Gemeinsamer Workshop der GI/ITG Fachgruppen „Betriebssysteme“ und „KuVS“

Virtualized IT infrastructures and their management

— LRZ-Bericht 2008-03 —

In Zusammenarbeit mit Vertretern des Munich Network Management Teams

Leibniz-Rechenzentrum Garching bei München

23.–24. Oktober 2008

Gemeinsamer Workshop der GI/ITG Fachgruppen „Betriebssysteme“ und „KuVS“

Virtualized IT infrastructures and their management

— LRZ-Bericht 2008-03 —

In Zusammenarbeit mit Vertretern des Munich Network Management Teams

Leibniz-Rechenzentrum Garching bei München

23.–24. Oktober 2008

Inhaltsverzeichnis

Keynote

Usage Scenarios I

Usage Scenarios II

Management Concepts

GI/VTG Fachgruppe Betriebssysteme und KuVS

23.10.2008 - 24.10.2008 , LRZ Garching

Donnerstag, 23.10.2008 / Beginn: 13:00 Uhr / Seminarraum 2, 2. OG, Raum-Nr. H.2.008

Timeslot	*Author*	*Titel*
13.00 - 13:15	<Organisator>	Welcome **Introduction**
13:15 - 15:00	Sven Graupner, HP Research	Virtualized IT Infrastructures and their Management
15.00 - 15:30	*Break*	
		Usage Scenarios I
15:30 - 16:15	Günther Aust, Fujitsu Siemens Computers	10 real challenges with virtual machines
16:15 - 17:00	Wolfgang Reichert IBM D Research & Development	Integration of Virtual System and Business Application Management Using Standardized Interfaces
17:00 - 18:00	<reserved>	<reserved>
18:00 - 18:30	(offenes) Treffen der GI/VTG Fachgruppe "Betriebssysteme"	

Social Event about 19:30 o´clock (somewhere Garching)

Freitag, 24.10.2008 / Beginn: 09:00 Uhr / Seminarraum 2, 2. OG, Raum-Nr. H.2.008

Timeslot	*Author*	*Titel*
		Usage Scenarios II
09:00 - 09:45	Reinhard Hohberger, IBM Software Group	Virtualization Aspects of Web Application Servers
09:45 - 10:30	Andre´ Brinkmann et al, Uni Paderborn	Virtual Supercomputer for HPC and HTC
10:30 - 11:15	Janczyk/v.Suchodoletz, Uni Freiburg	Broadening the Battle-zone - Some Thoughts on the Extension of Grid/Cluster Computing into the Desktop and Pool PC Domain
11:15 - 12:00	*Break*	*(Brotzeit statt Mittagspause)*
		Management Concepts
12:00 - 12:45	Friebel/Biemueller, AMD Research	How to Deal with Lock Holder Preemption
12:45 - 13:30	Andreas Fischer et al, Uni Passau	Virtual Network Management with XEN
13:30 - 13.45		**Conclusion**
13:45	*End*	

Keynote

Virtualized IT Infrastructures and Their Management

Dr. Sven Graupner

Hewlett-Packard Laboratories
Palo Alto, USA

Extended abstract for the joint workshop on "Virtualized IT Infrastructures and Their Management", University of Munich, Garching, Oct 23-24, 2008.

Virtualization has moved from an emerging trend to a mainstream technology in IT that is widely adapted today. Advantages are apparent: denser and more streamlined IT infrastructure, higher levels of utilization and sharing, improved returns on infrastructure investment, power and real estate savings, to name the most prominent ones.

While virtualization technology has emerged quite far for the basic elements in IT (machines, storage and networks), management of virtualization has been turning into an increasing problem. Underestimated initially, virtualization comes at a price of increased management effort. Traditional skills and roles in IT management are not prepared for virtualized IT infrastructure. Unlike the established and separated roles for managing physical systems, networks and storage, virtualization is not owned by a specific role. It rather cuts across the three domains. Virtual elements are often not even considered as managed entities since they don't exist physically and are not inventoried. Consequently, management practices are often not applied to virtual entities such as change, configuration and release management processes. Operational management processes such as incident, problem or fault management also often remain unassigned for virtual entities for similar reasons. Perception of virtual entities must change in IT to become first-class managed entities to which the established processes apply as they do for physical entities.

If the management practice is not changed recognizing virtual entities at a full extend, phenomena such as virtual resource sprawl, intertwined dependencies between virtual and the physically shared entities on which they reside, intransparency and the uncontrolled build-up of virtualization stacks and silos are the consequence. Due to the fact that virtual resources, at the end, rely on sharing physical resources, estimating workloads, planning capacity and achieving predictable behavior is becoming much harder with little evidence or experience from past providing guidance.

This threat is increased by the fact that most established management systems are not prepared and of no help with dealing with virtual elements. One reason is that virtual entities can be created in an ad hoc manner and may only exist for a short time. They also may not be active all the time and rather exist as saved state which can be resumed any time to recreate a virtual entity. This leads, in many cases, to the fact that virtual entities cannot be discovered, uniquely identified and registered in configuration and management databases. They are hence often unknown to management systems which rely on the information of those databases.

Fundamental concepts of management such as determining the existence and the identification of virtual entities are unsolved. Existing management systems hence remain unaware and incapable of managing virtual entities. Associations of virtual entities to underlying shared resources are also often not represented making fundamental management tasks such as monitoring implausible. While data can be collected using current monitoring systems, correct interpretation is often not possible because the context of the measurement or probing was not captured, such as the association of virtual entities to underlying physical entities at a given point in time.

Benefits of virtualization are undeniable. Virtualization can lead to a physical IT environment that is denser, more organized and more efficient. But the virtual world residing in that clean and organized physical world can easily become disaggregated, fragmented and unmaintained leading to high management overhead, unpredictable behavior and incalculatable risk of failure and chain effects.

While problems seem to be mounting, it does not mean they cannot be addressed and ultimately be solved by developing the appropriate management technology. IT vendors work with pressure on solutions to make existing IT management systems aware of virtualization, integrate management systems created by virtualization providers and also incorporate virtualization into established management practices such as ITIL. New initiatives are created such as the DMTF VMAN Initiative. Workshops and conferences are being held articulating the issues and educating customers for more comprehensive views on virtualization, its benefits and implications for management.

One interesting aspect with regard to virtualization is that it is not a new phenomenon per se. Virtualization has been introduced as a means of transparently sharing resources in the early days of computing and manifested itself as key technology operating systems. Operating systems fully automatically and transparently have been managing virtualized resources for many decades. Mechanisms and policies are well understood.

One reaction to this observation is to ask the question what can be learned from this technology in operating systems and carried over, adapted and applied in the context of data center infrastructure management. The fundamental underlying principles are the same: the need for transparently sharing resources (servers, storage or networks) for improving utilization and more effective use of resources available. The technologies are similar as well. Research has been conducted in Hewlett-Packard Laboratories over the past couple of years to learn from operating systems and carry over principles into lower-levels of operational IT infrastructure management to achieve higher degrees of automation, which includes the management of virtual resources.

Usage Scenarios I

INFORMATIONSTECHNISCHE GESELLSCHAFT IM VDE
Fachgruppe Betriebssysteme (6.1.4)

Gesellschaft für Informatik
Fachgruppe Betriebssysteme (BS)

We make sure

FUJITSU COMPUTERS SIEMENS

10 real Challanges with Virtual Machines

Best practice information gathered from successful virtualization projects

Günther Aust October 2008

What an SMB customer expects from VMware (prioritized)

We make sure

FUJITSU COMPUTERS SIEMENS

Midsize businesses

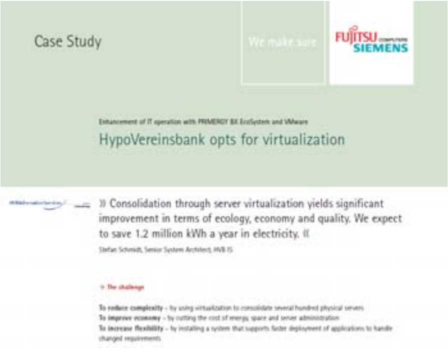
while also driven by the promise of consolidation and cost savings, see virtualization a chance to enable solutions what would otherwise be difficult and expensive

1. **Consolidation** —The level of savings is lower because of the scale of server deployments. By 2007, **40 percent of midsize businesses** will reduce their server population by at least 20 percent through
2. **Protection** — virtualization is coupled with **low-cost SAN** solutions. The cost and complexity of implementing disaster recovery is reduced.
3. **Agility** — Virtualization helps midsize businesses adapt server resources to address changes in workload demands. not at the same level of large enterprises. It makes it easier to bring up new services in remote branch offices
4. **Deployment** — Most midsize businesses have limited administrative resources; virtualization for them provides less effort and greater speed.
5. **Freedom of choice** — Virtualization allows midsize enterprises purchases based on competitive pricing without worrying about the overhead of supporting multiple vendors

Source: Gartner 2006

We make sure

Satisfied Customers are our Success



Case Study

We make sure

Enhancement of IT operation with PRIMERGY BX EcoSystem and VMware

HypoVereinsbank opts for virtualization

» Consolidation through server virtualization yields significant improvement in terms of ecology, economy and quality. We expect to save 1.2 million kWh a year in electricity. «

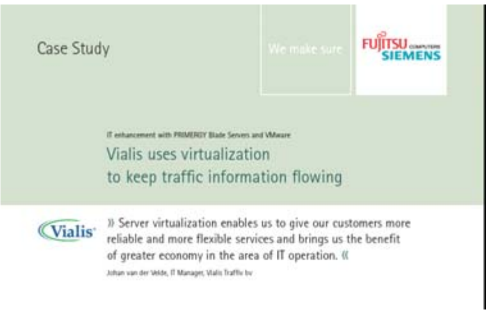
Stefan Schmidt, Senior System Architect, HVB IS

→ The challenge

To reduce complexity – by using virtualization to consolidate several hundred physical servers
To improve economy – by cutting the cost of energy, space and server administration
To increase flexibility – by installing a system that supports faster deployment of applications to handle changed requirements

We make sure

Satisfied Customers are our Success



Case Study

We make sure

IT enhancement with PRIMERGY Blade Servers and VMware

Vialis uses virtualization to keep traffic information flowing

» Server virtualization enables us to give our customers more reliable and more flexible services and brings us the benefit of greater economy in the area of IT operation. «

Johan van der Velde, IT Manager, Vialis Traffic bv

10 Real Challenges with Virtual Machines

What you should take care about #1 Uncontrolled Growth of Virtual Machines

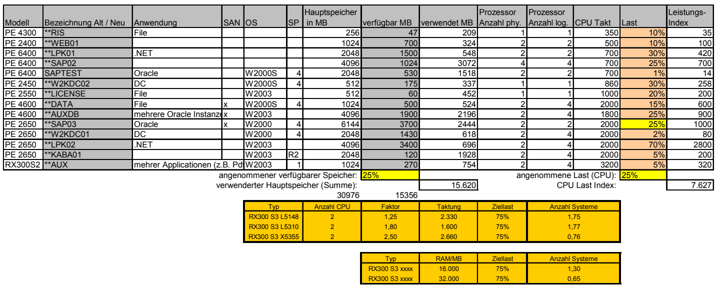
#1 STOP the Rabbits !!

- A new virtual machine is just a mouse click away!
  - □ But also a virtual machine needs to be managed and maintained
- Define strict rules for the provisioning of a new virtual machine
  - □ Strict cost models protect from uncontrolled requests for new VMs
- Continuously control the utilization of VMs (workload, period of utilization,….)
  - □ An unused VM wastes resources (e.g. memory)
- The lifecycle management of VMs needs new operational processes
  - □ VMware Lifecycle Manager could make your life much more easier

What you should take care about #2
Sizing is easy, but …

#2 Solid planning is key success factor in each virtualization project

- Carefully analyzing of the existing real infrastructure is essential and protects against nasty surprises during the implementation
- Virtualize the right applications!



Modell	Bezeichnung Alt / Neu	Anwendung	SAN	OS	SP	Hauptspeicher in MB	verfügbar MB	verwendet MB	Prozessor Anzahl phy.	Prozessor Anzahl log.	CPU Takt	Last	Leistungs-Index
PE 4300	**RIS	File				256	47	209	1	1	350	10%	35
PE 2400	**WEB01					1024	700	324	2	2	500	10%	100
PE 6400	**LPK01	.NET				2048	1500	548	2	2	700	30%	420
PE 6400	**SAP02					4096	1024	3072	4	4	700	25%	700
PE 6400	SAPTEST	Oracle		W2000S	4	2048	530	1518	2	2	700	1%	14
PE 2450	**W2KDC02	DC		W2000S	4	512	175	337	1	1	860	30%	258
PE 2550	**LICENSE	File		W2003		512	60	452	1	1	1000	20%	200
PE 4600	**DATA	File	x	W2000S	4	1024	500	524	2	4	2000	15%	600
PE 4600	**AUXDB	mehrere Oracle Instanze	x	W2003		4096	1900	2196	2	4	1800	25%	900
PE 2650	**SAP03	Oracle	x	W2000	4	6144	3700	2444	2	2	2000	25%	1000
PE 2650	**W2KDC01	DC		W2000	4	2048	1430	618	2	4	2000	2%	80
PE 2650	**LPK02	.NET		W2003		4096	3400	696	2	4	2000	70%	2800
PE 2650	**KABA01			W2003	R2	2048	120	1928	2	4	2000	5%	200
RX300S2	**AUX	mehrer Applicationen (z.B. Pdf		W2003	1	1024	270	754	2	4	3200	5%	320
						angenommener verfügbarer Speicher:	25%				angenommene Last (CPU):	25%	
						verwenderter Hauptspeicher (Summe):		15.620			CPU Last Index:		7.627
						30976	15356						

Typ	Anzahl CPU	Faktor	Taktung	Ziellast	Anzahl Systeme
RX300 S3 L5148	2	1,25	2.330	75%	1,75
RX300 S3 L5310	2	1,80	1.600	75%	1,77
RX300 S3 X5355	2	2,50	2.660	75%	0,76

Typ	RAM/MB	Ziellast	Anzahl Systeme
RX300 S3 xxxx	16.000	75%	1,30
RX300 S3 xxxx	32.000	75%	0,65

What you should take care about #2
Sizing is easy, but …

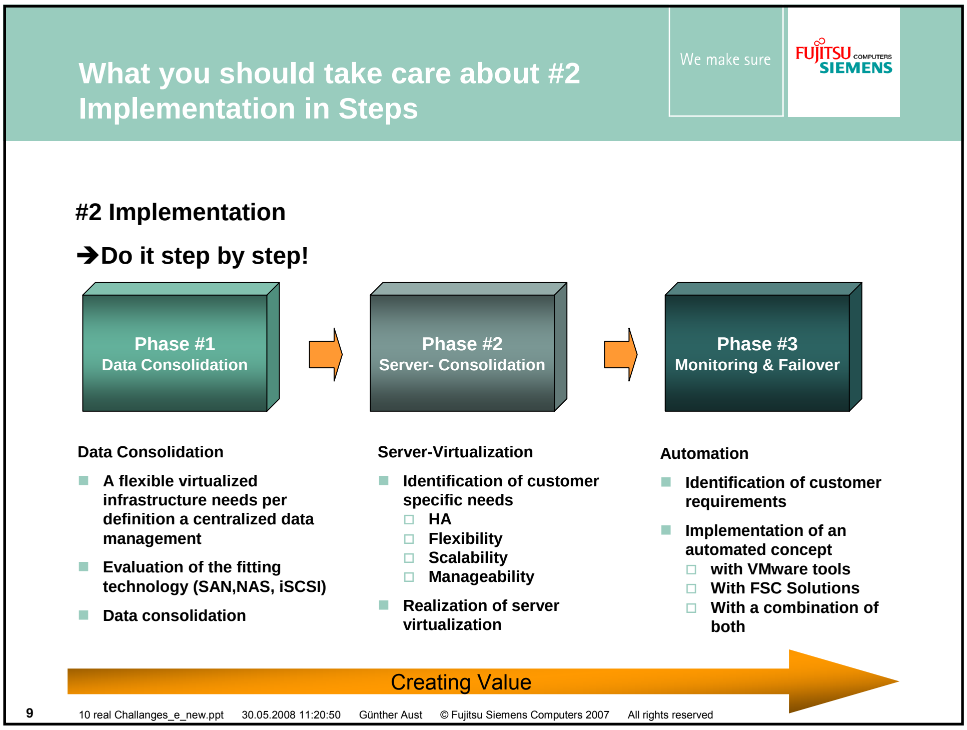
# 2 Solid planning is key success factor in each virtualization project

$$\Sigma_{\text{\# of target systems}} = \frac{\Sigma_S\,(N_{PS} \cdot F_{PS} \cdot C_{PS} \cdot \text{load}_{PS})}{N_{PH} \cdot F_{PH} \cdot C_{PH} \cdot \text{target load}_{Host}}$$

Σ_S	the sum of all servers
N_{PS}	# of processors of the servers to be consolidated
F_{PS}	processor frequency of the servers to be consolidated
C_{PS}, C_{PH}	CPU Factor
load_{PS}	measured load of the servers to be consolidated
N_{PH}	# of processors host

CPU Factor	
mono core:	1
dual core:	1,5
dual core LV:	1,25
quad core:	2,5
quad core LV:	1,75

- Just applicable for a limited number of servers
- Various tool from VMware, PlateSpin,… support & automate that process
- The migration of real servers into virtual servers should be supported by automated tool sets



We make sure

What you should take care about #2
Implementation in Steps

#2 Implementation

➔Do it step by step!

Data Consolidation

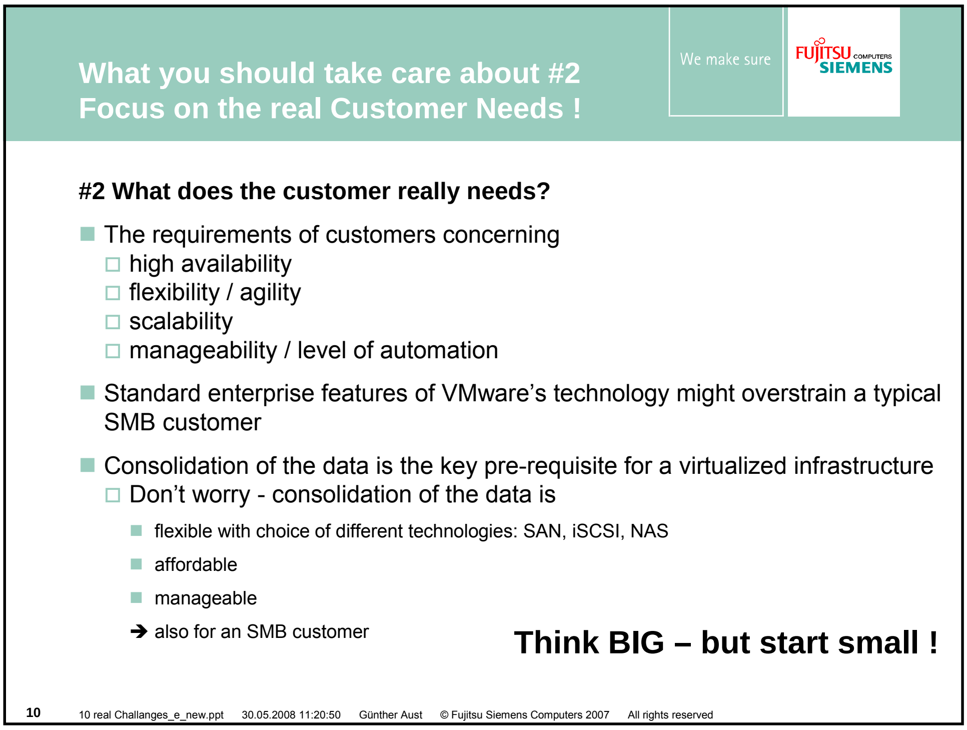
- A flexible virtualized infrastructure needs per definition a centralized data management
- Evaluation of the fitting technology (SAN,NAS, iSCSI)
- Data consolidation

Server-Virtualization

- Identification of customer specific needs
  - □ HA
  - □ Flexibility
  - □ Scalability
  - □ Manageability
- Realization of server virtualization

Automation

- Identification of customer requirements
- Implementation of an automated concept
  - □ with VMware tools
  - □ With FSC Solutions
  - □ With a combination of both



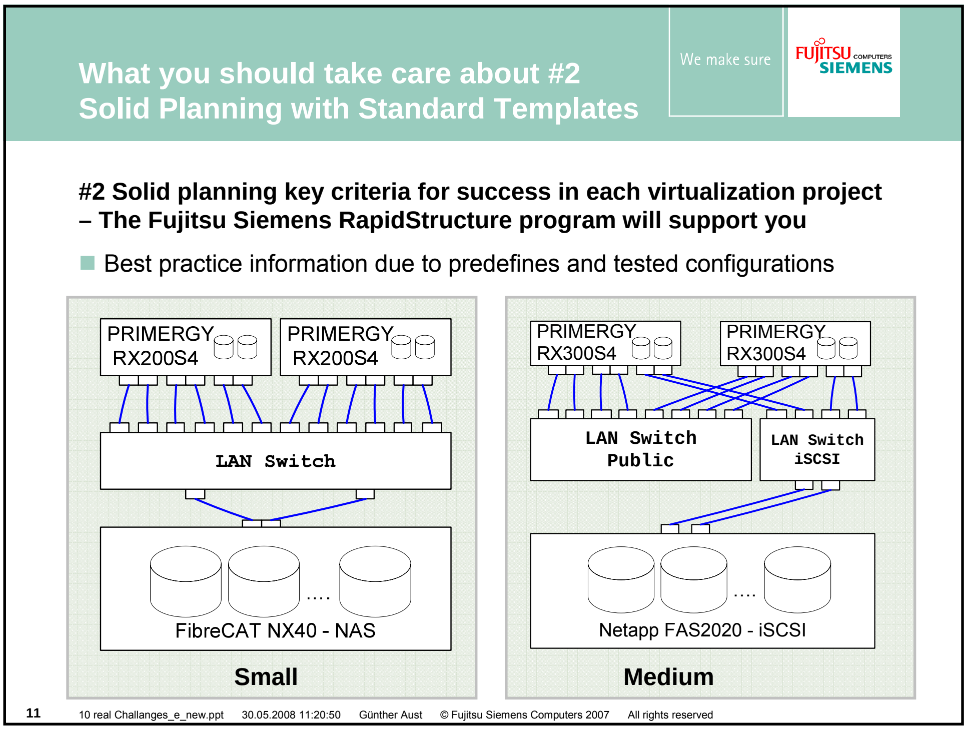
We make sure

What you should take care about #2
Focus on the real Customer Needs !

#2 What does the customer really needs?

- The requirements of customers concerning
  - □ high availability
  - □ flexibility / agility
  - □ scalability
  - □ manageability / level of automation
- Standard enterprise features of VMware’s technology might overstrain a typical SMB customer
- Consolidation of the data is the key pre-requisite for a virtualized infrastructure
  - □ Don’t worry - consolidation of the data is
    - flexible with choice of different technologies: SAN, iSCSI, NAS
    - affordable
    - manageable
    - ➔ also for an SMB customer

Think BIG – but start small !



