Qemu 0.7.x
  http://fabrice.bellard.free.fr/qemu/
- Bochs IA-32 Emulator Project
  http://bochs.sourceforge.net/
- Microsoft Virtual PC 2004
  http://www.microsoft.com/windows/virtualpc/default.mspx

further reading IV

The Xen Mailing Lists
Xen-users – Xen user discussion
http://lists.xensource.com/cgi-bin/mailman/listinfo/xen-users

The Xen Mailing Lists
Xen-devel – Xen developer discussion
http://lists.xensource.com/cgi-bin/mailman/listinfo/xen-devel

Questions ?