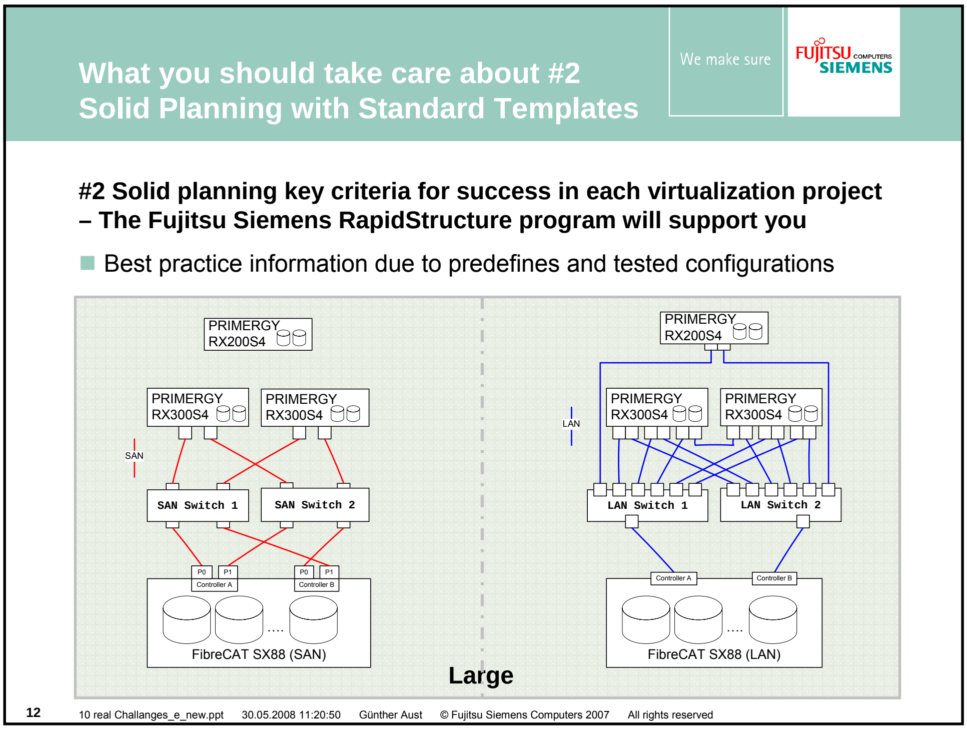
What you should take care about #2
Solid Planning with Standard Templates

We make sure

#2 Solid planning key criteria for success in each virtualization project – The Fujitsu Siemens RapidStructure program will support you

- Best practice information due to predefines and tested configurations

11 10 real Challanges_e_new.ppt 30.05.2008 11:20:50 Günther Aust © Fujitsu Siemens Computers 2007 All rights reserved



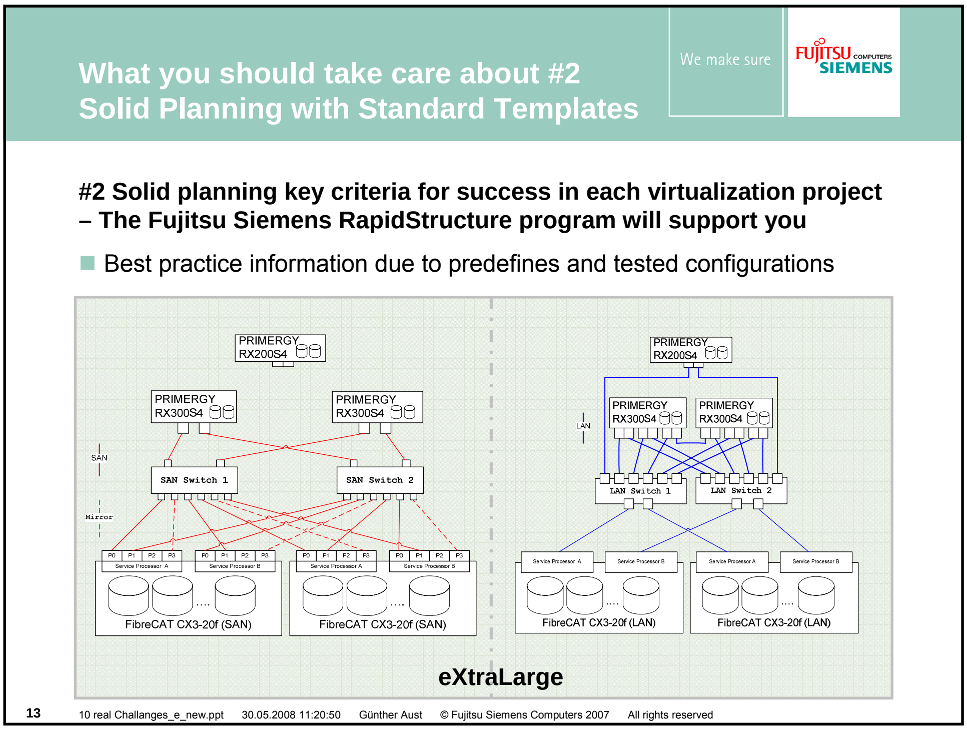
What you should take care about #2
Solid Planning with Standard Templates

We make sure

#2 Solid planning key criteria for success in each virtualization project – The Fujitsu Siemens RapidStructure program will support you

- Best practice information due to predefines and tested configurations

12 10 real Challanges_e_new.ppt 30.05.2008 11:20:50 Günther Aust © Fujitsu Siemens Computers 2007 All rights reserved

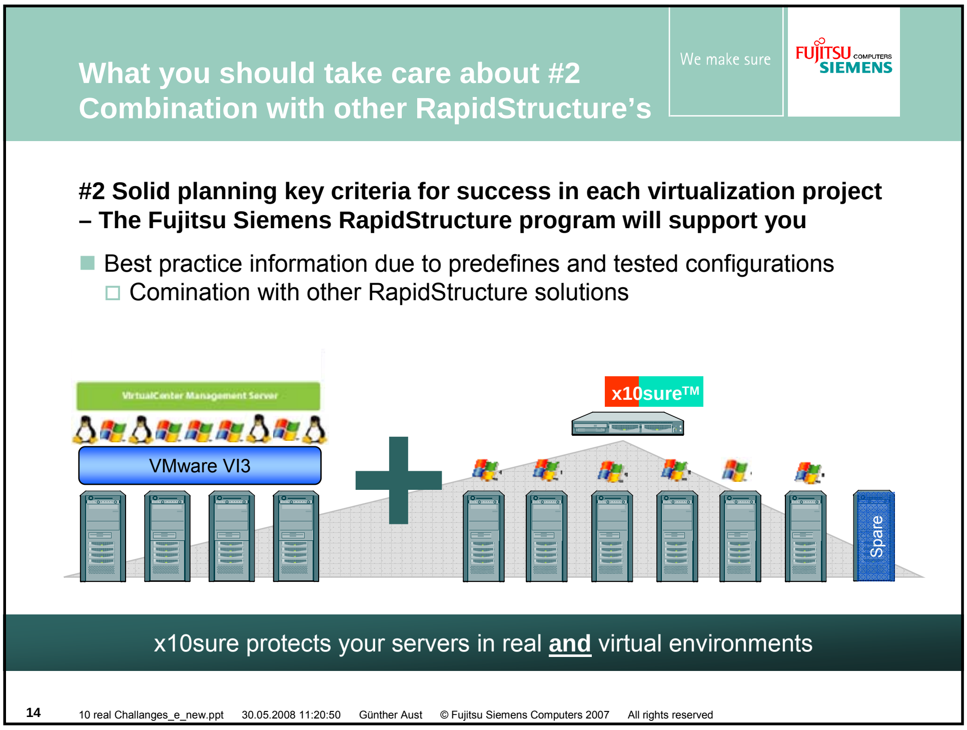


What you should take care about #2
Solid Planning with Standard Templates

We make sure

#2 Solid planning key criteria for success in each virtualization project – The Fujitsu Siemens RapidStructure program will support you

- Best practice information due to predefines and tested configurations



What you should take care about #2
Combination with other RapidStructure’s

We make sure

#2 Solid planning key criteria for success in each virtualization project – The Fujitsu Siemens RapidStructure program will support you

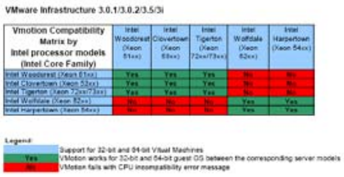
- Best practice information due to predefines and tested configurations
  - □ Comination with other RapidStructure solutions

x10sure protects your servers in real **and** virtual environments

What you should take care about #3
Homogeneity for Higher Flexibility

#3 The homogeneity of a virtualized platform

- Create a the most possible homogeny server platform
  - □ FC HBAs, NICs, CPU types
- More homogeneity provides higher flexibility and application mobility



VMware Infrastructure 3.0.1/3.0.2/3.5/3i

Vmotion Compatibility Matrix by Intel processor models (Intel Core Family)	Intel Woodcrest (Xeon 51xx)	Intel Clovertown (Xeon 53xx)	Intel Tigerton (Xeon 72xx/73xx)	Intel Wolfdale (Xeon 52xx)	Intel Harpertown (Xeon 54xx)
Intel Woodcrest (Xeon 51xx)	Yes	Yes	Yes	No	No
Intel Clovertown (Xeon 53xx)	Yes	Yes	Yes	No	No
Intel Tigerton (Xeon 72xx/73xx)	Yes	Yes	Yes	No	No
Intel Wolfdale (Xeon 52xx)	No	No	No	Yes	Yes
Intel Harpertown (Xeon 54xx)	No	No	No	Yes	Yes

Legend

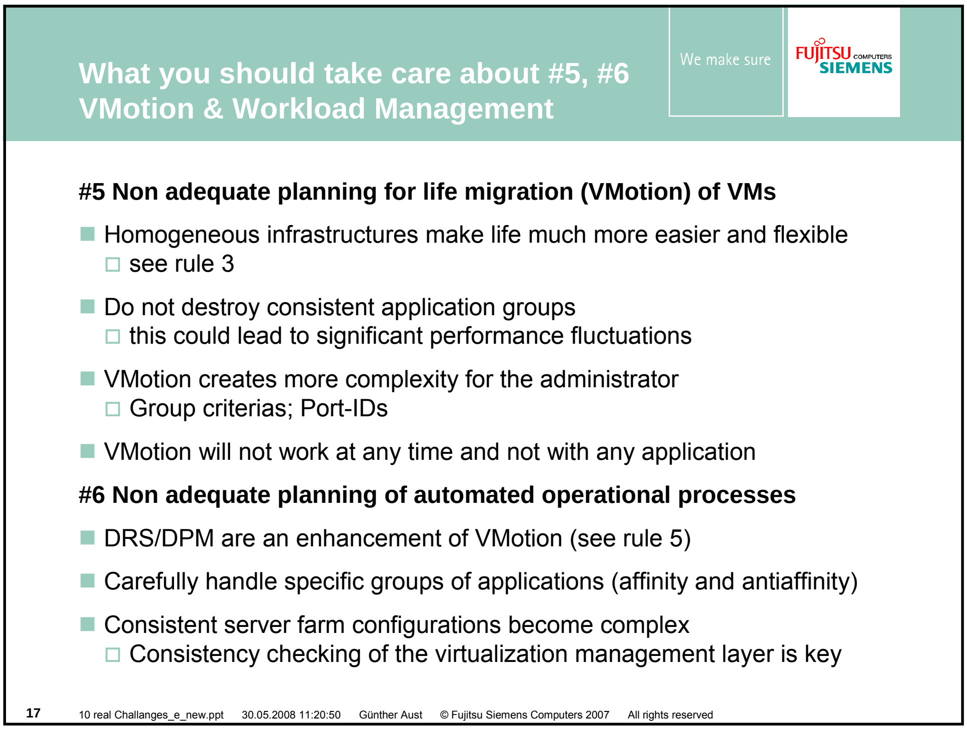
🟦	Support for 32-bit and 64-bit Vitual Machines
Yes	VMotion works for 32-bit and 64-bit guest OS between the corresponding server models
No	VMotion fails with CPU incompatibility error message

- Further aspects needs to be considered:
  - □ NIC Teaming (best practice)
  - □ Unique patch management creates simpler operational processes
  - □ Improved image compatibility for SAN boot scenarios

What you should take care about #4
Be aware of application-specific Conditions

#4 Keep attention to application-specific requirements

- VMs should be grouped for technical considerations
  - □ I/O, CPU, memory intensity
- But they should also be split or grouped for application specific considerations e.g.
  - □ A database server should be hosted within the same physical server as the related application server
  - □ Group related applications in dedicated affinity groups
  - □ Use dedicated ports in virtual switches



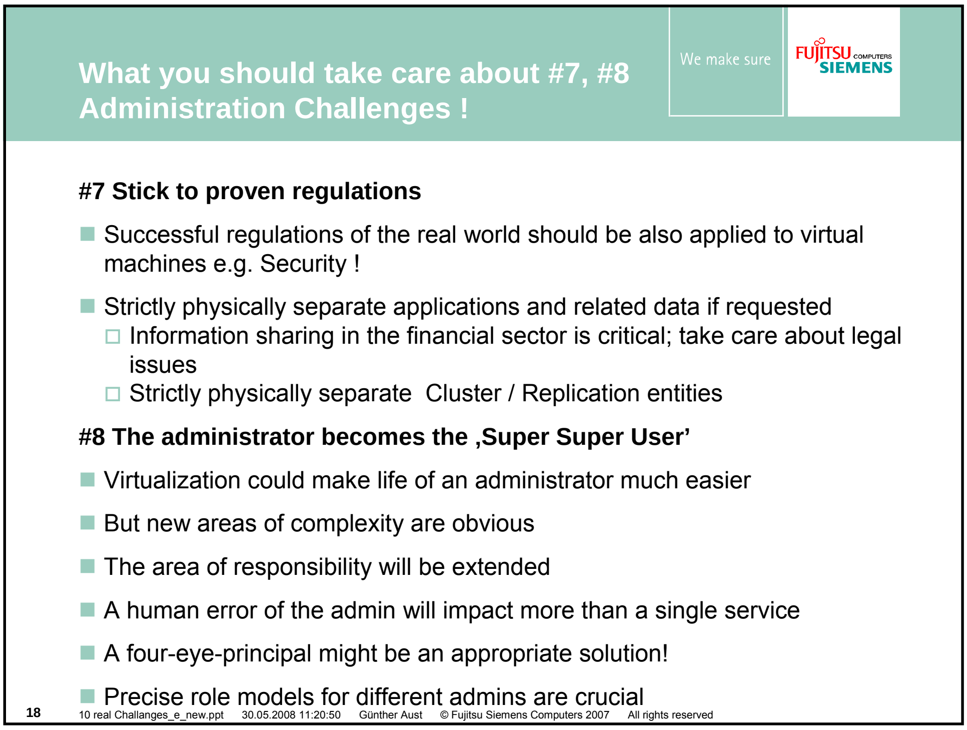
What you should take care about #5, #6 VMotion & Workload Management

#5 Non adequate planning for life migration (VMotion) of VMs

- Homogeneous infrastructures make life much more easier and flexible
  - □ see rule 3
- Do not destroy consistent application groups
  - □ this could lead to significant performance fluctuations
- VMotion creates more complexity for the administrator
  - □ Group criterias; Port-IDs
- VMotion will not work at any time and not with any application

#6 Non adequate planning of automated operational processes

- DRS/DPM are an enhancement of VMotion (see rule 5)
- Carefully handle specific groups of applications (affinity and antiaffinity)
- Consistent server farm configurations become complex
  - □ Consistency checking of the virtualization management layer is key



What you should take care about #7, #8 Administration Challenges !

#7 Stick to proven regulations

- Successful regulations of the real world should be also applied to virtual machines e.g. Security !
- Strictly physically separate applications and related data if requested
  - □ Information sharing in the financial sector is critical; take care about legal issues
  - □ Strictly physically separate Cluster / Replication entities

#8 The administrator becomes the ‚Super Super User'

- Virtualization could make life of an administrator much easier
- But new areas of complexity are obvious
- The area of responsibility will be extended
- A human error of the admin will impact more than a single service
- A four-eye-principal might be an appropriate solution!
- Precise role models for different admins are crucial

What you should take care about #9, #10
Political Impact and Isolated View

We make sure

#9 Do not underestimate the political impact in virtualization projects

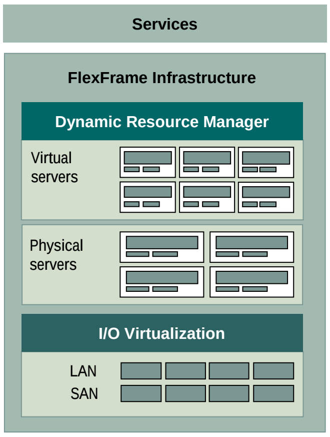
- Resource sharing, security concerns, mistrust and the fear to become overstrained from the end users side
- A transparent billing model normally helps
- If needed separate users on dedicated physical instances

#10 The isolated view of server virtualization

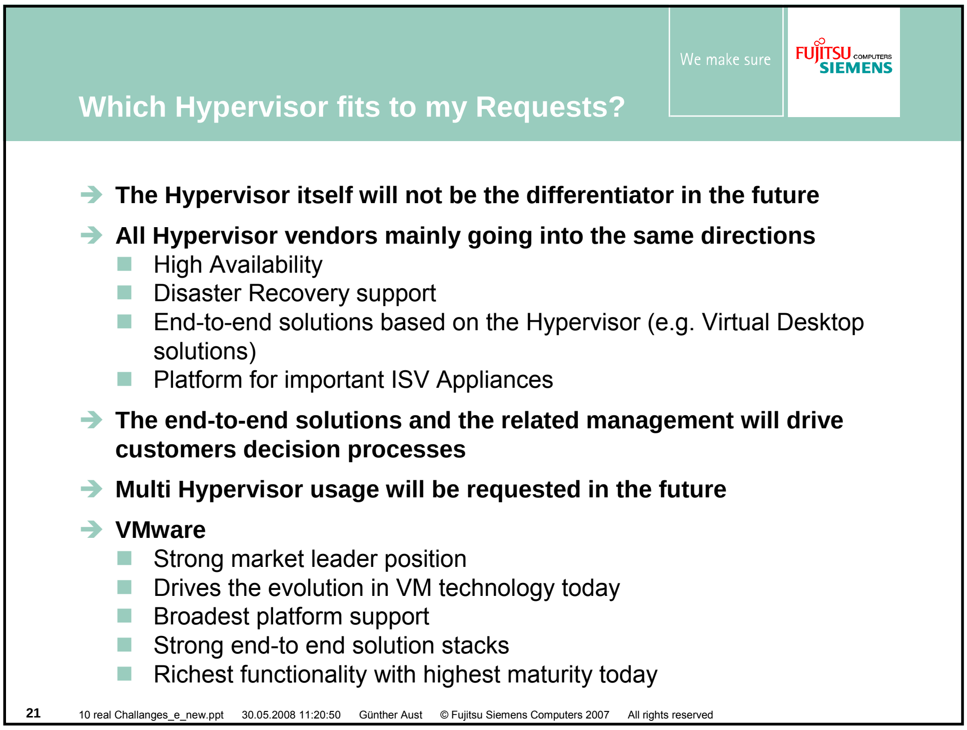
- Server virtualization alone is not the magic bullet
- It should be one building block of a holistic infrastructure optimization concept
- It should come along with other initiatives like I/O- and storage virtualization
- A common management of physical and virtual entities could significantly reduce complexity
- The answer of Fujitsu Siemens Computers is FlexFrame Infrastructure

The Answer of Fujitsu Siemens:
FlexFrame Infrastructure

We make sure



- Admin defines what is needed, the rest runs automatically
  - □ Allocate apps to real / virtual servers
  - □ Allocate LAN / SAN addresses
  - □ Automatic reaction (incl. instant DR) acc to defined regulations
- Benefits
  - □ Reduce complexity of virtualization
  - □ Speed of service delivery
  - □ Improved service quality
  - □ Lower investment and admin efforts
- Examples:
  - □ PRIMERGY BladeFrame
  - □ ServerView PAN



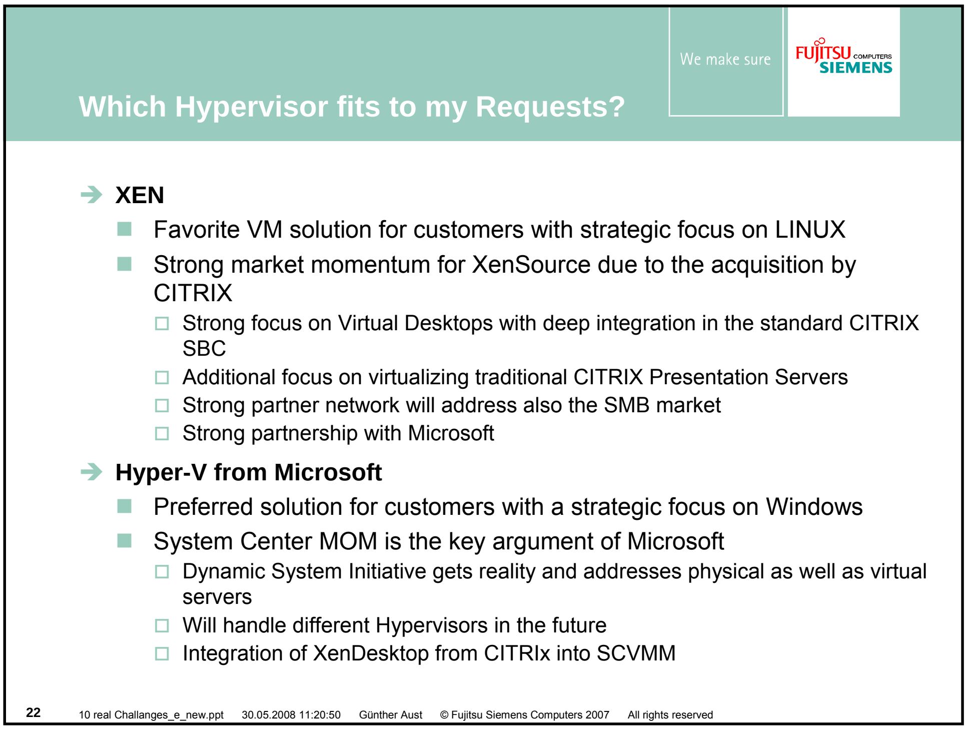
We make sure

FUJITSU COMPUTERS SIEMENS

Which Hypervisor fits to my Requests?

- ➔ **The Hypervisor itself will not be the differentiator in the future**
- ➔ **All Hypervisor vendors mainly going into the same directions**
  - High Availability
  - Disaster Recovery support
  - End-to-end solutions based on the Hypervisor (e.g. Virtual Desktop solutions)
  - Platform for important ISV Appliances
- ➔ **The end-to-end solutions and the related management will drive customers decision processes**
- ➔ **Multi Hypervisor usage will be requested in the future**
- ➔ **VMware**
  - Strong market leader position
  - Drives the evolution in VM technology today
  - Broadest platform support
  - Strong end-to end solution stacks
  - Richest functionality with highest maturity today

21 10 real Challanges_e_new.ppt 30.05.2008 11:20:50 Günther Aust © Fujitsu Siemens Computers 2007 All rights reserved



We make sure

FUJITSU COMPUTERS SIEMENS

Which Hypervisor fits to my Requests?

- ➔ **XEN**
  - Favorite VM solution for customers with strategic focus on LINUX
  - Strong market momentum for XenSource due to the acquisition by CITRIX
    - □ Strong focus on Virtual Desktops with deep integration in the standard CITRIX SBC
    - □ Additional focus on virtualizing traditional CITRIX Presentation Servers
    - □ Strong partner network will address also the SMB market
    - □ Strong partnership with Microsoft
- ➔ **Hyper-V from Microsoft**
  - Preferred solution for customers with a strategic focus on Windows
  - System Center MOM is the key argument of Microsoft
    - □ Dynamic System Initiative gets reality and addresses physical as well as virtual servers
    - □ Will handle different Hypervisors in the future
    - □ Integration of XenDesktop from CITRIx into SCVMM

22 10 real Challanges_e_new.ppt 30.05.2008 11:20:50 Günther Aust © Fujitsu Siemens Computers 2007 All rights reserved

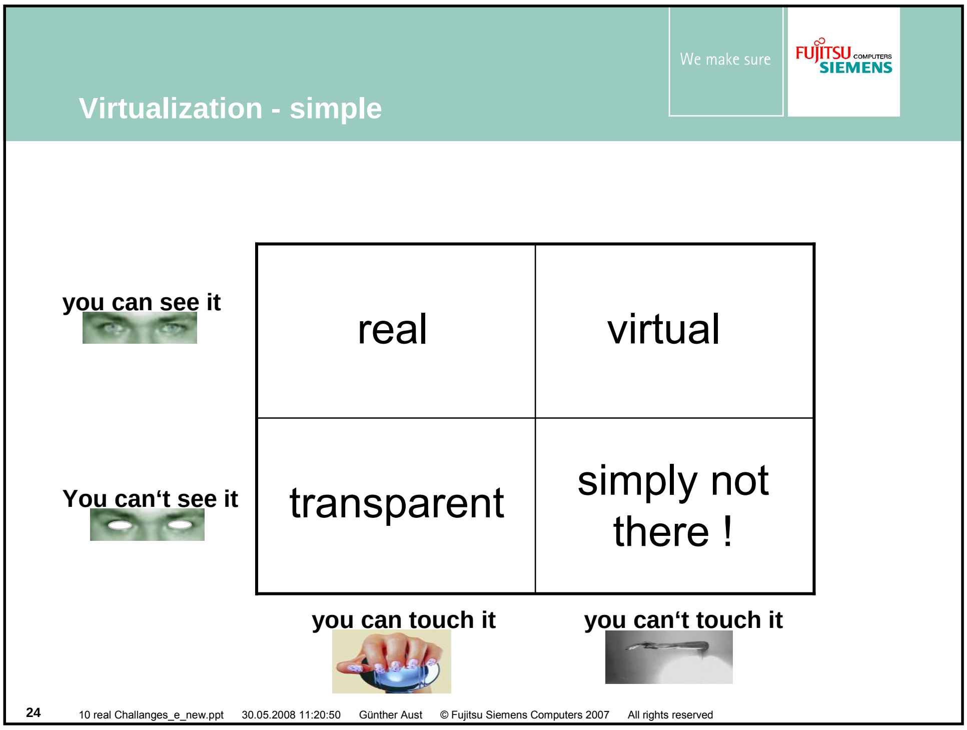
ITG INFORMATIONSTECHNISCHE GESELLSCHAFT IM VDE
Fachgruppe Betriebssysteme (6.1.4)

Gesellschaft für Informatik
Fachgruppe Betriebssysteme (BS)

We make sure

FUJITSU COMPUTERS SIEMENS

Questions?

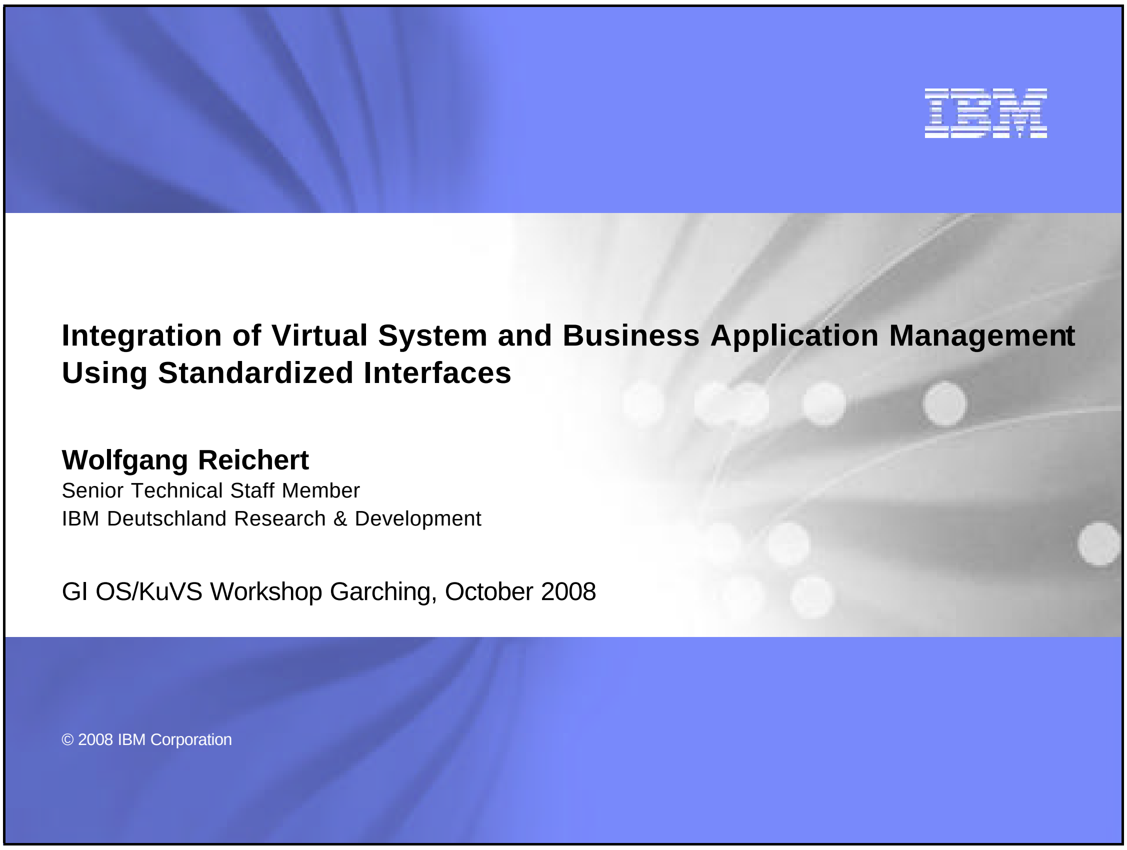


We make sure

FUJITSU COMPUTERS SIEMENS

Virtualization - simple

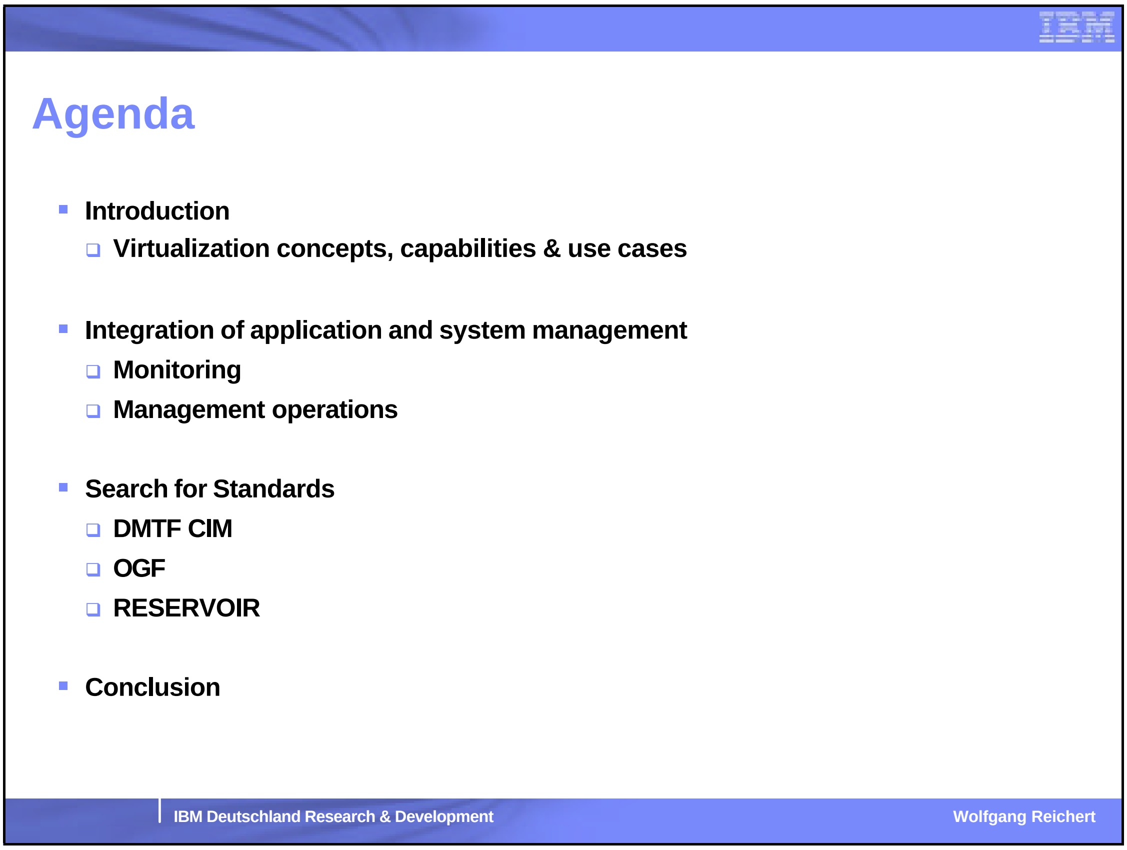
	you can touch it	you can‘t touch it
you can see it	real	virtual
You can‘t see it	transparent	simply not there !



Integration of Virtual System and Business Application Management Using Standardized Interfaces

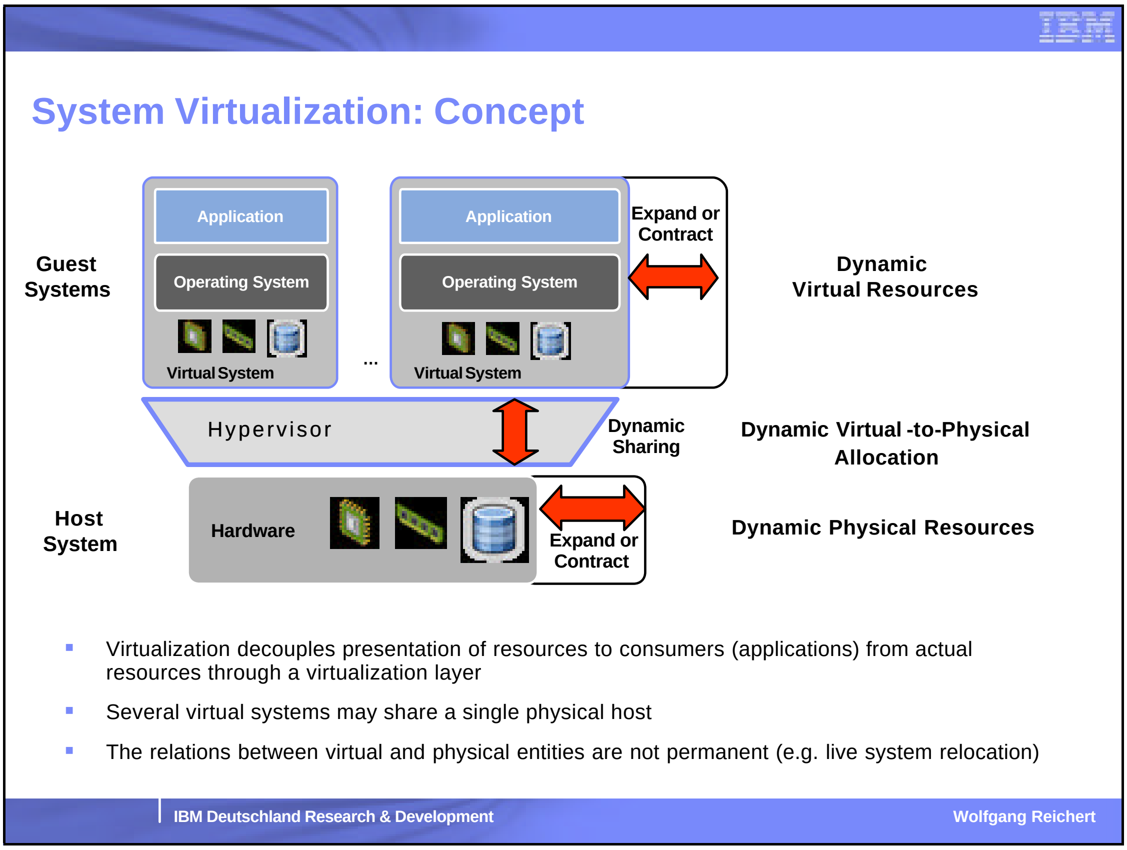
Wolfgang Reichert
Senior Technical Staff Member
IBM Deutschland Research & Development

GI OS/KuVS Workshop Garching, October 2008



Agenda

- Introduction
  - Virtualization concepts, capabilities & use cases
- Integration of application and system management
  - Monitoring
  - Management operations
- Search for Standards
  - DMTF CIM
  - OGF
  - RESERVOIR
- Conclusion



System Virtualization: Concept

Guest Systems

Application

Operating System

Virtual System

...

Application

Operating System

Virtual System

Expand or Contract

Dynamic Virtual Resources

Hypervisor

Dynamic Sharing

Dynamic Virtual -to-Physical Allocation

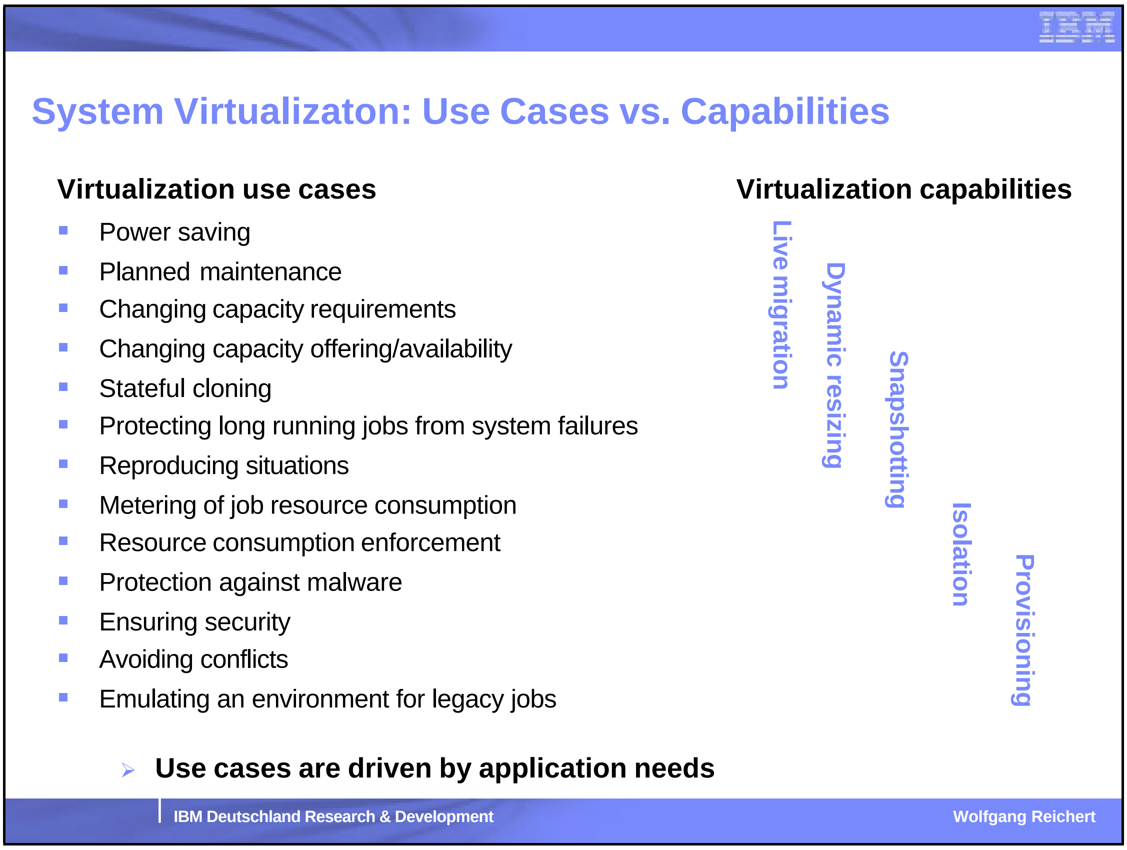
Host System

Hardware

Expand or Contract

Dynamic Physical Resources

- Virtualization decouples presentation of resources to consumers (applications) from actual resources through a virtualization layer
- Several virtual systems may share a single physical host
- The relations between virtual and physical entities are not permanent (e.g. live system relocation)



System Virtualizaton: Use Cases vs. Capabilities

Virtualization use cases

- Power saving
- Planned maintenance
- Changing capacity requirements
- Changing capacity offering/availability
- Stateful cloning
- Protecting long running jobs from system failures
- Reproducing situations
- Metering of job resource consumption
- Resource consumption enforcement
- Protection against malware
- Ensuring security
- Avoiding conflicts
- Emulating an environment for legacy jobs

Virtualization capabilities

Live migration

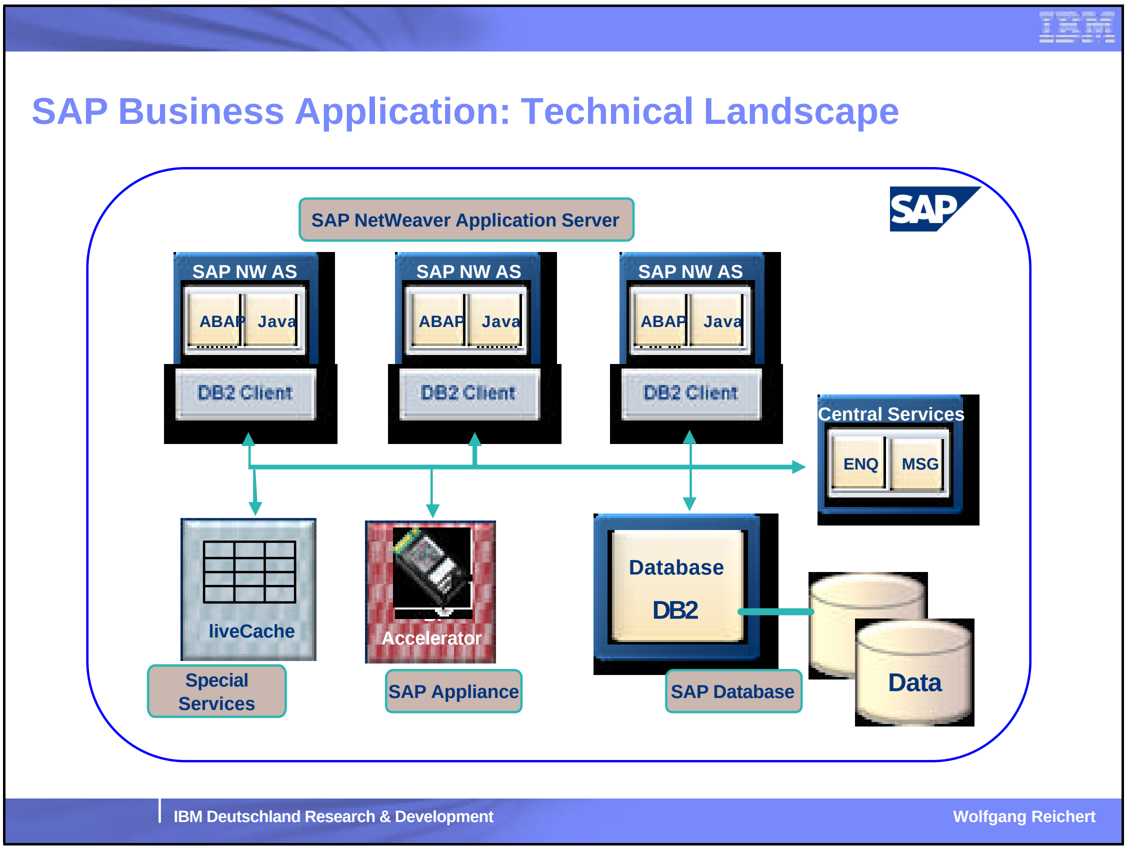
Dynamic resizing

Snapshotting

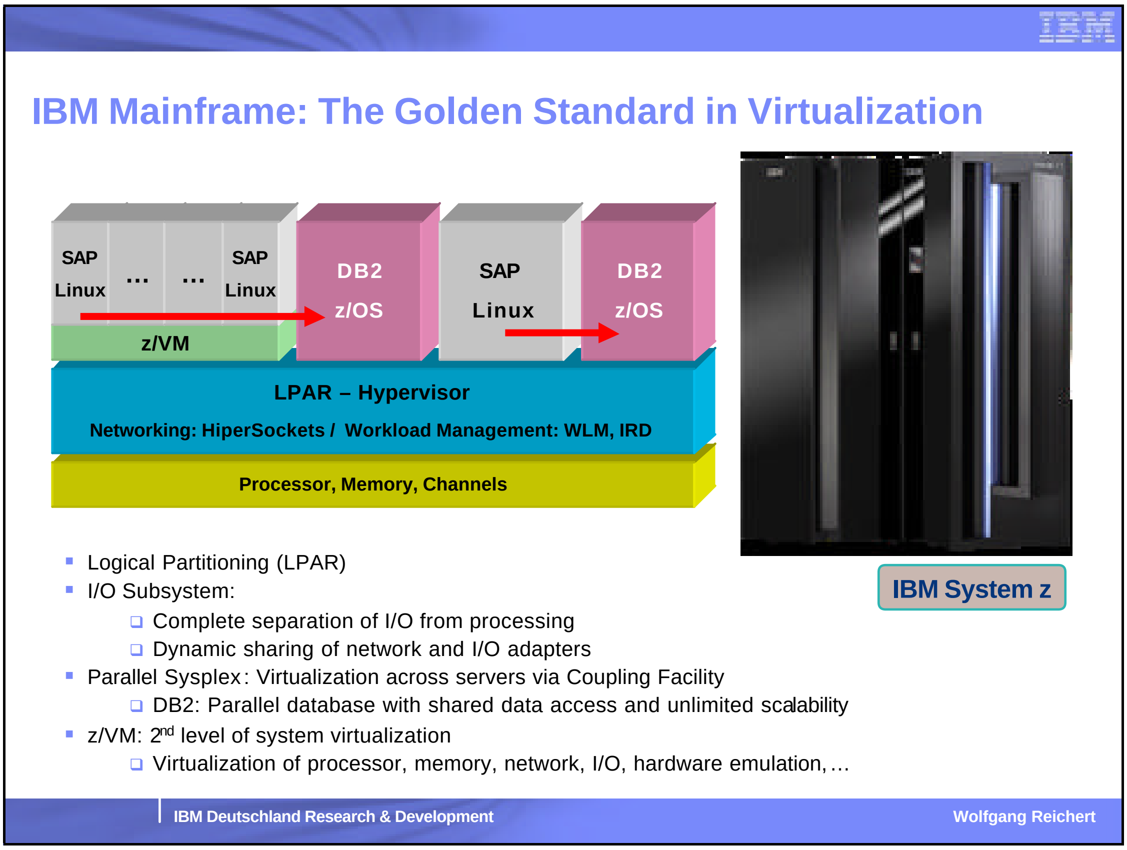
Isolation

Provisioning

➢ **Use cases are driven by application needs**



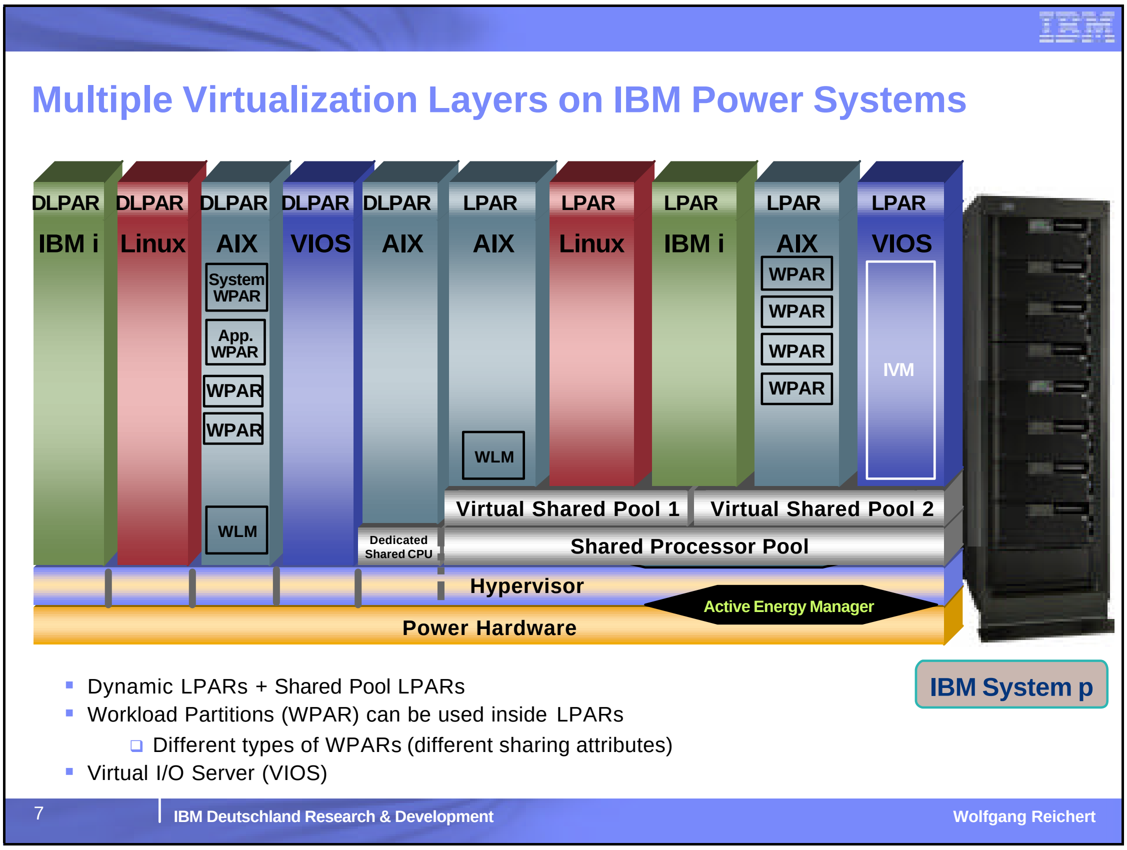
SAP Business Application: Technical Landscape



IBM Mainframe: The Golden Standard in Virtualization

SAP Linux
…
…
SAP Linux
z/VM
DB2 z/OS
SAP Linux
DB2 z/OS
LPAR – Hypervisor
Networking: HiperSockets / Workload Management: WLM, IRD
Processor, Memory, Channels
IBM System z

- Logical Partitioning (LPAR)
- I/O Subsystem:
  - Complete separation of I/O from processing
  - Dynamic sharing of network and I/O adapters
- Parallel Sysplex: Virtualization across servers via Coupling Facility
  - DB2: Parallel database with shared data access and unlimited scalability
- z/VM: 2nd level of system virtualization
  - Virtualization of processor, memory, network, I/O, hardware emulation, …



Multiple Virtualization Layers on IBM Power Systems

DLPAR IBM i

DLPAR Linux

DLPAR AIX: System WPAR, App. WPAR, WPAR, WPAR, WLM

DLPAR VIOS

DLPAR AIX

LPAR AIX: WLM

LPAR Linux

LPAR IBM i

LPAR AIX: WPAR, WPAR, WPAR, WPAR

LPAR VIOS: IVM

Virtual Shared Pool 1

Virtual Shared Pool 2

Dedicated Shared CPU

Shared Processor Pool

Hypervisor

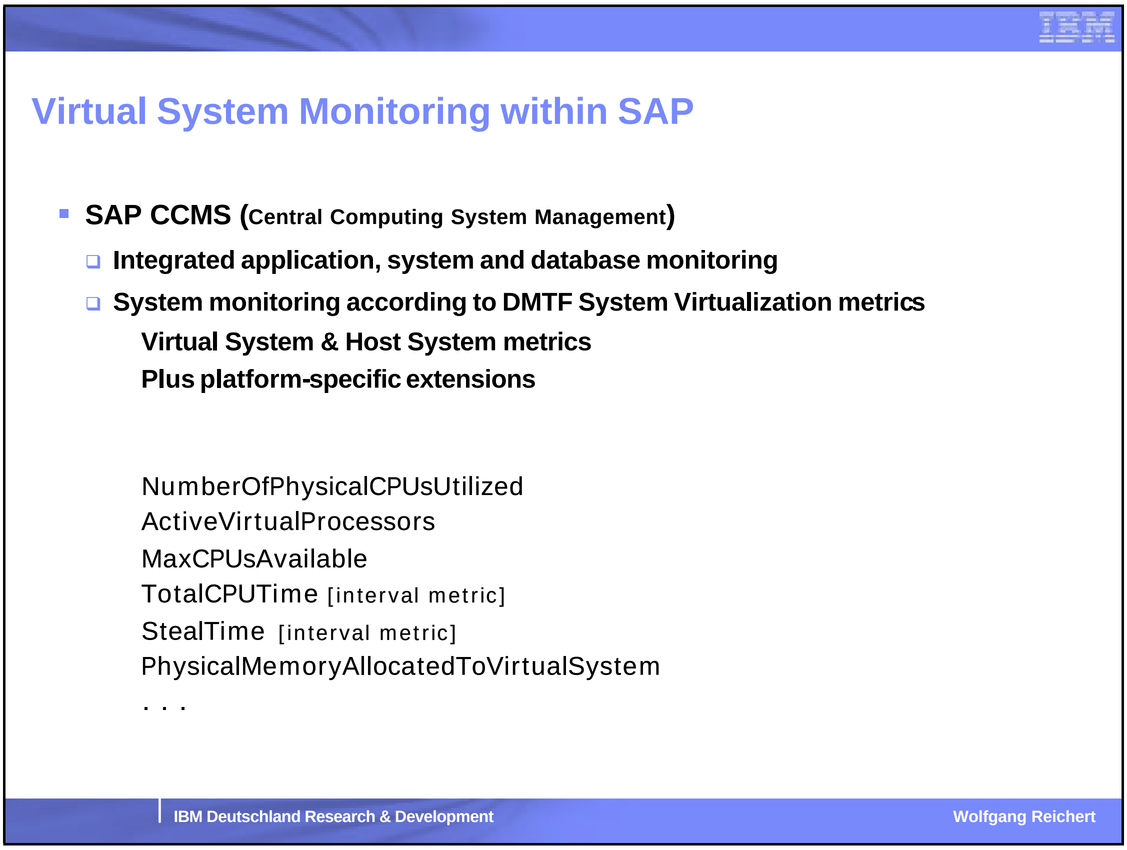
Active Energy Manager

Power Hardware

IBM System p

- Dynamic LPARs + Shared Pool LPARs
- Workload Partitions (WPAR) can be used inside LPARs
  - ❑ Different types of WPARs (different sharing attributes)
- Virtual I/O Server (VIOS)

7 | IBM Deutschland Research & Development Wolfgang Reichert



Virtual System Monitoring within SAP

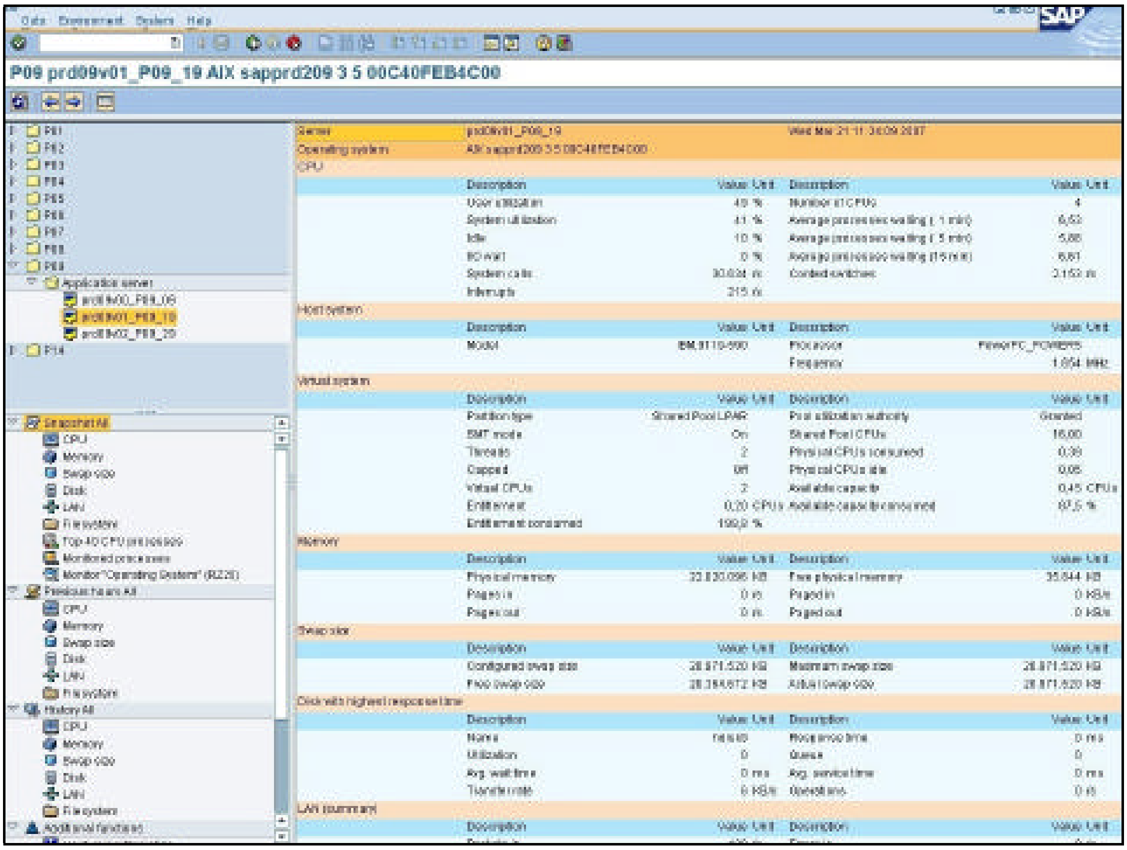
- **SAP CCMS (Central Computing System Management)**
  - ❑ **Integrated application, system and database monitoring**
  - ❑ **System monitoring according to DMTF System Virtualization metrics**

    Virtual System & Host System metrics

    Plus platform-specific extensions

    NumberOfPhysicalCPUsUtilized
    ActiveVirtualProcessors
    MaxCPUsAvailable
    TotalCPUTime [interval metric]
    StealTime [interval metric]
    PhysicalMemoryAllocatedToVirtualSystem
    . . .

| IBM Deutschland Research & Development Wolfgang Reichert



P09 prd09v01_P09_19 AIX sapprd209 3 5 00C40FEB4C00

Server	prd09v01_P09_19	Wed Mar 21 11:36:09 2007
Operating system	AIX sapprd209 3 5 00C40FEB4C00	

CPU

Description	Value	Unit	Description	Value	Unit
User utilization	49	%	Number of CPUs	4	
System utilization	41	%	Average processes waiting (1 min)	6,52	
Idle	10	%	Average processes waiting (5 min)	5,86	
I/O wait	0	%	Average processes waiting (15 min)	6,81	
System calls	30.024	/s	Context switches	3.153	/s
Interrupts	215	/s			

Host system

Description	Value	Unit	Description	Value	Unit
Model	IBM,9119-590		Processor	PowerPC_POWER5	
			Frequency	1.654	MHz

Virtual system

Description	Value	Unit	Description	Value	Unit
Partition type	Shared Pool LPAR		Pool utilization authority	Granted	
SMT mode	On		Shared Pool CPUs	16,00	
Threads	2		Physical CPUs consumed	0,39	
Capped	Off		Physical CPUs idle	0,05	
Virtual CPUs	2		Available capacity	0,45	CPUs
Entitlement	0,20	CPUs	Available capacity consumed	87,5	%
Entitlement consumed	199,2	%			

Memory

Description	Value	Unit	Description	Value	Unit
Physical memory	32.620.096	KB	Free physical memory	35.844	KB
Pages in	0	/s	Paged in	0	KB/s
Pages out	0	/s	Paged out	0	KB/s

Swap size

Description	Value	Unit	Description	Value	Unit
Configured swap size	28.871.520	KB	Maximum swap size	28.871.520	KB
Free swap size	28.384.672	KB	Actual swap size	28.871.520	KB

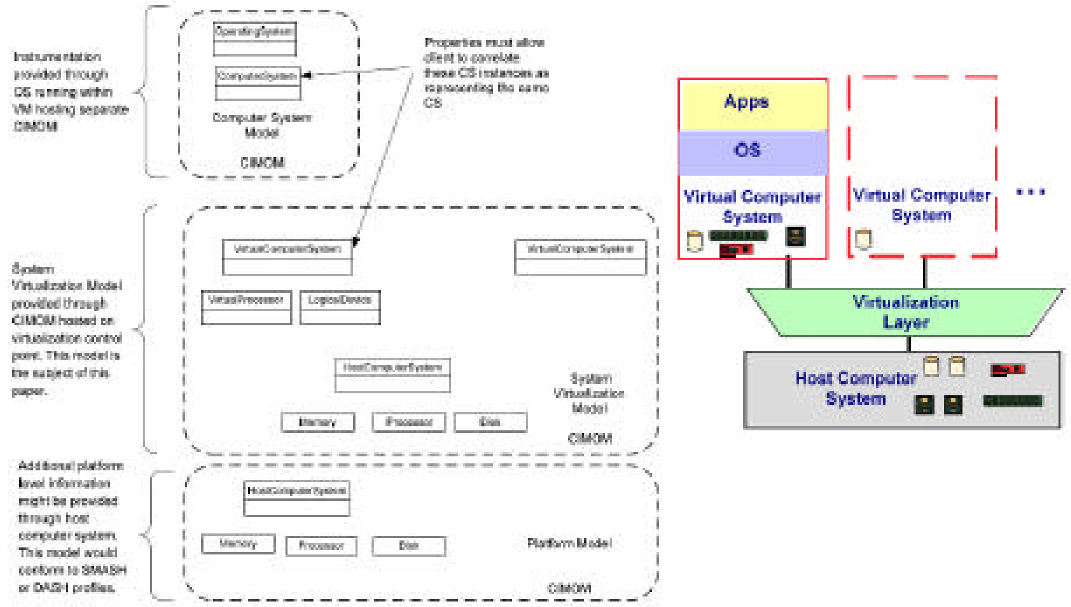
Disk with highest response time

Description	Value	Unit	Description	Value	Unit
Name	hdisk0		Response time	0	ms
Utilization	0		Queue	0	
Avg. wait time	0	ms	Avg. service time	0	ms
Transfer rate	0	KB/s	Operations	0	/s

LAN (summary)

Description	Value	Unit	Description	Value	Unit

CIM System Virtualization Model



Instrumentation provided through OS running within VM hosting separate CIMOM

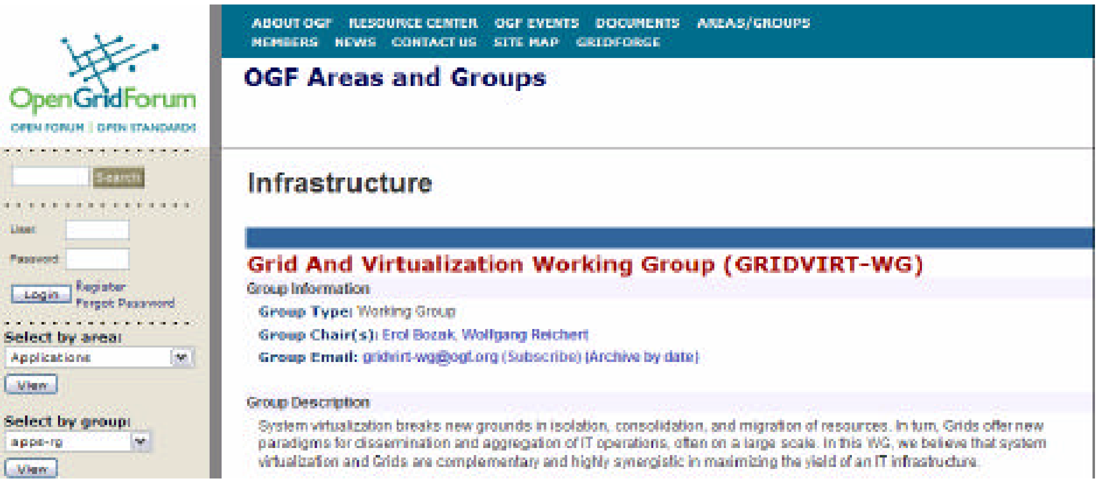
Properties must allow client to correlate these CS instances as representing the same CS

System Virtualization Model provided through CIMOM hosted on virtualization control point. This model is the subject of this paper.

Additional platform level information might be provided through host computer system. This model would conform to SMASH or DASH profiles.

OGF Grid & Virtualization

https://forge.gridforum.org/sf/projects/gridvirt-wg



OGF Areas and Groups

Infrastructure

Grid And Virtualization Working Group (GRIDVIRT-WG)

Group Information

Group Type: Working Group

Group Chair(s): Erol Bozak, Wolfgang Reichert

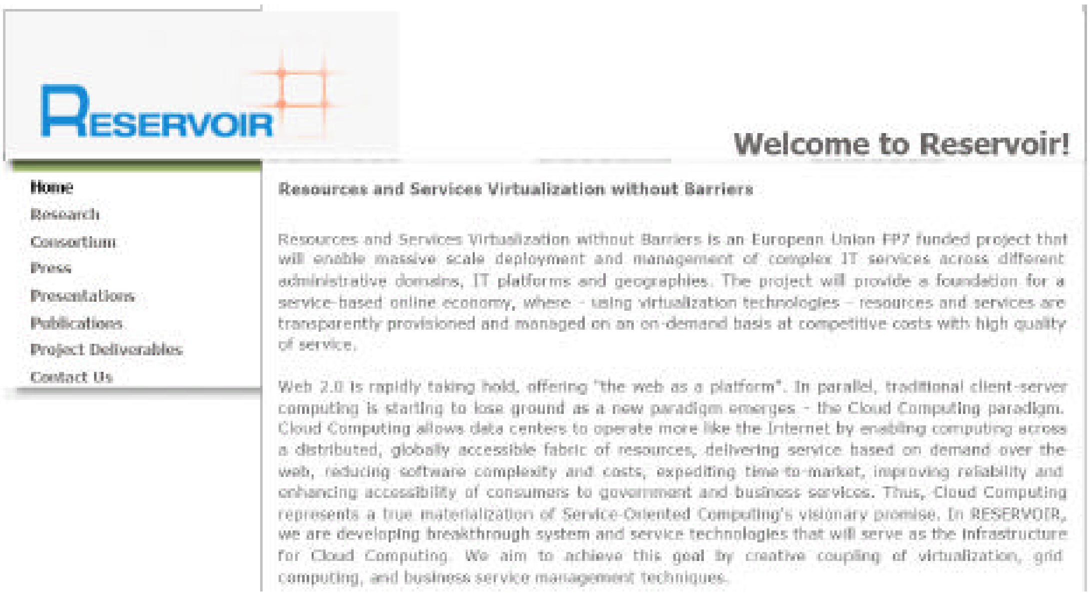
Group Email: gridvirt-wg@ogf.org (Subscribe) (Archive by date)

Group Description

System virtualization breaks new grounds in isolation, consolidation, and migration of resources. In turn, Grids offer new paradigms for dissemination and aggregation of IT operations, often on a large scale. In this WG, we believe that system virtualization and Grids are complementary and highly synergistic in maximizing the yield of an IT infrastructure.

RESERVOIR Project

http://www.reservoir-fp7.eu/

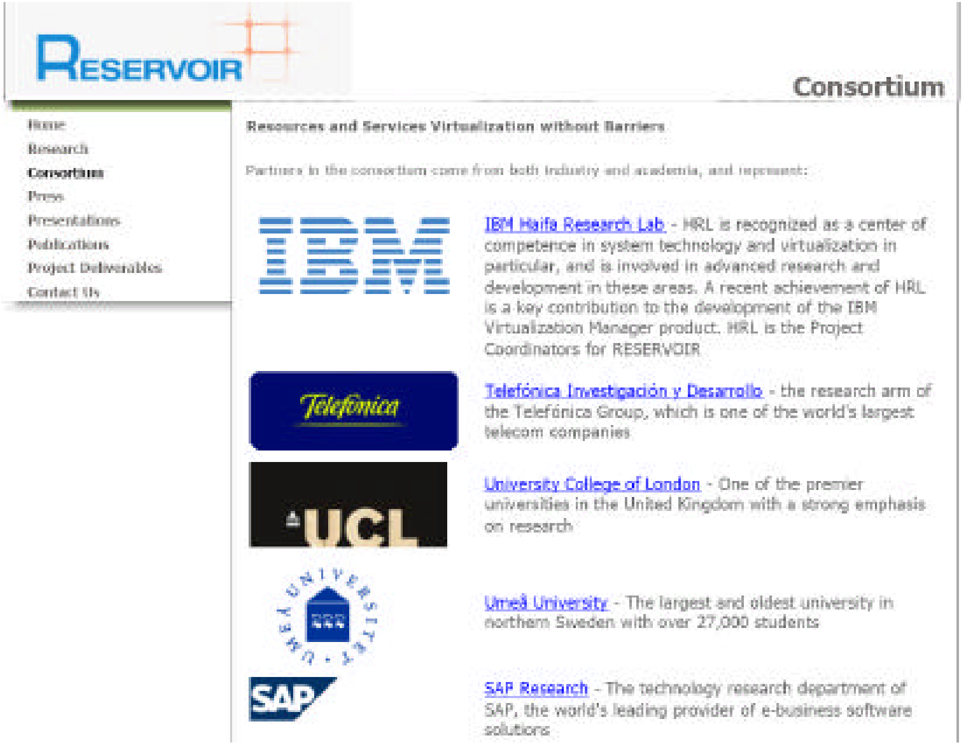


Welcome to Reservoir!

Resources and Services Virtualization without Barriers

Resources and Services Virtualization without Barriers is an European Union FP7 funded project that will enable massive scale deployment and management of complex IT services across different administrative domains, IT platforms and geographies. The project will provide a foundation for a service-based online economy, where - using virtualization technologies - resources and services are transparently provisioned and managed on an on-demand basis at competitive costs with high quality of service.

Web 2.0 is rapidly taking hold, offering "the web as a platform". In parallel, traditional client-server computing is starting to lose ground as a new paradigm emerges - the Cloud Computing paradigm. Cloud Computing allows data centers to operate more like the Internet by enabling computing across a distributed, globally accessible fabric of resources, delivering service based on demand over the web, reducing software complexity and costs, expediting time-to-market, improving reliability and enhancing accessibility of consumers to government and business services. Thus, Cloud Computing represents a true materialization of Service-Oriented Computing's visionary promise. In RESERVOIR, we are developing breakthrough system and service technologies that will serve as the infrastructure for Cloud Computing. We aim to achieve this goal by creative coupling of virtualization, grid computing, and business service management techniques.



Consortium

Resources and Services Virtualization without Barriers

Partners in the consortium come from both industry and academia, and represent:

IBM Haifa Research Lab - HRL is recognized as a center of competence in system technology and virtualization in particular, and is involved in advanced research and development in these areas. A recent achievement of HRL is a key contribution to the development of the IBM Virtualization Manager product. HRL is the Project Coordinators for RESERVOIR

Telefónica Investigación y Desarrollo - the research arm of the Telefónica Group, which is one of the world's largest telecom companies

University College of London - One of the premier universities in the United Kingdom with a strong emphasis on research

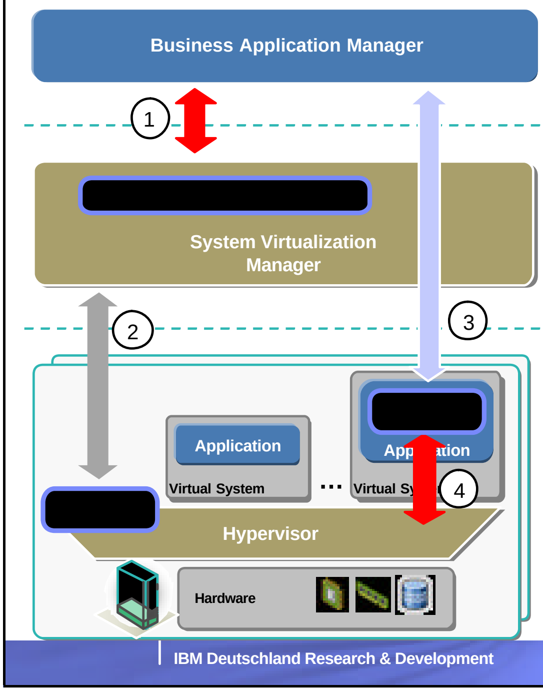
Umeå University - The largest and oldest university in northern Sweden with over 27,000 students

SAP Research - The technology research department of SAP, the world's leading provider of e-business software solutions

RESERVOIR: SAP Use Cases and Requirements

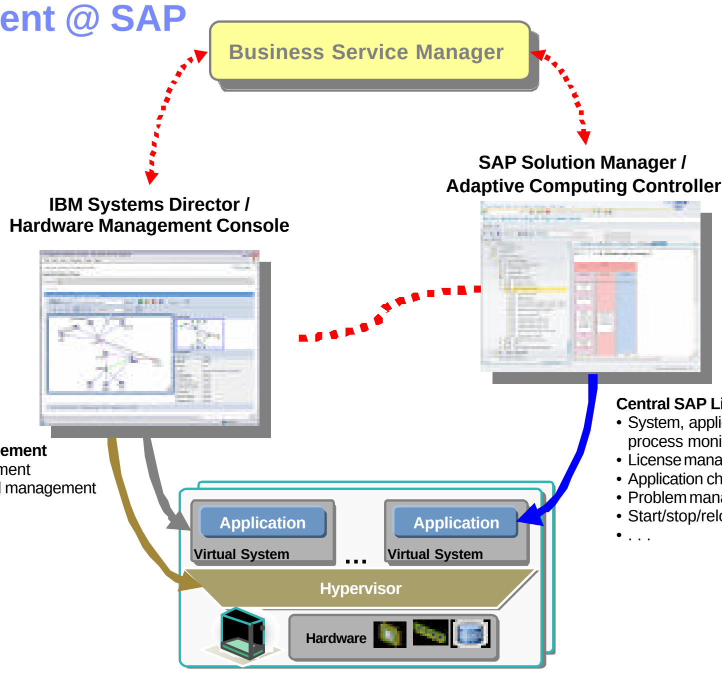
- Creating a manifest of a service application
  - Images, contextualization scripts, DB content, all other configuration data
  - Means to contextualize and customize
  - Considering OVF (Open Virtual Format, DMTF Standard) + Extensions
- Provisioning a service application from a manifest
- Dynamic adjustment of resource allocation
  - Capacity planning
  - Automatic adaptive resource allocation / Self-optimization based on SLA and actual workload
- Elastic array of virtual execution environments
  - Dynamic scale -out by adding virtual servers to a service application
- Live migration
- . . .

Objective: Standardization of Management Interfaces



- Standardization of System Virtualization interfaces (1):
  - ❑ Ease to use for business application managers
- Platform specific System Virtualization Manager manages distributed sets of virtual systems and resources in a cluster (2)
- Application specific management of application components (3)
- Standardization of interfaces between Virtual System ←→ application management agent (4)
  - ❑ Event notification (e.g. failover)
  - ❑ Local dynamic resource allocation

Integration of Virtualization and Business Application Management @ SAP

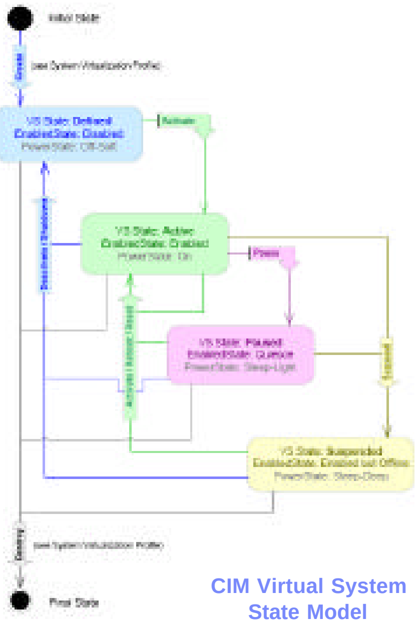


Central System Management
- Virtualization management
- System monitoring and management
- Image management
- System provisioning
- Energy management
- Software distribution
- . . .

Central SAP Lifecycle Management
- System, application and business process monitoring
- License management
- Application change management
- Problem management
- Start/stop/relocation of services
- . . .

Simplified Interfaces Between Application Manager and System Virtualization Manager

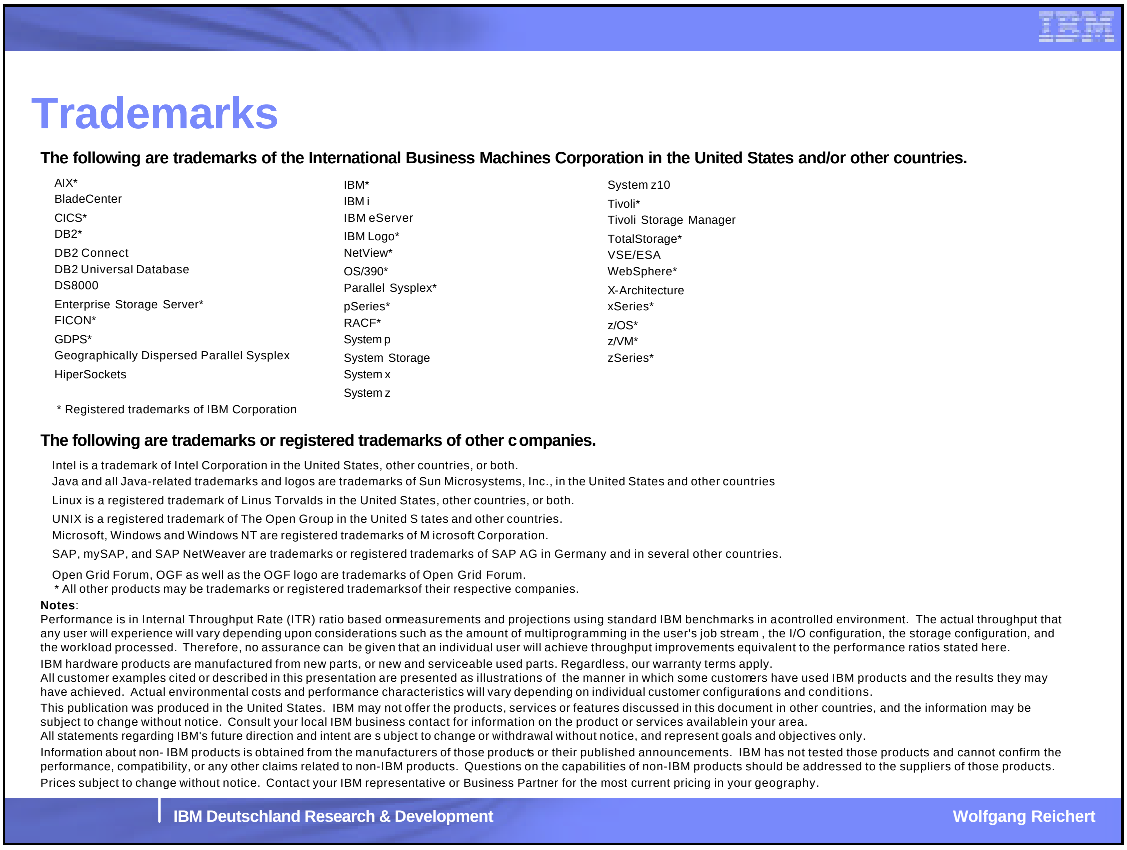
- Topology discovery
  - GetAllHostSystems
  - GetAllVirtualSystems
  - GetAllVirtualSystemsOnHost
  - GetVirtualSystemTopology
- System info
  - GetSystemInfo
  - GetVirtualSystemManagementCapabilities
  - GetSystemMetrics
- System operations
  - Activate, Reboot
  - Deactivate, Shutdown
  - MigrateVirtualSystemToHost
  - . . .



CIM Virtual System State Model

Summary

- *Virtualization is designed to be transparent.*
- *However, when management of complex business application is concerned the application management components must be aware of virtualization.*
- *In a proof-of-concept the author has shown how to integrate SAP application management with system virtualization managers like IBM Power Systems management console.*
- *The integration has been built on top of the newly defined DMTF System Virtualization standard. Most likely it is the first exploitation of this new DMTF standard in the context of commercial applications.*



Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

AIX*
BladeCenter
CICS*
DB2*
DB2 Connect
DB2 Universal Database
DS8000
Enterprise Storage Server*
FICON*
GDPS*
Geographically Dispersed Parallel Sysplex
HiperSockets

IBM*
IBM i
IBM eServer
IBM Logo*
NetView*
OS/390*
Parallel Sysplex*
pSeries*
RACF*
System p
System Storage
System x
System z

System z10
Tivoli*
Tivoli Storage Manager
TotalStorage*
VSE/ESA
WebSphere*
X-Architecture
xSeries*
z/OS*
z/VM*
zSeries*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other c ompanies.

Intel is a trademark of Intel Corporation in the United States, other countries, or both.
Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
UNIX is a registered trademark of The Open Group in the United S tates and other countries.
Microsoft, Windows and Windows NT are registered trademarks of M icrosoft Corporation.
SAP, mySAP, and SAP NetWeaver are trademarks or registered trademarks of SAP AG in Germany and in several other countries.
Open Grid Forum, OGF as well as the OGF logo are trademarks of Open Grid Forum.
* All other products may be trademarks or registered trademarksof their respective companies.

Notes:
Performance is in Internal Throughput Rate (ITR) ratio based onmeasurements and projections using standard IBM benchmarks in acontrolled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream , the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services availablein your area.
All statements regarding IBM's future direction and intent are s ubject to change or withdrawal without notice, and represent goals and objectives only.
Information about non- IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Usage Scenarios II

Virtualization Aspects of Web Application Servers

- Abstract -

Reinhard Hohberger

Over last several years many IT departments - especially in large companies or organizations - built up huge and complex application server environments. Often the number of machines involved reaches into the hundreds. Administration and maintenance of all these servers very quickly becomes a challenging task while on the other hand there is almost no additional value of having so many of them (apart from simply having multiple times the system resources of one single server).

Virtualization techniques offer a way to reduce this complexity, e.g. in the areas of administration, monitoring or runtime behaviour. The aim of virtualization is not only to simplify these tasks but also to introduce completely new functions into the architecture, like the definition of ´'Service Level Agreements'' (SLAs) or some aspects of Autonomic Computing (self-healing, self-optimizing, etc.).

Most of the times the basic technology for these scenarios is the Java Enterprise specification from Sun Microsystems Inc., which defines a standard for Web Application Servers. Other technologies for distributed environments like plain CORBA or DCE are therefore not covered here.

Some aspects of virtualization have been part of J2EE / JEE (Java 2 Enterprise Edition / Java Enterprise Edition) application servers since many years, e.g. the idea of grouping single server instances into clusters that can be managed as one single unit containing all the resources of all servers. Though these clusters can span multiple physical machines and therefore use all their resources, they usually have also a lot of limitations and restrictions like:

- All the servers need to be of the same version, running on the same operating system or must be configured identically. Therefore it is harder to find a large number of servers that can be put together into a cluster.
- The relationship of applications (resource consumers) and servers (resource providers) is very static and not flexible. As a consequence it is not possible to dynamically assign system resources to those applications that currently need them.
- The administration, maintenance and monitoring still is very focused on the individual server and not onto the whole group.

Due to the growth of IT in the industries and also due to company mergers and acquisitions the size and complexity of application server architectures has steadily increased. And as the cost of running these environments has done the same there is a need for more sophisticated virtualization support here.

Apart from administration, maintenance and monitoring costs there is another important aspect of virtualization: If system resources can be shared between applications, it is often possible to provide the same level of performance and availability with less hard- and software (software often is licensed on a cpu basis) than before.

On the other hand virtualization raises many challenges, not only technical ones but also organizational and ''cultural'' ones. System administrators don't like the idea that some kind program takes control over their production systems, distributing system resources automatically to the applications that need them. There is a non zero risk that errors in the virtualization soft- or hardware lead to system outages. (Of course, the same holds true for manual administration or self written scripting frameworks but this is often seen not as critical.)

Using the IBM product ''WebSphere® Virtual Enterprise'' as an example the current status of industry proven solutions is explained and a lookout for possible future developments of web application server environments is given.

About my person:
After studying Mathematics at the University of Bayreuth I joined the Heidelberg Science Center of IBM in 1995. Since 8 years I'm working as a WebSphere Specialist in the IBM Software Group with special focus on application servers.

Walldorf, Oktober 2008

Virtual Supercomputer for HPC and HTC

Georg Birkenheuer, André Brinkmann, Hubert Dömer, Sascha Effert,
Christoph Konersmann, Oliver Niehörster, and Jens Simon

Paderborn Center for Parallel Computing PC^2
University of Paderborn, Germany
{birke, brinkman, homerj, fermat, c_k, nieh, jens}@upb.de

Abstract This paper evaluates the behavior of virtualization environments for high performance computing (HPC) and high throughput computing (HTC). The focus is on performance measurements and analysis of virtualized storage and network traffic. The results show that with modern hardware architectures the loss of performance due to virtualization is low, but that improvements are still possible and necessary. All in all, the use of virtualization environments seems to be promising for both HPC and HTC.

1 Introduction

Virtualization is a well-known approach in modern computer centres, where several operating systems are executed encapsulated on one server. The aim is to share the computing power of the system and therewith to increase the energy efficiency and thus the ecological and economical environmental performance. The reason for the great benefit of virtualization in computer centres is the high capacity of current hardware architectures, which have typical low workloads of about 15 percent for a single application, while their response times are still acceptable for an average workload of 70 percent. Thus, several virtual machines can be executed on a single server.

Users of HPC and HTC applications often suffer from a huge management burden to transfer their applications between different cluster environments. The programs are often released for a special operating system and even on the correct one different library versions can impede the use of HPC and HTC. Based on the provision of virtualized images, this management burden can be significantly reduced.

At least in HTC, it might be also beneficial to use virtualization technology to improve overall throughput. On the one hand the applications have computing intensive problems to calculate and the computing nodes have a very high workload even with a single application. On the other hand, the scientific applications are never using the complete computing element, but are dependent on some parts of the machine while other parts of the machine are underutilized. There might be a very CPU time consuming application, which still has to wait sometimes for many IO cycles or there might be a memory-consuming application that uses little network traffic. There might be even a single threaded application, which only use one of many CPU cores. Thus even with high load, combining two applications on one computing element can be beneficial if they use different properties of the node.

Today, the use of virtualization in HPC and HTC is still very uncommon. Nevertheless, in both scenarios, the usage of virtualization might be very benefical. Thus an analysis about possible bottlenecks in virtualization for these applications seems necessary. Inside this paper, we evaluate the influence of the virtualization solutions VMWare Server, Xen and VirtualBox on HPC and HTP computing scenarios.

The properties we focus are IO performance when reading and writing massive amount of data and the performance using TCP-IP and MPI communication protocols. They are essential for virtualization in HTC, especially when increased throughput by shared resource usage should prevail the loss of performance due to parallel execution and virtualization. We have measured and analyzed their impact on network communication and disk IO performance for different processors. In contrast to standard evaluations, we focus on typical HPC and HTC settings. The results of disk IO have been mostly measured with only a single virtual server running on each hosting physical server. Network communication tests have been done on configurations with up to two physical servers and up to four VMs per server. The results show that with modern hardware architectures the loss of performance within virtualization is low, but still demands for some advancement to make HPC and HTC virtualization practicable.

The paper is organized as follows. In the following Section 2 we firstly discuss related work. In Section 3 we introduce the experimental setup. In Section 4 we evaluate the experiments about network traffic. Section 5 shows the results of the storage IO measurements. A conclusion and outlook on future work finishes this paper.

2 Related Work

Camargos et al. have presented a practical performance study of open source virtualization technologies [1]. They have evaluated the overhead of VMM layers and the scalability of VMMs by benchmarking multiple VM environments. They have used a set of common open source tools covering a wide range of applications. For CPU intensive tasks, most virtualization solutions based on OS-level- and para-virtualization technologies have a performance close to native performance, while the performance of full-virtualization solutions is significantly worse. The full-virtualization solutions have also a poor performance in simple networking experiments. An interesting result is the reached network performance of VirtualBox that is higher than the native one, which is constituted by possibly existing special network drivers (this is in contrast to results presented inside this paper). Furthermore, the paper shows how good the VMMs manage host resources between multiple VMs.

Another analysis is presented by Youseff et al. [2]. They have examined the overhead of the open-source hypervisor XEN on HPC applications with the focus on para-virtualized computation and MPI-based communication. A comparison with native linux systems on their HPC cluster environment has shown that XEN produces no statistical significant overhead in general. The only exception they found is the bidirectional MPI network bandwidth.

Based on performance losses in MPI environments, Huang et al. have developed a VM-aware communication library, which supports near-native network performance in HPC environments and that is based on the MPI-2 implementation MVAPICH2 [3]. Their library uses shared-memory communication among computing processes running on virtual machines deployed on the same physical host system. A similar approach is done in [4]. It takes advantage of IVC that support VM-migration to other physical hosts and enables a transparent use of MPI. It has been shown [4] that the shared memory optimization can improve the performance by 11% and that the library has no significant overhead compared to MVAPICH2. A previous work [5] describes a framework with the goal of reducing the management overhead of VMMs by using VMM-bypassing I/O. This technique removes the VMM from the critical path running on the guest to the device. The paper shows that this enables the VM a near-native I/O performance.

Publications measuring the pure disk IO performance in virtual environments are very rare. Ahmad et. al. analyzed the behavior of the ESX Server in 2003 [6]. The results are mostly comparable to the native host IO performance. For different kinds of accesses the average ratio between native host performance and virtual host performance accessing physical disks differs between 0.95 and 1.08. Higher performance in the virtual machine is explained by better drivers in the ESX Server. In 2007, VMWare published a technical report comparing the performance of virtual and physical disks used from inside the virtual machine [7]. It is shown that in most cases there is no difference between using a virtual disk or a physical disk. The enterprise solutions VMWare ESX and XenEnterprise are also analyzed in [8] and [9] with slightly different conclusions. The paper published by VMWare has shown that the ESX Server is sometimes faster than XenEnterprise, but mostly the results are close together. The paper published by XenSource can not reconstruct the results and gets much closer results. Ahmad describes how to analyze the workload inside a VMWare ESX Server and offers some hints to the kind of IO patterns appearing in different scenarios [10] .

3 Experimental Setup

In this Section we describe the setup of our experiments.We have used two host systems running SLES10 SP1 with a 32 Bit kernel in version 2.6.16.46-0.12. Each machine has a E5405 Intel processor (4 cores, 2 x 6MB cache), 6 GByte RAM and a gigabit Ethernet network interface card. The used virtualization platforms are the open-source environments XEN 3.0.4, VMWare Server 1.0.4 and VirtualBox 1.5.2. XenSource uses the technique of paravirtualization [11], while the others perform a full-virtualization of the host system. Table 1 summarizes our experiment configurations.

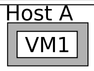
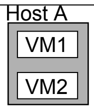
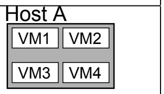
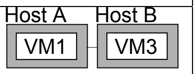
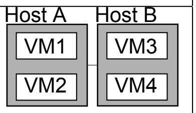
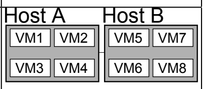
To measure the influence of the server configuration, we have also performed some tests with an elder transtec server with one Intel Xeon 2.4 GHz CPUs with 512 KB Cache and 1GB main memory. The system is equipped with a QLogic QLA2300 64-bit Fiber Channel Adapter.

All machines are attached attached to twelve Seagate ST373207FC Fiber Channel disks inside a transtec JBod.

4 Network performance Measurements

Our evaluation is based on the MPI implementation MPICH2 in version 1.1.1-a1 on guests with one GByte RAM and one virtual CPU. The communication performance depends on the memory and network bandwidth and latency. Memory bandwidth on the four processor core server is about 3 GByte/s and the network performance has 2 x 1 Gbit/s bidirectional bandwidth. One very important property that determines performance

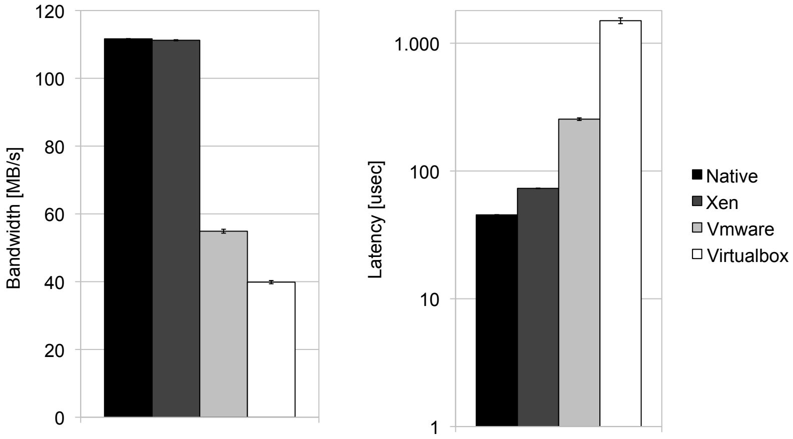
Table 1. Experimental setups

label	description	figure
1x1	one VM running on a physical host	
1x2	2 VMs running on the same physical host	
1x4	4 VMs running on the same physical host	
2x1	2 VMs running on 2 different hosts	
2x2	4 VMs running on 2 different hosts	
2x4	8 VMs running on 2 different hosts	 Host A Host B VM1 VM2 VM5 VM7 VM3 VM4 VM6 VM8

is the network device of the virtual machines. XEN implements a software device without a bandwidth limitation. VMWare Server supports the three different devices pcnet32, e1000 and vmxnet with different performance properties. The guest-systems are configured to communicate over a vmxnet device. We have compared this with e1000 configuration and have found no significant differences. VirtualBox offers four network devices. We use the e1000 device.

4.1 MPI point-to-point communication

In the next two subsections we are evaluating the Intel MPI benchmark suite IMB 3.0. At first we are studying the MPI_PingPong benchmarks of this suite. Bandwidth and latency are analyzed given the configurations of table 1. Figure 1 contains firstly (black) the native performance of the machine, secondly (dark grey) the performance of the para-virtualized Xen Server, thirdly (light grey) the performance of the full-virtualized VMWare and fourthly (white) the performance of the full-virtualized VirtualBox. Every bar shows a confidence interval of $\alpha = 0,05$ and for every value 50 experimental runs have been performed.



(a) Bandwidth

(b) Latency

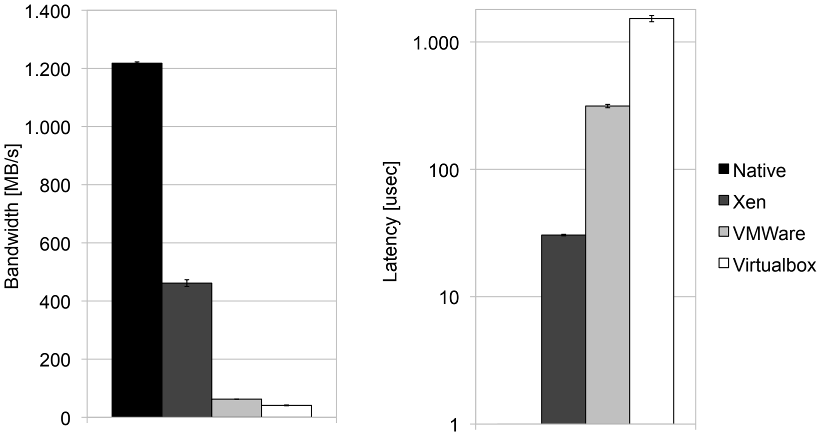
Figure 1. Communication behavior for MPI_PingPong and inter-node communication.

Figure 1(a) shows the bandwidth between two virtual machines on two different hosts, connected with 1 GBit Ethernet. Clearly the best performance in this test is achieved by the native configuration of the gigabit interconnect (110 MByte/s). It is the usable limit for direct applications and thus for every virtual machine too.

The bandwidth of the para-virtualized Xen Server is nearly the same as the native bandwidth, whereas the bandwidth of both full-virtualized solutions is limited to the half (VMWare 55 MByte/s) or less VirtualBox (40 MByte/s).

Figure 1(b) shows the latency between two virtual machines on two different hosts, connected with 1 GBit Ethernet. The latency function shows a more distinct result. Native execution shows the smallest (45 μsek) latency, followed by Xen. Its latency is nearly twice as big as the native one (73 μsek).

The latency of VMWare (250 μsek) is 5 times larger and the latency of VirtualBox (1500 μsek) is 33 times as large as the native one. This means, next to the fact that the fully-virtualized solutions have a poor bandwidth usage, VirtualBox also causes high latencies. The implementation of the network interfaces seems to be on a higher level than the of the other virtualization solutions. Xen and VMWare are using own the network devices (xennet and vmxnet), whereas VirtualBox emulates the e1000 interface.



(a) Bandwidth

(b) Latency

Figure 2. Communication behavior for MPI_PingPong and intra-node communication.

Figure 2 shows the bandwidth and latency behavior between two virtual machines on the same host, which allows the MPI library to use shared memory data and to avoid the network card.

Firstly, Figure 2(a) shows that the native performance win for communication on the same host (1200 MByte/s) over communication on another host (110 MByte/s) as seen in figure 1(a) is immense (factor 11). The para-virtualized Xen implementation is also able to increase the bandwidth using caching mechanisms (460 MByte/s). The advantage from para-virtualizion in this case is that the special booted Xen kernel is aware of the fact that it is running as virtual environment and can also use caching and avoid the network card. Nevertheless, the communication has to go through one kernel and the underlying Xen kernel into the other guest system, therefore the performance increases by a factor 4, which is much less than the native use of MPI.

The fully-virtualized solutions VMWare (60 MByte/s) and VirtualBox (40 MByte/s) show nearly the same performance as in the previous runs (55 MByte/s and 40 MByte/s), as they are not aware of the fact that they are using the same host machine. This is based on the encapsulation of the virtual machines, they cannot use shared memory access. Every communication has to be done over the network card, causing an immense performance loss.

Figure 2(b) shows the latency behavior between two virtual machines on the same host. The latency of the native solution ($0,9\,\mu$sek) shows that a message which uses shared memory (probably cached) can be accessed nearly without loss of time. The win of the native shared memory to the native access over an network card ($45\,\mu$sek) is the factor 50. The Xen solution ($30\,\mu$sek) can also profit from the caching effects (prior $70\,\mu$sek). In contrast the fully-virtualized VMWare can profit little ($310\,\mu$sek to $250\,\mu$sek) and VirtualBox cannot profit from the shared memory ($1500\,\mu$sek).

4.2 MPI group communication

In the last section we focused on point-to-point communication. Now, the network communication performance of configurations with up to four VMs per node are investigated. The results are produces by the MPI_Alltoall benchmark of the IMB suite.

Figure 3 shows the accumulated bandwidth of communicating MPI processes with different configurations. It can be observed that the native communication performance over the network is about the rough bidirectional 1 Gbit/s (190 MByte/s). The intra-node communication bandwidth is limited by the memory bandwidth because communication is realized by copying messages from the process user space of the sender to the process user space of the receiver with buffering. Differences between 1x2 and 1x4 configuration are due to different efficiency in cache usage. Xen intranode performance with two VMs shows the expected 380 MByte/s

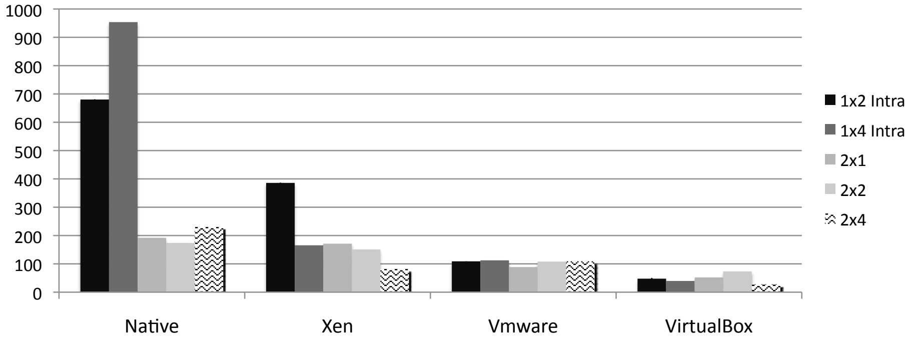


Figure 3. Bandwidth with MPI_All2all with message size of 4 MB (intra node) and 512kB (internode).

(the same value as the MPI_PingPong performance), but in the case of 4 VMs on a server, the performance drops to less than halve of this value (160 MByte/s). The same behavior can be observed, when 4 VMs per node are communicating over the network. The performance decreases from 170 MByte/s down to 80 MByte/s with 4 VMs. The para-virtualization requires a extra process for kernel operations. This process generates a high CPU load, so that the four VMs do not have an exclusive processor. The other virtualization technologies are worse than Xen. VMware is not able to sustain the potential network performance, even if more than one process uses the network connection. The bottleneck is in this case the implementation of the network interface in the VM. VirtualBox has only half the performance of VMware. It can also be observed that the communication bandwidth over the physical network is higher than between VMs sharing the main memory of a physical server. This shows the hard limitations of VirtualBox using local memory. Splitting the copying from user space to communication buffers and back from buffers to user space of the other VM on two physical machines seems to relieve the memory interface of the single physical machine. The communication performance over the network increases with one VM to two VMs on a node. But with 4 VMs in VirtualBox, the total communication bandwidth decreases by the factor of two. VirtualBox is in general not able to efficiently sustain as many VMs as processor cores are available in the system. As seen in the intra-node communication, the memory interface limits the performance of the VMs and also the implementation of the network interface seems not to be optimized for multi-VM usage.

In conclusion the Xen virtualization technology is the most efficient implementation for inter VM message communication. With more hardware support for para-virtualization, this technology can even be more efficient in case of highly loaded (HPC and HTC) system configurations.

5 Storage performance Measurements

Virtual machines have to cope with very different IO workloads [10]. To measure the storage performance of the environment, we have used IOMeter 2006.07.27. In this section, we analyze the performance impact based on the different virtualization approaches and therefore we have performed sequential read and write tests with 64 KB sized blocks as well as the random read performance for 4 KByte sized blocks.

The first approach is to compare the native performance of the complete physical disk without virtualization with the performance of the

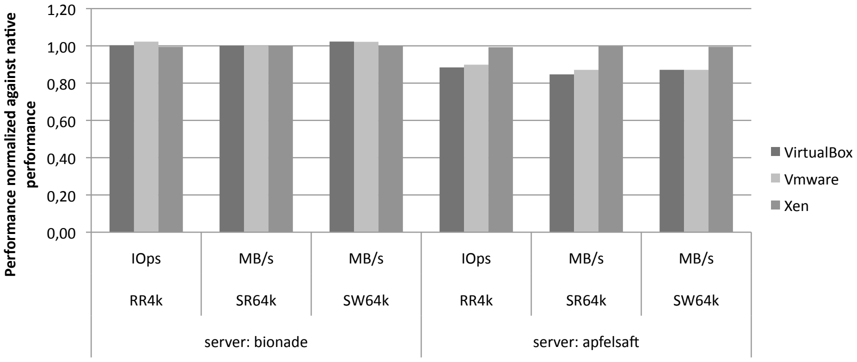


Figure 4. Relative virtual disk performance of Xen, Virtual Box, and VMWare.

complete physical disks, which are used as pass-through device (e.g. /dev/sda under Linux). Figure 4 shows on the left side this comparison for the server bionade, which is the 4 core machine. All measurement results for the different virtualization solutions for bionade are within a few percent of the native performance of this server, indicating that the disk is the limiting factor inside this setup. This holds for random reads (RR4k), sequential reads (SR64k), and sequential writes (SW64k) and is independent on the virtualization technology.

This changes for a smaller server with an elder 2,4 GHz Xeon processor (apfelsaft). In this case, the performance of the para-virtualized Xen environment is still within a few percent of the native performance, but the fully-virtualized VMWare and Virtual Box environments endure an overhead for not being able to directly pass-through the IO commands to the disk. Therefore, the performance of full-virtualized systems seems to be much more processor-power dependent than the performance of the para-virtualized solution Xen.

Both fully-virtualized environments show an interesting effect when importing partitions like /dev/sda1 under Linux as pass-through devices (see Figure 5). The performance of random reads and sequential reads is similar to the performance of a complete device, but the sequential write performance drops to 50% of the native disk performance. This effect is documented by VMWare and can be explained by an improper alignment of the partition, which typical occurs under Linux when no extra care is taken [12]. Under VMWare, reading or writing is performed in so-called *clusters*, which are a multiplicative of a standard sector. These clusters

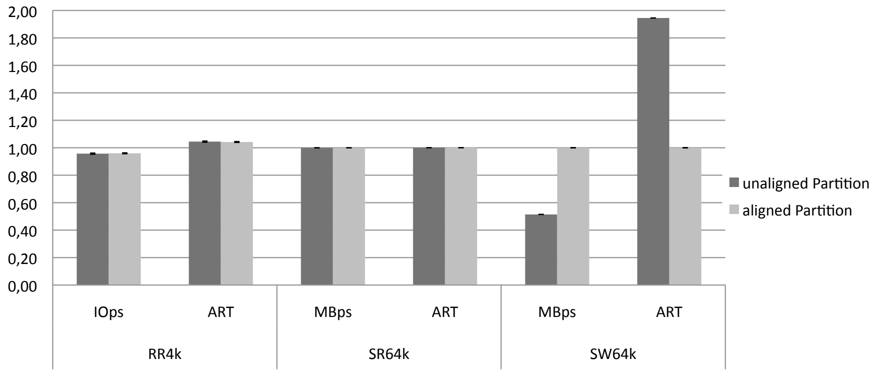


Figure 5. Influence of partition alignment for VMWare.

are mapped on *blocks*, which are again mapped on *chunks*. Therefore, one read or write access to a block can lead to a read or write access of multiple chunks, significantly decreasing the performance of the storage system. Based on the test pattern, this effect has the biggest impact on write performance.

Using a virtual disk placed on a file system makes the administration of a huge number of virtual machines easier, therefore this way is often preferred by administrators. In this case, paravirtualization does not always benefit compared to hosted virtualization solutions. Thus, a

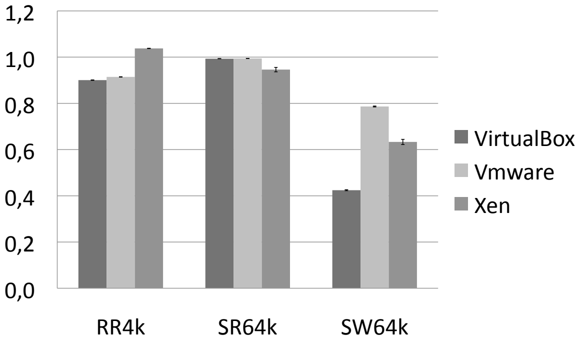


Figure 6. Relative virtual disk performance of Xen, Virtual Box, and VMWare on pre-allocated Ext 2 file system.

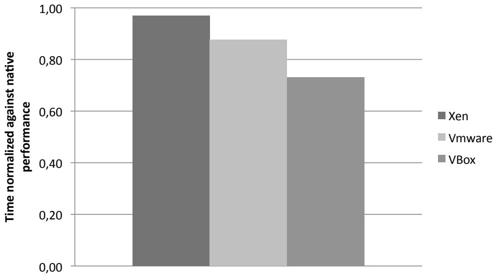


Figure 7. Kernel build time.

closer look on the performance results is required (see Figure 6). For random IOs, the performance of Xen is as good (and even slightly better) as the native implementation and also the sequential read performance is within 6% of the native read performance. Nevertheless, in case of sequential write performance, the performance of Xen drops to little more than 62% of the native performance. This looks different for VMWare: While random reads are within 92% of the native performance and therefore significantly slower than Xen, the performance of sequential reads is as fast as the native performance and the sequential write performance is 80% of the native performance and therefore 30% faster than Xen. This is especially important for HPC and HTC environments, where for many applications most disk IOs are sequential writes to checkpoints.

In this case, VirtualBox perform similar to VMWare for random reads and sequential reads, but significantly lacks behind the performance of VMWare and Xen for sequential writes.

Putting this all together, Figure 7 shows the Linux kernel build time, involving disk accesses and processing time. In this case, we have used a FibreChannel disk as pass-through device, which has been formated with an EXT 2 file system.

6 Conclusion and Future Work

The choice of the virtualization environment has a big impact on the IO performance of the virtual machines and therefore on the IO performance of the virtualized clusters. Especially network and MPI traffic in virtualized environments are not yet able to compete with native performance. In this case, new concepts like multi-root virtualization might help to increase throughput and decrease latencies [13]. Furthermore, it is interesting to follow new software approaches to increase network performance, which are at the moment mostly based on academic research. Nevertheless, VirtualBox announced its new version 2.0.2, which should have a significantly improved network stack. At least the storage performance of the para-virtualized solution Xen is able to be competitive with native storage performance. Nevertheless, this only holds for pass-through devices and not for storing virtual volumes inside a file system.

References

1. F. L. Camargos, G. Girard, and B. des Ligneris. Virtualization of linux servers: a comparative study. In *Proceedings of the 2008 Ottawa Linux Symposium*, pages 63–76, July 2008.
2. L. Youseff, R. Wolski, B. Gorda, and C. Krintz. Evaluating the performance impact of xen on mpi and process execution for hpc systems. In *First International Workshop on Virtualization Technology in Distributed Computing (VTDC 2006)*, 2006.
3. W. Huang, M. Koop, Q. Gao, and D. K. Panda. Virtual machine aware communication libraries for high performance computing. In *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing (SC)*, 2007.
4. F. Diakhate, M. Perache, R. Namyst, and H. Jourdren. Efficient shared-memory message passing for inter-vm communications. In *Proceedings of the 3rd Workshop on Virtualization in High-Performance Cluster and Grid Computing (VHPC)*, August 2008.
5. W. Huang, J. Liu, B. Abali, and D. K. Panda. A case for high performance computing with virtual machines. In *Proceedings of the* 20^{th} *ACM International Conference on Supercomputing (ICS)*, pages 125–134, 2006.
6. I. Ahmad, J. M. Anderson, A. M. Holler, R. Kambo, and V. Makhija. An analysis of disk performance in vmware esx server virtual machines. In *Proceedings of the Sixth Annual Workshop on Workload Characterization*, pages 63–76, Austin, Texas, USA, 2003.
7. Vmware. Performance characteristics of vmfs and rdm. Technical report, vmware inc., 2007.
8. Vmware. A performance comparison of hypervisors. Technical report, vmware inc., 2007.
9. XenSource. A performance comparison of commercial hypervisors. Technical report, XenSource inc., 2007.
10. I. Ahmad. Easy and efficient disk i/o workload characterization in vmware esx server. In *Proceedings of the 2007 IEEE International Symposium on Workload Characterization*, pages 149–158, Boston, MA, USA, 2007.
11. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. *ACM SIGOPS Operating Systems Review*, 37(5):164–177, 2003.
12. Vmware. Recommendations for aligning vmfs partitions. Technical report, vmware inc., 2006.
13. B. Homölle, B. Schräder, and S. Brütt. Multi Root I/O Virtualization (MRIOV). In *Proceedings of the 1. Fachgespräch Virtualisierung*, pages 11 – 18, 2007.

Broadening the Battle-zone

Some Thoughts on the Extension of Grid/Cluster Computing into the Desktop and Pool PC Domain

Michael Janczyk, Dirk von Suchodoletz

Professorship for Communication Systems
Institute of Computer Science, University of Freiburg

Abstract. Virtualization is established very well on the desktop of software engineers and testers or people who would like to run another operating system within their favorite desktop environment and it sells rather well for computer center machinery consolidation. This paper would like to explore the suitability of virtualization concepts for the idea of distributed computing. The idea itself is not really new, but the introduction of virtualization here might solve a set of open problems in this field like data security issues, separation of functionality and management of each domain: The desktop and the cluster node environments. Optimally both could be kept away from each other allowing completely different entities to manage their domain without interfering with the other. Linux operating systems especially in stateless deployment allow a high grade of flexibility and ease of use in large scale setups, which are of interest to cycle stealing endeavors. Thus the research outlined is focused on this host and cluster environment but open to other options too if applicable.

1 Introduction and Basic Idea

The number of networked machinery is still rising. Nearly every workplace in scientific institutions will be computer equipped. Additionally you will find a number of public workplaces and lecture pools in educational institutions. In the last few years the cluster computing gained some significant attention to meet the rising demand for number crunching. All these machines are to be installed, managed and operated. Cluster and grid computers tend to be separate infrastructure duplicating much of the already offered resources and services: They implement their own network infrastructure, need administration and system management, filling up 19" racks and requiring a substantial amount of cooling.

Computer centers are interested in solutions for an effective resource utilization of the installed machinery: Each node requires some efforts in setup, network connection, management and disposal. Thus new approaches for the operation of the machinery are searched for: How these resources could be used more effectively and efforts to be bundled? Is there any option of cooperation of completely different faculties to share their machinery? Which requirements and conditions are to be met? The answer to these questions will influence the consulting services of the computer center, the resource planning and decisions on purchasing of new equipment.

This paper will outline the basic requirements for idle cycle stealing, the several virtualization scenarios, the setup options and the research questions to prove. The first part will deal with different virtualization options available. The second will compare these options under a number of different aspects. At the end several conclusions are given and proposals for further research are discussed. In this paper a Linux operating environment for the number crunching is presumed: Most of the cluster or grid computing environments use this flexible open source operating system. The host environment of the research discussed here is Linux too, offering a broader range of virtualization options. Some of the tools discussed later on are also transferable to Windows host environments, extending the number of possible machines involved.

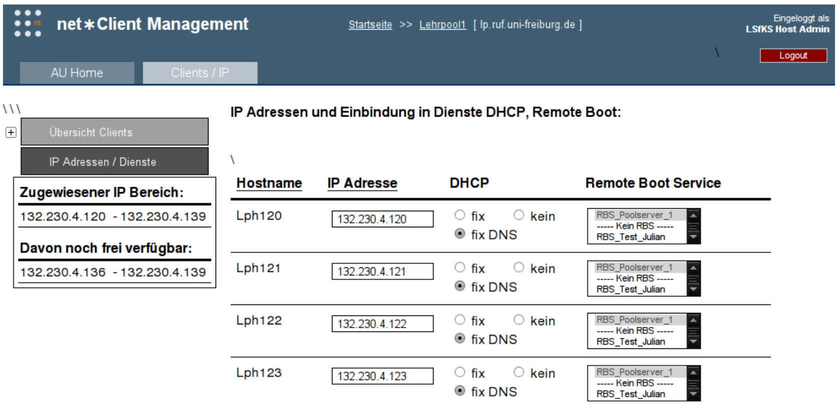
Virtualization for secure cycle stealing Two relevant modes of operation in this field exist: The traditional mode of changing the operation environment after hours into cluster computing. Or: Using virtualization technologies to utilize idle CPU cycles without interfering with the prior ranking desktop operation. The potential of different strategies of virtualization are to be evaluated.

2 Advantages of Network Booting and Central Management

The time of autonomous workstations is over: The majority of machines in larger organizations are centrally managed. The machines are provided with IP configuration via DHCP service. Their setup is done automatically with either software distribution frameworks, several kinds of network installations or completely stateless operation. This delivers the fundamentals for centralized management. Today networks offer huge bandwidths and thus allow fast network setup or network dependent operation. Most of the every day tasks are network dependent. So there is no loss in productivity if the operating systems are provided over the network too.

The most interesting option under the viewpoint of installation and maintenance is stateless operation mode for desktop and cycle stealing environments. This would offer a fast rollout and easy update without the need to install anything relevant on the nodes themselves. The idea of network booting has been around for a while: The first implementations with the Unix workstations or Novell network boot occurred at the beginning of the 1990s. By now Intel's PreBooteXtension (PXE) is well established. Every machine used today is equipped with this boot option. Thus every machine could be configured to ask the network first before starting from some local or optical disk.

Additionally to the partially centralized DHCP and DNS management the computer center of the University of Freiburg started to setup a centrally managed system to configure DHCP and DNS services (Fig. 1) [4]. These services allow operation modes to be specified for the managed machines. In the near



Zugewiesener IP Bereich:
132.230.4.120 - 132.230.4.139

Davon noch frei verfügbar:
132.230.4.136 - 132.230.4.139

IP Adressen und Einbindung in Dienste DHCP, Remote Boot:

Hostname	IP Adresse	DHCP	Remote Boot Service
Lph120	132.230.4.120	○ fix ○ kein ● fix DNS	RBS_Poolserver_1 ----- Kein RBS ----- RBS_Test_Julian
Lph121	132.230.4.121	○ fix ○ kein ● fix DNS	RBS_Poolserver_1 ----- Kein RBS ----- RBS_Test_Julian
Lph122	132.230.4.122	○ fix ○ kein ● fix DNS	RBS_Poolserver_1 ----- Kein RBS ----- RBS_Test_Julian
Lph123	132.230.4.123	○ fix ○ kein ● fix DNS	RBS_Poolserver_1 ----- Kein RBS ----- RBS_Test_Julian

Fig. 1. Computer center offers a web interface to manage the operation of machines configuring DHCP and PXE menus.

future this will allow them to be scheduled on time with a resolution of ten minutes definable for every day of a week.

There are two modes for handling virtual machines from the view point of network management: Assign them a valid IP just out of the same pool the host system is using or run the guest systems behind a NAT masquerading service of the host. The first option has the advantage that there is no difference for the management of the virtual machines compared to traditional cluster nodes: They are directly accessible over the network. But you will need exactly the double amount of addresses compared to the traditional operation without virtualization in such a network. You may need to assign additional ranges of addresses in your DHCP address pools. If NAT is in place extra IP address consumption is not an issue. But getting into the virtual environment could get more complicated. If only a few rather specific ports are in use, the port forwarding done by the host might offer a proper solution. In the case of the cluster node just do all the initialization without outside triggering, no directly accessible address is required.

The Freiburg University computer center has been deploying virtualization on the workstation desktop for quite a while: In 2003 it started a fundamentally new approach to operating lecture pool environments by putting the typical Windows desktop into a virtual machine on top of stateless Linux systems radically simplifying the maintenance of a large number of computers and offering much more flexibility in operation. See the “history” section at the end of the article for reference.

The results of this kind of machine management were very positive, so the concept was promising, thus to be extended on other applications. The challenges for number crunching are concentrated around the different goals: The desktop virtualization focuses on the best performance and users experience. For number crunching the desktop user should not see and feel any of the cluster computing running in the background. Thus tools are needed which are able to run completely without screen output to the local machine and could be set to idle cycle consumption only.

3 The Experiments Matrix

The constraints of using a standard workplace environment differ from the usual grid or cluster setups:

- In the best case the average desktop user shouldn't detect that someone else is using his/her machine too.
- Neither the desktop user should be able to interfere with the cluster operation nor the other way round.
- Both domains should be kept as far as possible from each other: Users logged on into one environment shouldn't be able to read and alter data of other environments.
- The overall overhead of different virtualization methods is to be estimated. The less any deployed environment consumes the more is left for computing.

For a later deployment of the results it is of interest how to overcome the complexity: The different virtualization tools and environments might require special kernels and/or modules or additional tools to be installed on the host machine. Options to separate the setup and operation of the different environments: Optimally the cluster system could be maintained rather independently from the host system and vice versa.

4 Different Virtualization Concepts and Cluster Computing

In this project, virtualization will be used to conveniently separate the desktop and cluster computing environment. Virtualization can be described as an abstraction of resources. This means that a resource like a network adapter can be duplicated through virtualization, or that multiple computers act like one cluster node. A virtual machine on the other hand describes an independent environment which offers virtual resources to the operating system in this environment.

The virtual resources act as their real counterparts. Otherwise the result could differ to the execution on native hardware. These environments are isolated against the host system and other virtual machines. That way the guest should not affect the host system if one neglects the loss in velocity and storage needed to run a virtual machine. Protection of malware when running insecure software is one application, in this solution virtualization offers the possibility of separating the desktop user from the cluster user without adjusting the desktop system to the needs of a cluster. There are different concepts of virtualization around deploying different concepts. For the further analysis the following virtualization techniques were selected to be evaluated for this project:

4.1 Full virtualization with VMware Server

This refers to a technique, which allows you to simultaneously run an unmodified guest operating system on the same hardware as the host system. But traditional X86 architecture is not virtualizable. The first implemented solution to this problem is **binary translation**. The instructions are analyzed and if required emulated on the fly.

Representatives of this type are **VMware Server** and **Virtual Box**. The first one was chosen because of the maturity of the product. VMware being among the pioneers of X86 virtualization. The installation is quite simple and could be easily operated on a stateless client easing large scale rollouts and tests. It does not require any further modification of the system kernel. VMware uses kernel modules which provide the necessary modifications to trap non-virtualizable instructions of the guest operating system.

Further testing involves the creation of guests and exporting them via NFS or NBD to the workstations. First the exports are setup as read-write, but later on it should be preferable to export as read-only. Using the option `independent-nonpersistent` VMware offers an elegant way for local guest file system changes without modifying the common source. VMware itself does not offer any means to control the share of computation power consumed, so this has to be done on the host system.

Installation and handling of the VMware Server is quite straight forward. Install the software on the reference system from which the stateless client root file system is derived from. Depending on the distribution the RPM or TGZ package should be used. After the installation the kernel modules have to be created using the provided perl script `vmware-config.pl`.

```
numvcpus = "2"
memsize = "1024"
MemTrimRate = "0"

ethernet0.present = "TRUE"
ethernet0.allowGuestConnectionControl = "FALSE"
ethernet0.virtualDev = "e1000"
ethernet0.wakeOnPcktRcv = "FALSE"
ethernet0.networkName = "NAT"      # bridged

RemoteDisplay.vnc.enabled = "TRUE"
RemoteDisplay.vnc.port = "5900"
```

4.2 Full virtualization using CPU Features

To overcome the shortcomings of the X86 architecture Intel and AMD introduced virtualization extensions to their CPUs. It is called **hardware-assisted virtualization**. Besides the existing kernel and user mode, these extensions implement a root and non-root mode.

Kernel-based Virtual Machine (KVM) and **Xen 3** are two applications which use the virtualization extensions. KVM is part of the Linux kernel since version 2.6.20. It implements only an interface for the virtualization extensions, which can be used by other programs. The user space part for this type of virtualization is a modified **QEMU**. Guest systems can be created with the command line interface or with external programs. The complexity of getting KVM-QEMU to work in a stateless environment is about the same as the complexity of VMware Server which makes it a candidate for further analysis. But QEMU lacks CPU limitation and at the moment sound usability and stability.

Installation of the kernel modules and the user space program can be done very easily. Many new Linux distributions offer KVM packages in their repositories. But there still exists the possibility to compile the kernel and the user space part separately when using the sources.[1] The handling of QEMU is very simple, since QEMU does not work with configuration files. The whole configuration has to be submitted using the command line. One helpful application called `virt-manager` brings quite a similar GUI to VMware Workstation, where configuration and creation of virtual machines can be done.

```
kvm -M pc -m 1024 -smp 2 -monitor pty \
    -drive file=hda.img,if=ide,boot=on \
    -net nic,macaddr=00:16:3e:2e:c3:18,vlan=0,model=virtio \
    -net tap,script=qemu-ifup -usb -vnc 127.0.0.1:0
```

4.3 Paravirtualization

Another approach to the X86 architecture without virtualization support is paravirtualization. This technique requires a modification of the kernel of the host operating system and the virtualized kernel. The hypervisor is executed in kernel mode.The drawback of this technique is that only open source kernels can be “paravirtualized”.

Xen is the best known paravirtualization for Linux at the moment and thus object in the evaluation field. Guest systems can be created via external programs or `debootstrap`. For the special environment of a stateless client a few changes have to be made, including early bridge configuration in case of using a network bridge. To finally boot Xen, PXELinux with multiboot capabilities is needed.

Xen with bridge configuration seems to be the most complex solution. In stateless mode the bridge has to be configured very early, so that the network connection isn’t interrupted later on which would freeze the client. The fact that the system kernel is loaded in ring 1 makes this solution even more complex. The load distribution options between the desktop hypervisor system and the cluster node guest are to be explored in depth.

[1] KVM site: `http://kvm.qumranet.com`

Installation of the Xen hypervisor can be done quite easily, since many Linux systems still have Xen packages in their repositories. But there exist sources which can be compiled as well.

```
builder = 'linux'
disk = [ 'file:debian-4.0/sda,sda1,w' ]
vcpus = 2
memory = 1024
vif = [ 'mac=00:50:56:0D:B1:26' ]
kernel = '/boot/vmlinuz-xen'
ramdisk = '/boot/initrd-xen'
root = '/dev/sda1'
vncviewer=0
```

As mentioned above, Xen 3 can use the virtualization extensions and therefore can be used for full virtualization. To enable Xen to boot unmodified kernels, it is necessary to adapt the configuration.

```
builder='hvm'
disk = [ 'file:/tmp/install,ioemu:sda,w' ]
device_model = '/usr/' + arch_libdir + '/xen/bin/qemu-dm'
vif = [ 'type=ioemu, bridge=xenbr0' ]
kernel = '/usr/lib/xen/boot/hvmloader'
```

4.4 Operating system-level virtualization

This technique is also called **Jails**. Which describes the character of this virtualization. For each operating system a container is created. Each container corresponds to a system partition and contains the user space part. The kernel is modified to operate well with all user space instances.

Applications using OS-level virtualization are **Linux-VServer**, **Virtuozzo** or its open source part **OpenVZ**. For the first tests Linux-VServer is chosen. Later other products could be tested for advanced capabilities in limiting CPU and Memory. For the installation first a vanilla kernel is patched with VServer extensions. These extensions and their functionality can be controlled via `util-vserver` tools.

Installation of Linux-VServer is a bit more complex. Usually the kernel patch has to be fetched and applied. After the kernel and modules are compiled, the user space tools have to be compiled separately.

```
./interfaces/lback:
127.0.0.1

./interfaces/00/ip:
192.168.1.1

./context:
101
```

5 Preliminary Results

For the first run of tests the Java Linpack benchmark was used. It does not need to reflect the actual demands a certain cluster computing process might impose. But it could deliver some first insights into the difference of direct and virtualized computing. In this setup the user desktop part of the machine is in idle mode not consuming any significant resources.

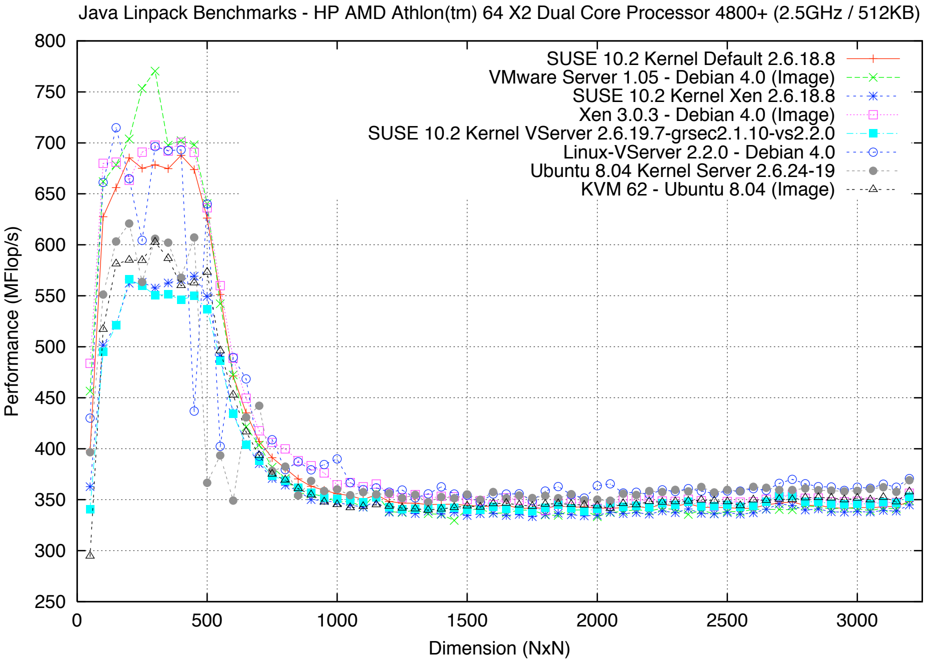


Fig. 2. Java Linpack run on the host system and in the virtual machines

Benchmarks starting with the name of the operating system characterize a run on the host with the described kernel. Those starting with a virtualization technique denote a run of Java Linpack in this particular virtual machine.

There are two parts in figure 2 which are of interest for interpretation: The curve on the left side displays computation handled mainly in CPU cache thus yielding much better results. The middle and right part definite need the machines memory, so do not produce any significant differences between virtualized and direct operation.

In the next step the users desktop was put under load: Simply running the `glxgears` application testing the 3D performance of the system, which is rather synthetic and not a perfect load scheme but needs no user interaction. To simulate disk I/O, the command '`find /`' was started in a loop. The measurements were taken with no load in the virtual environment and while running the Java

Linpack test suite. Thus the influence between the two systems could be measured and compared to a certain degree.

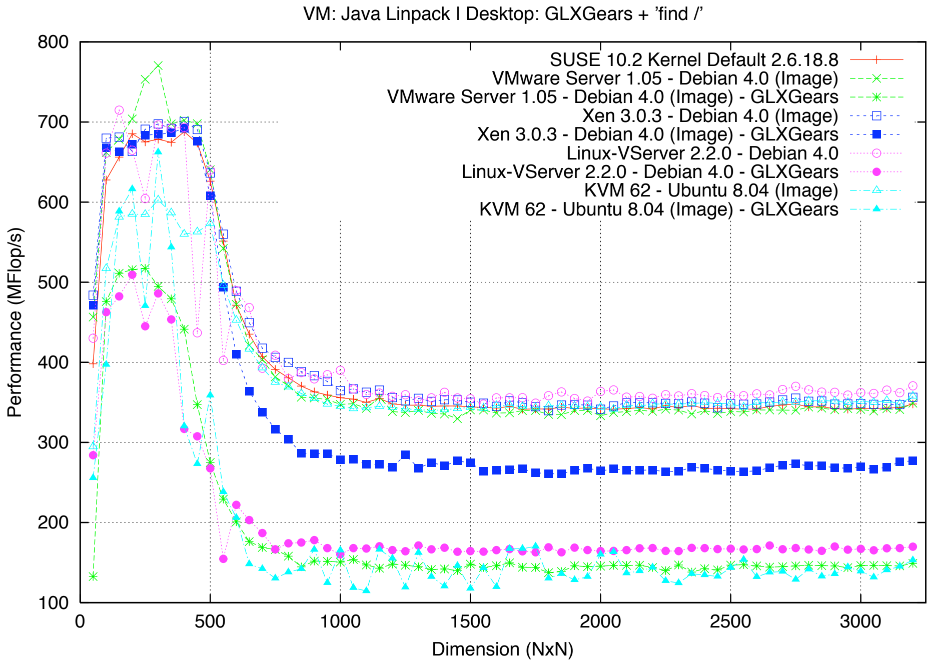


Fig. 3. Java Linpack with the desktop main system put under load by glxgears and find

The Benchmarks marked with 'GLXGears' denotes the run with `glxgears` and `find` running in the desktop environment (Fig. 3). The second graph (Fig. 4) shows frames per second of the program `glxgears` while Java Linpack running in the virtual machine. The results of KVM aren't comparable to the others, since a different OS and X.Org version was used. For further testing, the parameters should be regularized, so that all host systems and all guest systems are equal.

The Linux-VServer seems the best solution to hide the running computation processes in the background. Close behind follows VMware Server 1.05. Perhaps the new version 2 will bring other results. Both show no changes in frames per second during Java Linpack running, which suggests that working in the desktop environment should perform without bigger drawbacks. KVM has to be tested in a normalized environment. But it does not seem to have any advantages to its predecessors in this field. With this configuration Xen uses many more resources from the desktop. The desktop is no longer usable. This means that it must be controlled to ensure it uses fewer resources.

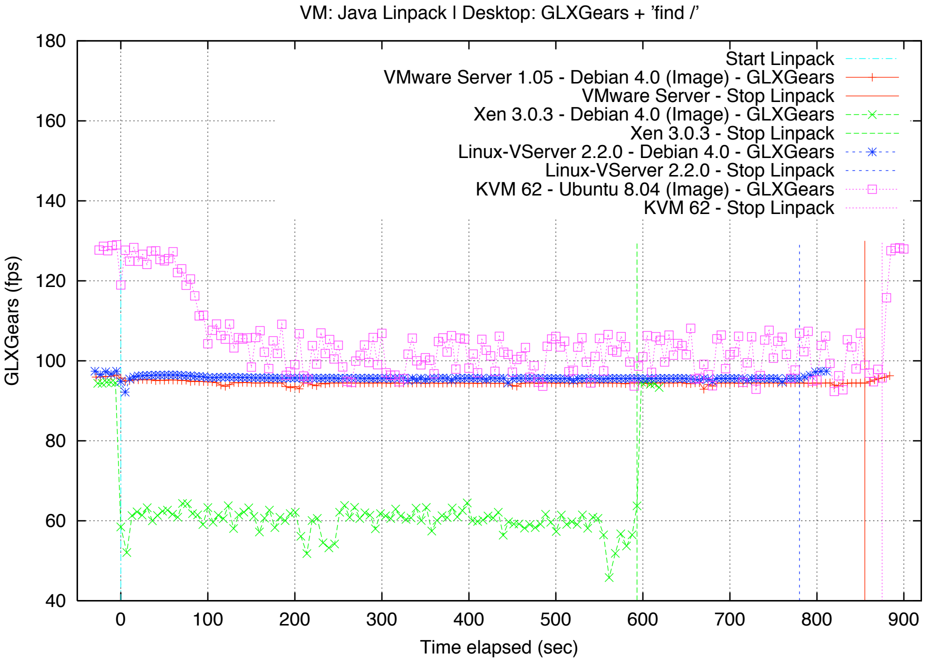


Fig. 4. FPS of glxgears during Java Linpack run in the VM

A general problem exists in the time source for the measurements. At least in full virtualization the evaluation might be disputable. Some real life tests should be run and timed to get more resilient results.

6 Conclusion and Perspectives

This paper discussed how the several virtualization concepts could be brought to an average (stateless Linux) desktop workstation. This delivers the prerequisites for the evaluation of virtualization in cycle stealing scenarios which will follow in ongoing work.

Full virtualization has the advantage of a dedicated kernel running for each the host and guest systems. The overhead is a little bit greater to run two distinct machines, but it doesn't seem to be of much significance. The Linux-VServer approach uses just one Linux kernel for both environments. This is not a problem as long as both environments are happy with the features/ABI the kernel provides.

There are a number of cluster related issues to be discussed in more detail. The stability of the tools involved might be less proven than of a standard setup without virtualization. Nevertheless the user might restart his/her machine any moment. Thus the period of uninterrupted operation might be shorter than on a cluster worker node. Could checkpoint/resume strategies improve the overall results: Recover from interrupts faster or move whole virtual computing environments around from one machine to another? The economic impact could be attempted to be estimated: Which option is better – deploying more dedicated cluster nodes needs to take into account not only the direct costs of the machines, but also the whole infrastructure of power supply and air conditioning, network sockets and rack space.

It would be of interest to compare the virtualization mode to direct operation of the same machines: Are better results feasible if the machine is rebooted into cluster mode after hours? While CPU performance seems only marginally reduced in virtualization mode the impact of amounts of memory allocable is of interest.

7 History: Mastering Windows through Virtualization

Virtualization on the workstation desktop is in use for quite a while at the computer center of the University of Freiburg: In 2003 the first version of a Linux stateless machine operating VMware version 3.2 and later 4.0 was installed, radically changing the mode of operation of lecture pool environments.

Traditionally a number of machines is running its own copy of a Windows operating system with the required applications for the several lectures installed to it. This concept has a number of drawbacks. The large number of applications needed for the several courses led to incompatibilities, exclusions preventing the use of only one installation. The problem was "solved" through exchangeable hard disks duplicating or tripling the number of Windows installations. If a course was prepared some administrator had to prepare up to 20 disks with the typical problems and never all setups are synchronized. Thus the concept of local Windows installations was dropped. A single installation of Windows was made into a VMware Workstation. This image was stored on a NFS server exporting that image to all machines in the lecture rooms (the number is only limited by machine performance and the number of available licenses) in a read-only fashion. Every machine mounts the NFS directory and the local VMware machine starts the virtual guest. To solve the problem of in-operation changes the non-persistent mode is deployed. These changes are stored locally on every machine only during the current session. Thus several issues are solved:

- Every user gets exactly the same "copy" of the image and will see exactly the desktop which was centrally prepared and installed.
- When a system is restarted all local changes are lost: So no changes of the lecture systems are permanent. No malicious software or some disputable changes are persistent.
- It is possible to delegate the setup of applications and installation of requested software to the lecturer.
- Several copies of different lecture environments could be stored in parallel.
- Virtualization in combination with stateless Linux allows much more flexible use of machines than with locally installed Windows copies.

This system could lead to more efficient license management: You might need fewer licenses if all stateless Linux machines with license requiring software are never powered on at the same time. Virtual machine images could easily be distributed and started on a quite diverse range of machines over the campus.

To handle a number of different lecture environments properly (they are not restricted to Windows of a certain version only, but open to a wide range of guest systems supported by VMware Player) some scripting changes to the Linux graphical user interface were made and a VMware configuration is automatically generated. The virtual machine sessions are seamlessly integrated into the graphical login procedure. The user is presented with a list of Linux GUI environments and a list with enabled virtual machine sessions.

Depending on the users selection in the case of any virtual machine chosen a script prepares the session: Creating directories for the temporary configuration file and a directory storing the non-persistent files of VMware Player sessions. The session directly starts into the virtual machine running full screen. Thus the users experience does not differ much from a directly run session. A little tool within the virtual environment does some additional "personalizations" like mounting the home directory and configuring some printers.

References

1. Keith Adams and Ole Agesen. A comparison of software and hardware techniques for x86 virtualization. *SIGARCH Comput. Archit. News*, 34(5):2–13, 2006.
2. Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 164–177, New York, NY, USA, 2003. ACM.
3. Simon Crosby and David Brown. The virtualization reality. *Queue*, 4(10):34–41, 2007.
4. Tarik Gasmi. Ldap site management. Diplomarbeit, University of Freiburg, 11 2006.
5. R. P. Goldberg. Architecture of virtual machines. In *Proceedings of the workshop on virtual computer systems*, pages 74–112, New York, NY, USA, 1973. ACM.
6. Gerald J. Popek and Robert P. Goldberg. Formal requirements for virtualizable third generation architectures. *Commun. ACM*, 17(7):412–421, 1974.
7. Fabian Thorns, editor. *Das Virtualisierungs-Buch.* C&L Computer & Literaturverlag, Böblingen, Deutschland, 2007.
8. Geoffroy Vallee, Thomas Naughton, and Stephen L. Scott. System management software for virtual environments. In *CF '07: Proceedings of the 4th international conference on Computing frontiers*, pages 153–160, New York, NY, USA, 2007. ACM.

Management Concepts

How to Deal with Lock Holder Preemption

[Extended Abstract]

Thomas Friebel and Sebastian Biemueller

Advanced Micro Devices
Operating System Research Center
http://amd64.org/research

1 Introduction

Spinlocks are a synchronization primitive widely used in current operating system kernels. With spinlocks a thread waiting to acquire a lock waits actively monitoring the lock. With sleeping locks in contrast a waiting thread blocks, yielding the CPU to other threads. While sleeping locks seem to provide better functionality and overall system performance, there are cases in which spinlocks are the better alternative.

First, under some circumstances, like in interrupt handler top halves, blocking is not feasible. Second, saving and restoring a thread's state, as sleeping locks do when yielding the CPU, costs time. If the lock-protected critical section is very short, waiting for the lock to be released offers better performance. In both cases, spinlocks provide advantages over sleeping locks. But spinlocks are used for very short critical sections only to avoid wasting CPU time waiting actively.

Spinlocks are built on the assumption that a lock-holding thread is not preempted. In a virtualized environment this assumption is no longer true. Virtual machine monitors (VMMs) schedule virtual CPUs (VCPUs) on physical CPUs for time slices to achieve pseudo-parallel execution. At the end of a time slice the current VCPU is preempted, the VCPU state is saved and the next VCPU starts executing.

If a VCPU is preempted inside the guest kernel while holding a spinlock, this lock remains acquired until the VCPU is executed again. This problem is called lock-holder preemption, identified and analyzed by Uhlig et al. [3] for a paravirtualized version of Linux 2.4 running on top of an L4 microkernel.

This work investigates the influence of lock-holder preemption in the Xen hypervisor, a commodity virtualization system. We show that lock-holder preemption can have a severe performance impact in today's systems. Furthermore, we describe two approaches to counteract the performance degradation, give some details on our implementation of one of the approaches, and show that we are able to fully prevent any performance degradation caused by lock-holder preemption.

2 Spinlocks and Virtualization

Lock-holder preemption describes the situation when a VCPU is preempted inside the guest kernel while holding a spinlock. As this lock remains acquired during the preemption, any other VCPUs of the same guest trying to acquire this lock will have to wait until the VCPU is executed again and releases the lock. Lock-holder preemption is possible if two or more VCPUs run on a single CPU concurrently. Furthermore, the more VCPUs of a guest run in parallel the more VCPUs have to wait if trying to acquire a preempted lock. And as spinlocks imply active waiting the CPU time of waiting VCPUs is simply wasted.

Traditionally virtualization systems do not handle spinlocks in a special way. But as multi- and many-core machines are becoming more and more common, the impact of lock-holder preemption grows. Table 1 shows execution times and spinlock wait times for kernbench — a Linux kernel compilation benchmark — running under Xen 3.1 on a 4-socket 16-core machine.

In the single-guest setup a single 16-VCPU guest is running on the host system and executing kernbench. Here, lock-holder preemption is very unlikely as each VCPU can run on a distinct CPU and thus no preemption is necessary. The two-guests setup introduces a second 16-VCPU guest running a CPU-bound job without any I/O. We simply used 16 processes executing an endless loop. This results in an overcommited system, provoking lock-holder preemption. Table 1 shows an 8.8-second increase in time spent waiting for a spinlock. The kernbench execution time increases by 7.6 seconds, or 7.0 %.

Setup	Guest time [s]		Time spent spinning [s]	
Single guest	109.0		0.2	(0.2 %)
Two guests	117.3	(+7.6)	9.0	(7.6 %)

Table 1. Performance numbers for kernbench i) as a single VM, and ii) in an overcommited system running the kernbench VM and a CPU-bound VM concurrently to cause lock-holder preemption.

To analyze the behavior of Linux' spinlocks in more detail, we instrumented the spinlock code to collect histogram information of spinlock wait times. Figure 1 shows the distribution of number of waits over their time spent waiting. Most of the waits (97.8 %) do not take longer than 2^{16} CPU cycles. A second small fraction of waits, taking between 2^{24} and 2^{26} cyles, occurs only in the two-guests setup. These newly introduced waits match Xen's time slice length of 30 ms and show lock-holder preemption: The VCPUs of the CPU-bound guest always run for complete time slices as they do not block for I/O. Therefore, a lock-holder preempted by a CPU-bound VM will keep the lock for at least a complete time slice. Any other VCPUs trying to acquire that lock will busy wait for at least a part of their time slice – until the lock-holder is rescheduled and releases the lock.

Figure 2 plots the time spent waiting rather than the number of waits. This reveals that almost all of the time spent waiting is caused by the small number of waits caused by lock-holder preemption.

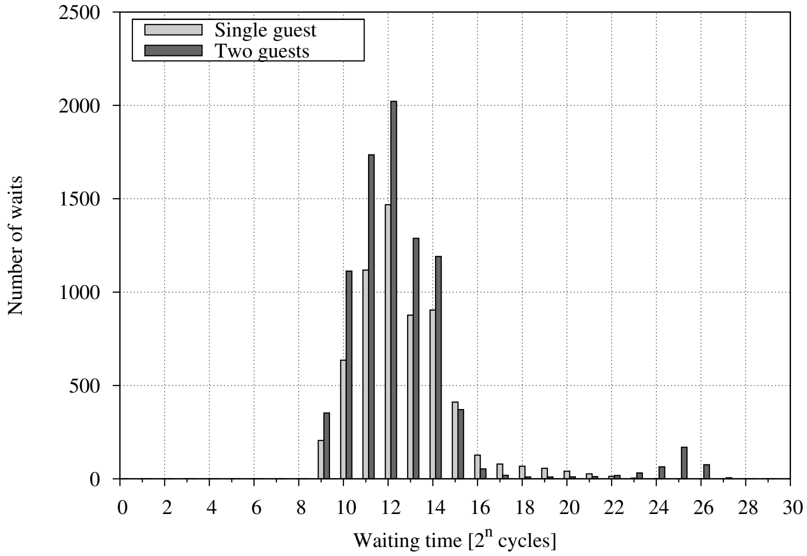


Fig. 1. Histogram of time spent waiting for a spin lock – number of waits for each histogram period. The spinlock waits are aggregated by waiting time into bins of exponentially growing size, e.g. bin 10 shows the number of waits that took between 2^9 and 2^{10} CPU cycles.

3 Tolerating Lock-Holder Preemption

We found two approaches to avoid the overhead caused by lock-holder preemption. First, preventing lock-holder preemption entirely by instrumenting the guest operating system as discussed by Uhlig et al. [3]. Their work leverages three specifics of spinlocks: 1) spinlocks are only used inside the kernel, 2) inside the kernel almost always one or more spinlocks are held, and 3) spinlocks are released before leaving the kernel. This allows to delay the preemption of a VCPU found to run in kernel space until it returns to user space, thus effectively preventing preempting a lock-holder.

The second approach, the approach we follow in this work, tolerates lock-holder preemption but prevents unnecessary active waiting. To achieve this, we need to detect unusually long waits, and switch to a VCPU that is likely to not suffer from lock-holder preemption. Ideally, we would switch to the preempted

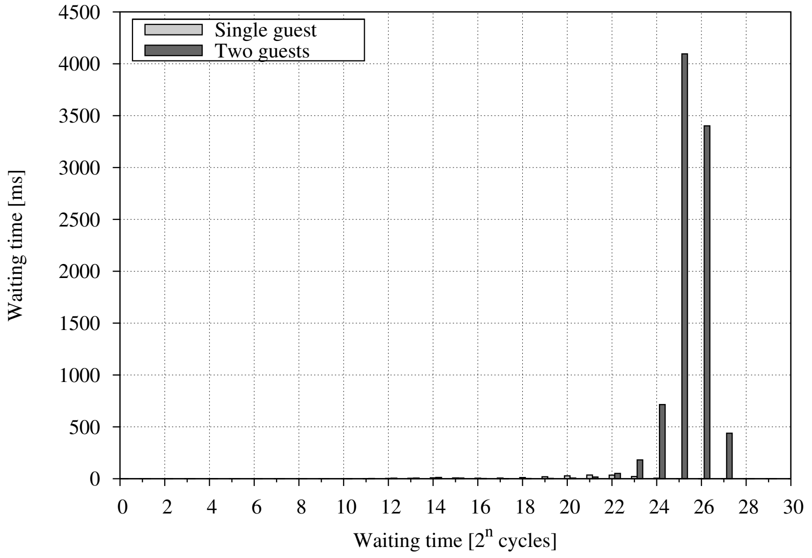


Fig. 2. Spinlock wait time histogram – accumulated duration of waits for each histogram period. The small number of very long waits around 2^{25} account for almost all of the time spent waiting.

lock-holder to help it finish its critical section and release the lock. This is similar to locking with helping as described by Hohmuth and Haertig in [2].

To inform the virtual machine monitor of unusually long waits, we extended the spinlock backoff code to issue a hypercall when waiting longer than a certain threshold. Motivated by the results of Figure 1 we chose a threshold of 2^{16} cycles. After this time almost all native spin-lock waits are finished as the results of the single-guest setup show. On reception of the hypercall, the VMM schedules another VCPU of the same guest, preferring VCPUs preempted in kernel mode because they are likely to be preempted lock-holders. The performance results after these modifications are presented in Table 2. Virtually no time is spent busy waiting in spinlocks anymore. The CPU time spent for kernbench guest CPUs decreased by 7.6 % compared to the unmodified two-guest setup and even by 0.6 % compared to the single-guest setup.

Wall-clock time decreased by 3.9 %, which is only about half of the 7.6 % guest time decrease. This is expected because our setups use shadow paging and kernbench induces a lot of shadow paging work into the VMM by creating a high number of processes. The VMM needs about as much time to handle the shadow paging requests as kernbench needs to complete the kernel compilation. As our modifications only affect the kernbench performance and not the hypervisor we achieve only about half of the guest performance improvement for the complete system. Switching to nested paging would probably yield additional performance.

Setup	Wall-clock [s]	Guest time [s]	Time spent spinning [s]
Two guests	34.8	117.3 (+7.6)	9.0 (7.6 %)
Two guests, helping	33.5	108.4 (-0.6)	0.0 (0.0 %)
Helping improvement	3.9 %	7.6 %	9.0 (7.6 %)

Table 2. Kernbench performance numbers for lock-holder preemption and our helping approach.

4 FIFO Ticket Spinlocks

In early 2008, Piggin [1] introduced FIFO ticket spinlocks to the Linux kernel. Ticket spinlocks try to improve fairness in multi-processor systems by granting locks to threads in the order of their requests.

This intentionally constrains the number of threads able to acquire a contended lock to one – the next thread in FIFO order. In case of contention, a released lock cannot be acquired by any other thread when the next thread in FIFO order is preempted. This effect, called ticket-holder preemption, heavily impairs performance.

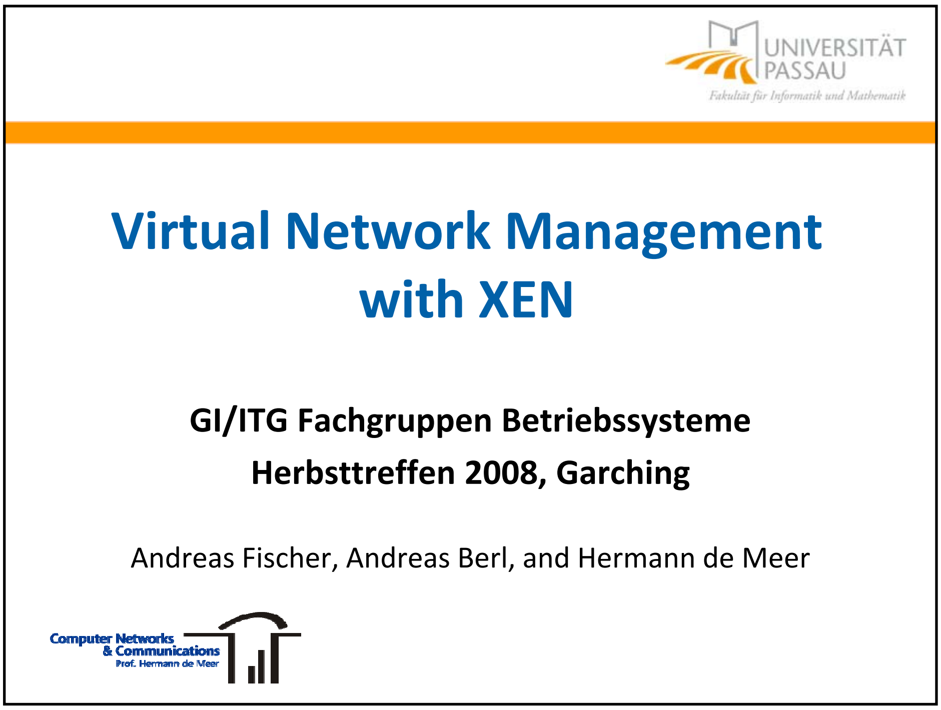
Table 3 shows the performance impact of ticket-holder preemption for our kernbench setup. The observed execution time drastically increases from 33 seconds to 47 minutes. The kernbench guest spends 99.3 % of its time actively waiting to acquire a preempted spinlock. Using our lock-holder preemption aware scheduling, the execution time decreases to 34.1 seconds.

Setup	Wall-clock [s]	Guest time [s]	Time spent spinning [s]
Two guests	2825.1	22434.2	22270.4 (99.3 %)
Two guests, helping	34.1	123.6	6.6 (5.4 %)

Table 3. Lock-holder preemption with ticket spin locks: Kernbench performance numbers for lock-holder preemption and our helping approach.

References

1. J. Corbet. Ticket spinlocks. *LWN.net*, 2008.
2. M. Hohmuth and H. Härtig. Pragmatic nonblocking synchronization for real-time systems. In *Proceedings of the General Track: 2002 USENIX Annual Technical Conference*, pages 217–230, Berkeley, CA, USA, 2001. USENIX Association.
3. V. Uhlig, J. LeVasseur, E. Skoglund, and U. Dannowski. Towards scalable multiprocessor virtual machines. In *VM'04: Proceedings of the 3rd conference on Virtual Machine Research And Technology Symposium*, pages 4–4, Berkeley, CA, USA, 2004. USENIX Association.

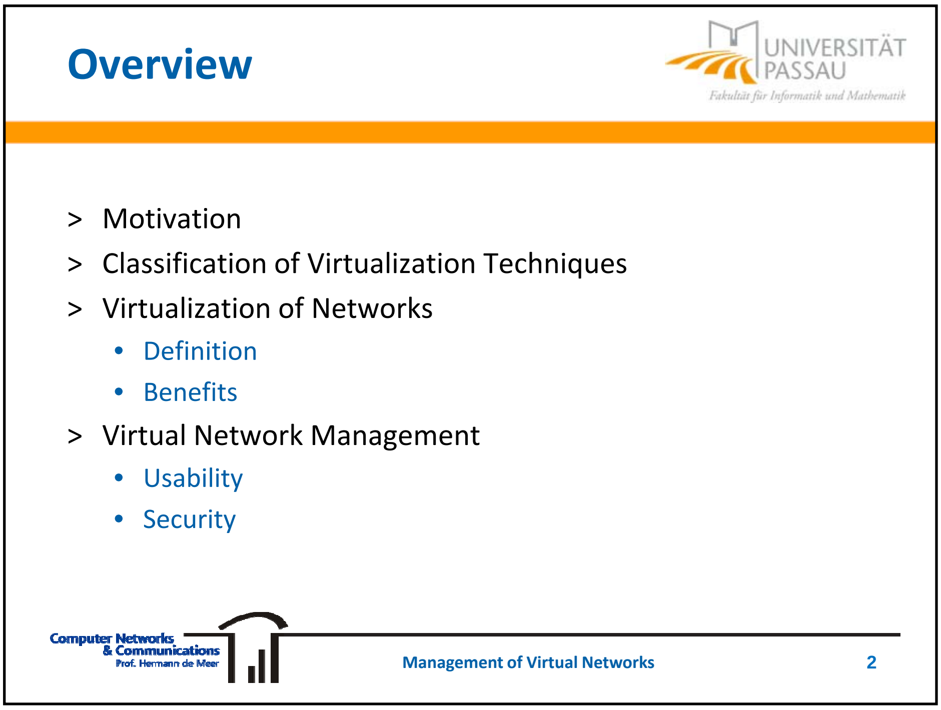


Virtual Network Management with XEN

GI/ITG Fachgruppen Betriebssysteme

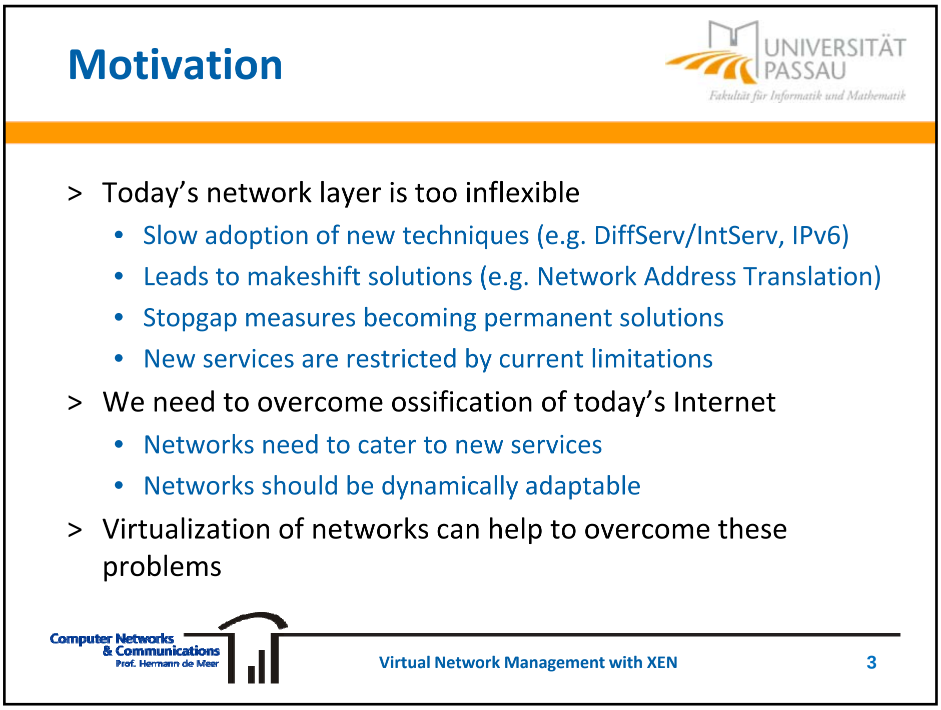
Herbsttreffen 2008, Garching

Andreas Fischer, Andreas Berl, and Hermann de Meer



Overview

> Motivation
> Classification of Virtualization Techniques
> Virtualization of Networks
  - Definition
  - Benefits
> Virtual Network Management
  - Usability
  - Security

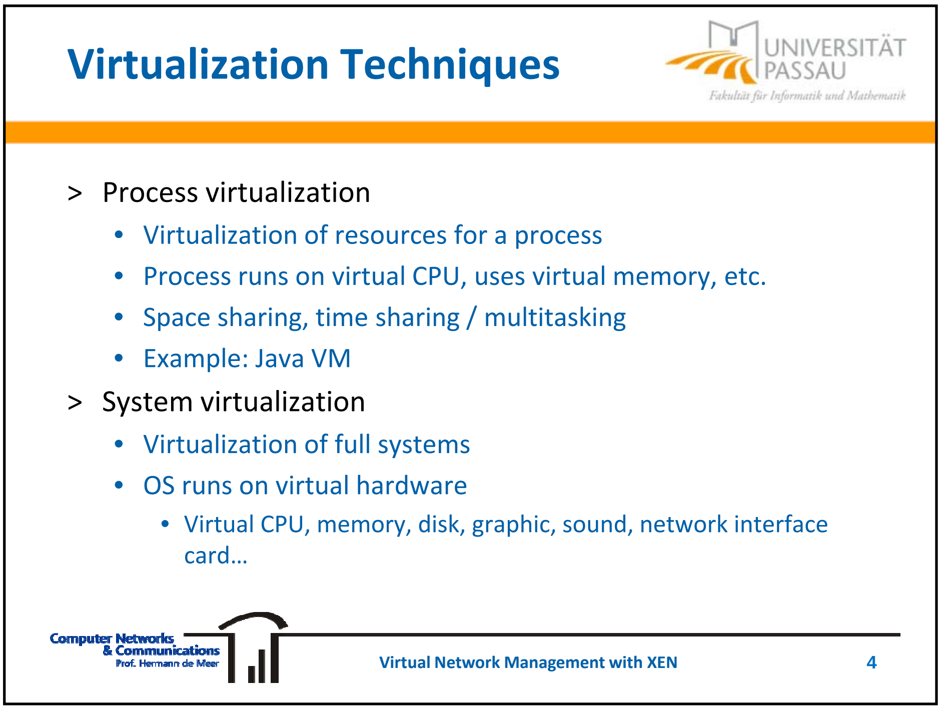


Motivation

> Today's network layer is too inflexible
  - Slow adoption of new techniques (e.g. DiffServ/IntServ, IPv6)
  - Leads to makeshift solutions (e.g. Network Address Translation)
  - Stopgap measures becoming permanent solutions
  - New services are restricted by current limitations

> We need to overcome ossification of today's Internet
  - Networks need to cater to new services
  - Networks should be dynamically adaptable

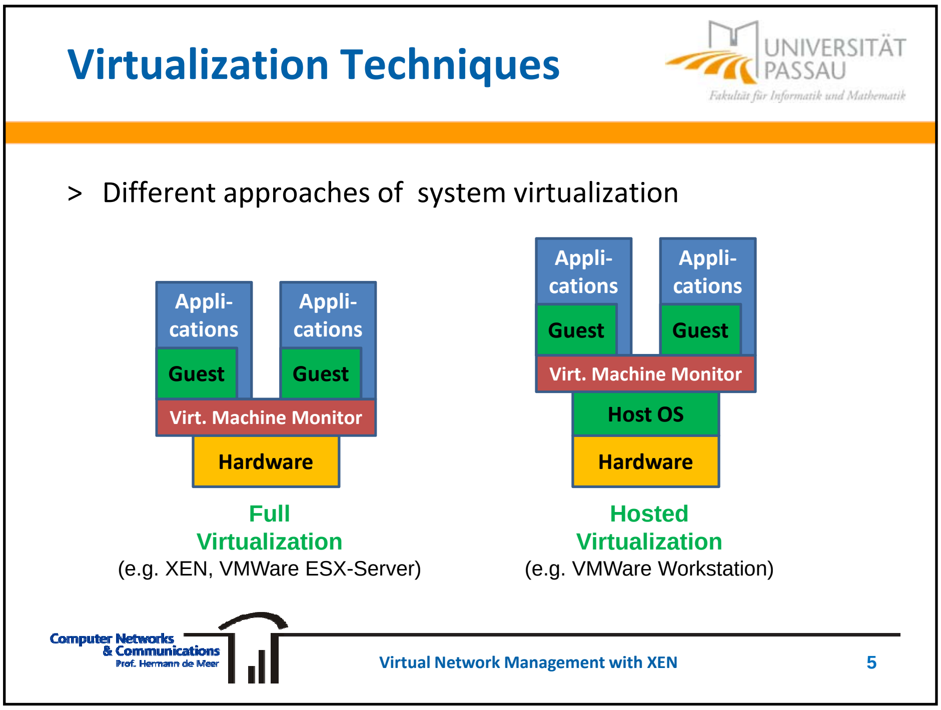
> Virtualization of networks can help to overcome these problems



Virtualization Techniques

> Process virtualization
  - Virtualization of resources for a process
  - Process runs on virtual CPU, uses virtual memory, etc.
  - Space sharing, time sharing / multitasking
  - Example: Java VM

> System virtualization
  - Virtualization of full systems
  - OS runs on virtual hardware
    - Virtual CPU, memory, disk, graphic, sound, network interface card…



Virtualization Techniques

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

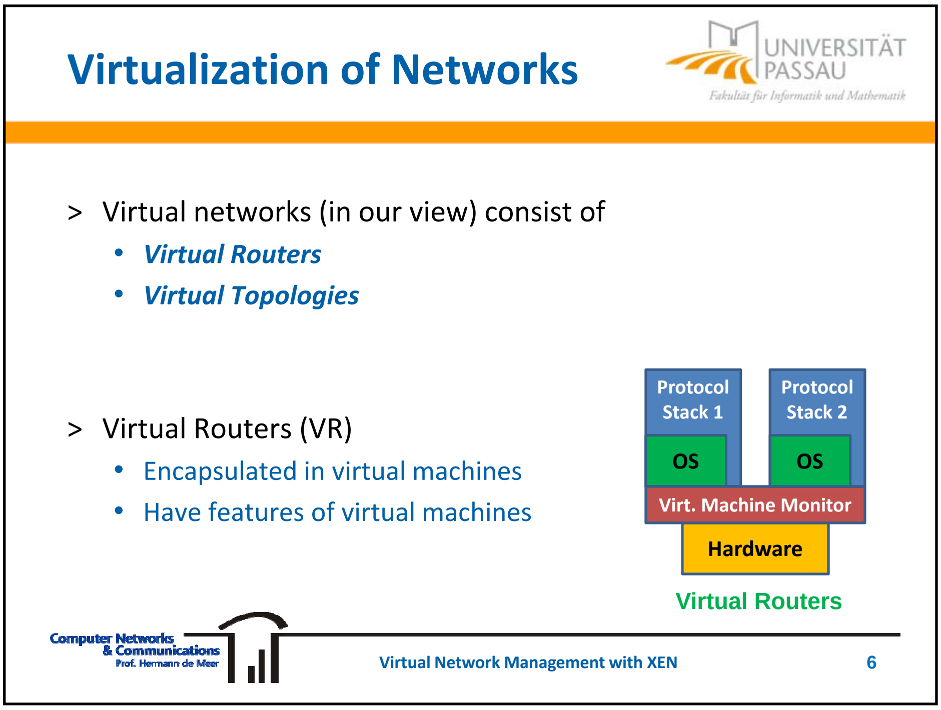
> Different approaches of system virtualization

Appli-cations | Appli-cations
Guest | Guest
Virt. Machine Monitor
Hardware

Full Virtualization
(e.g. XEN, VMWare ESX-Server)

Appli-cations | Appli-cations
Guest | Guest
Virt. Machine Monitor
Host OS
Hardware

Hosted Virtualization
(e.g. VMWare Workstation)



Virtualization of Networks

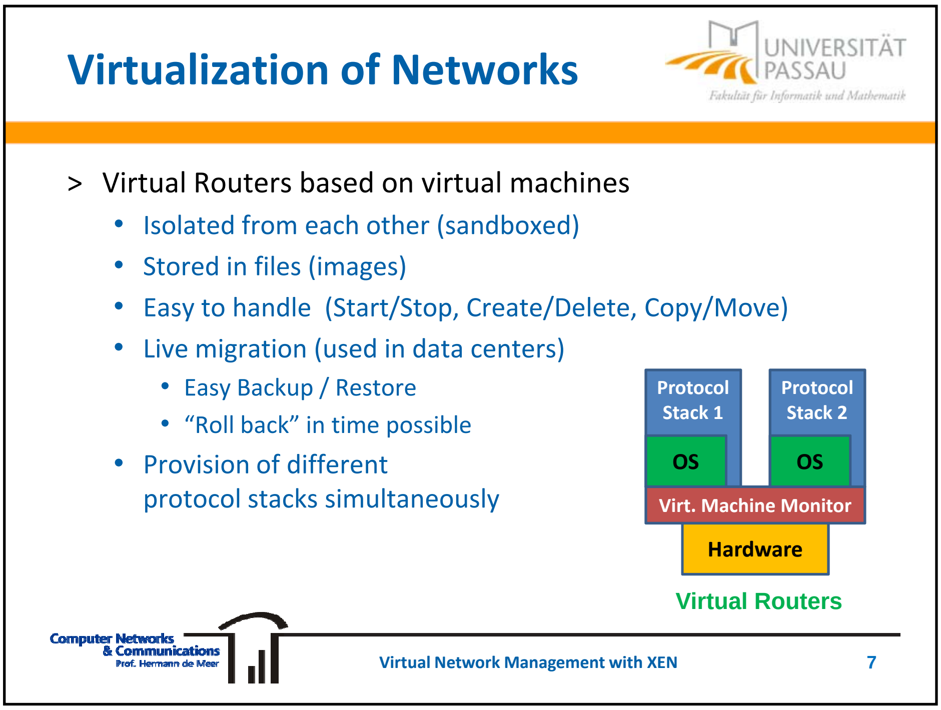
UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

> Virtual networks (in our view) consist of
- ***Virtual Routers***
- ***Virtual Topologies***

> Virtual Routers (VR)
- Encapsulated in virtual machines
- Have features of virtual machines

Protocol Stack 1 | Protocol Stack 2
OS | OS
Virt. Machine Monitor
Hardware

Virtual Routers

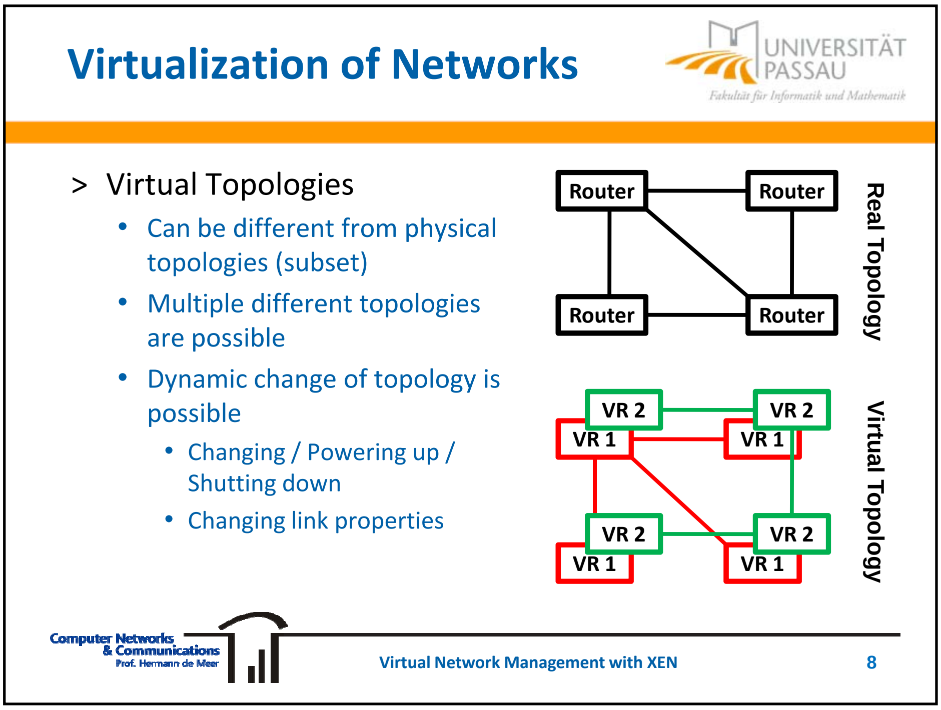


Virtualization of Networks

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

> Virtual Routers based on virtual machines
- Isolated from each other (sandboxed)
- Stored in files (images)
- Easy to handle (Start/Stop, Create/Delete, Copy/Move)
- Live migration (used in data centers)
  - Easy Backup / Restore
  - “Roll back” in time possible
- Provision of different protocol stacks simultaneously

Virtual Routers



Virtualization of Networks

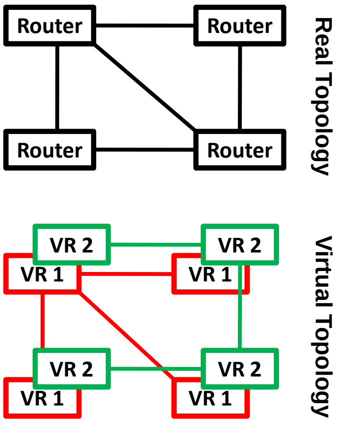
UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

> Virtual Topologies
- Can be different from physical topologies (subset)
- Multiple different topologies are possible
- Dynamic change of topology is possible
  - Changing / Powering up / Shutting down
  - Changing link properties

Router
Router
Router
Router
Real Topology
VR 2
VR 1
VR 2
VR 1
VR 2
VR 1
VR 2
VR 1
Virtual Topology

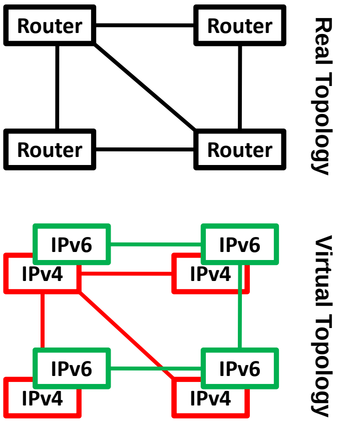
Benefits of Virtual Networks

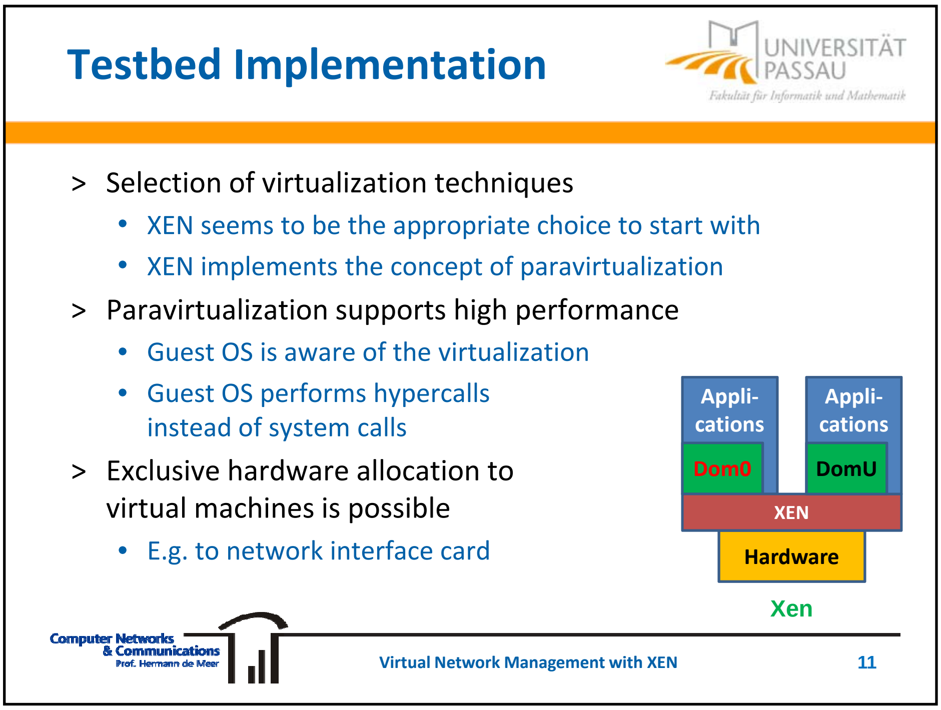
- > Create networks with different characteristics
- > Adapt to service demands
  - Optimized topology
  - Adjustable link properties (e.g. bandwidth)
- > Dynamic reconfiguration
  - Within hours
  - Network can adapt to changing business rules



Benefits of Virtual Networks

- > Encapsulation: Different networks don't interfere with each other
- > Use different techniques in parallel
  - E.g. IPv4/IPv6
  - Smooth transition possible
- > Add new functionality (IPv8?) without disturbing legacy network





Testbed Implementation

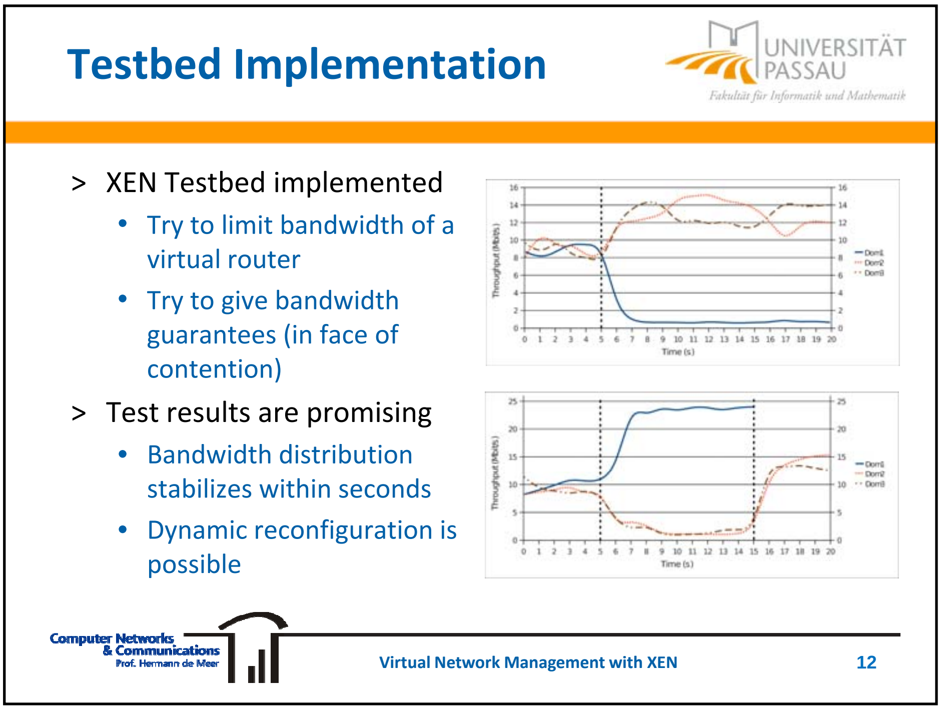
UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

> Selection of virtualization techniques
- XEN seems to be the appropriate choice to start with
- XEN implements the concept of paravirtualization

> Paravirtualization supports high performance
- Guest OS is aware of the virtualization
- Guest OS performs hypercalls instead of system calls

> Exclusive hardware allocation to virtual machines is possible
- E.g. to network interface card

Computer Networks & Communications
Prof. Hermann de Meer



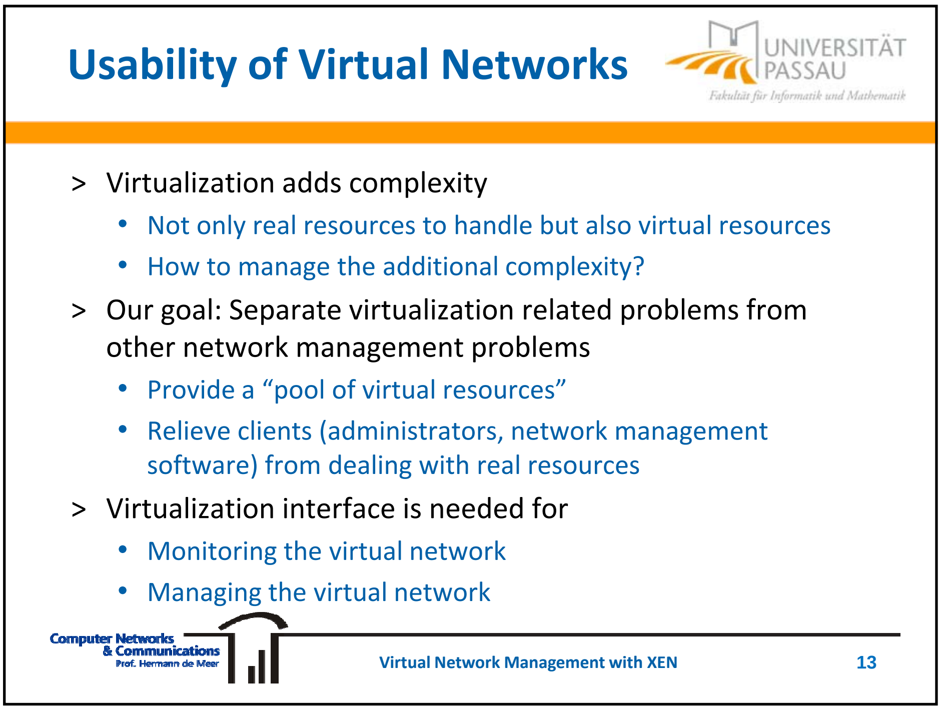
Testbed Implementation

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

> XEN Testbed implemented
- Try to limit bandwidth of a virtual router
- Try to give bandwidth guarantees (in face of contention)

> Test results are promising
- Bandwidth distribution stabilizes within seconds
- Dynamic reconfiguration is possible

Computer Networks & Communications
Prof. Hermann de Meer



Usability of Virtual Networks

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

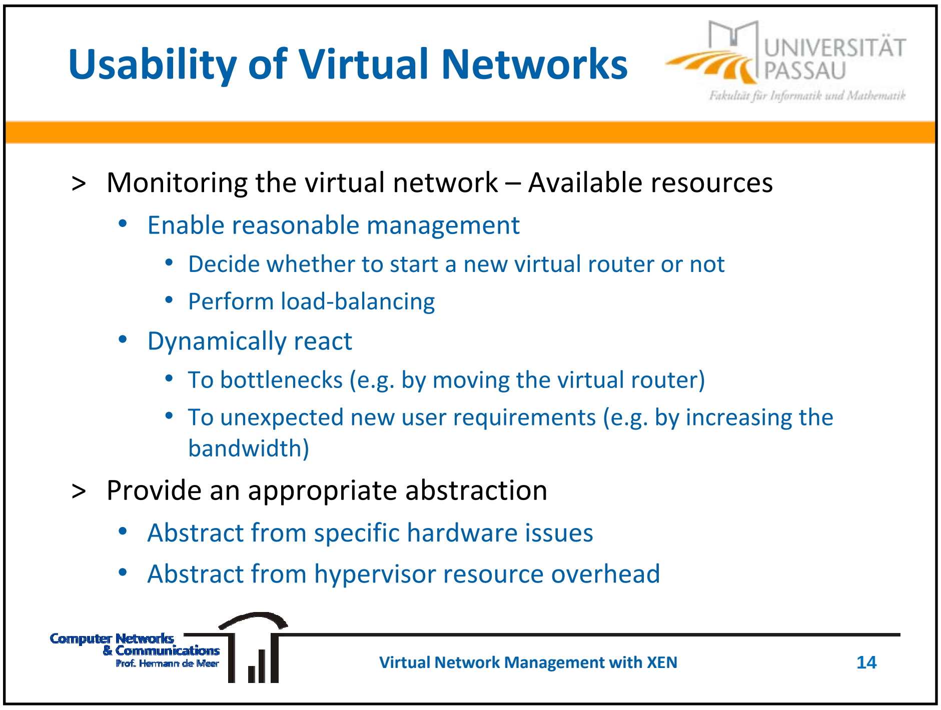
> Virtualization adds complexity
- Not only real resources to handle but also virtual resources
- How to manage the additional complexity?

> Our goal: Separate virtualization related problems from other network management problems
- Provide a “pool of virtual resources”
- Relieve clients (administrators, network management software) from dealing with real resources

> Virtualization interface is needed for
- Monitoring the virtual network
- Managing the virtual network

Computer Networks & Communications Prof. Hermann de Meer

Virtual Network Management with XEN 13



Usability of Virtual Networks

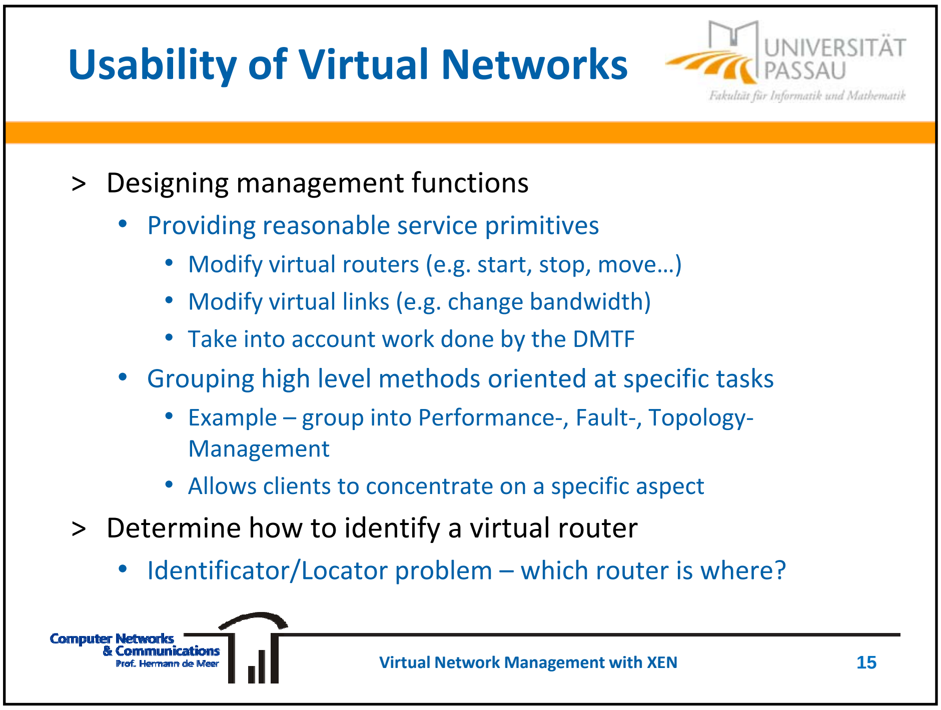
UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

> Monitoring the virtual network – Available resources
- Enable reasonable management
  - Decide whether to start a new virtual router or not
  - Perform load-balancing
- Dynamically react
  - To bottlenecks (e.g. by moving the virtual router)
  - To unexpected new user requirements (e.g. by increasing the bandwidth)

> Provide an appropriate abstraction
- Abstract from specific hardware issues
- Abstract from hypervisor resource overhead

Computer Networks & Communications Prof. Hermann de Meer

Virtual Network Management with XEN 14



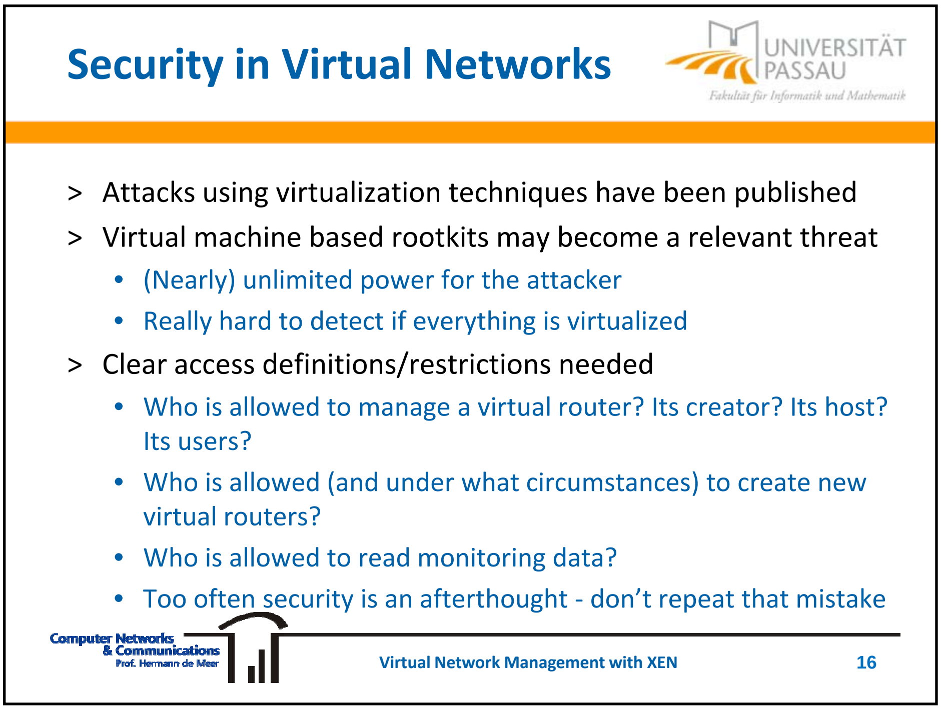
Usability of Virtual Networks

> Designing management functions

- Providing reasonable service primitives
  - Modify virtual routers (e.g. start, stop, move…)
  - Modify virtual links (e.g. change bandwidth)
  - Take into account work done by the DMTF
- Grouping high level methods oriented at specific tasks
  - Example – group into Performance-, Fault-, Topology-Management
  - Allows clients to concentrate on a specific aspect

> Determine how to identify a virtual router

- Identificator/Locator problem – which router is where?



Security in Virtual Networks

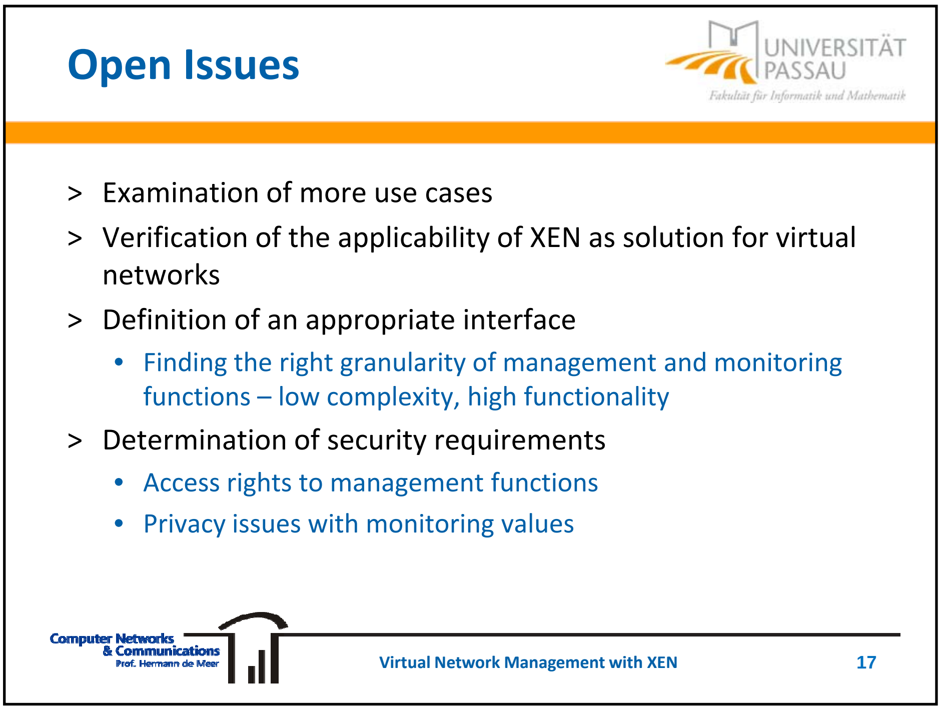
> Attacks using virtualization techniques have been published

> Virtual machine based rootkits may become a relevant threat

- (Nearly) unlimited power for the attacker
- Really hard to detect if everything is virtualized

> Clear access definitions/restrictions needed

- Who is allowed to manage a virtual router? Its creator? Its host? Its users?
- Who is allowed (and under what circumstances) to create new virtual routers?
- Who is allowed to read monitoring data?
- Too often security is an afterthought - don’t repeat that mistake



Open Issues

> Examination of more use cases

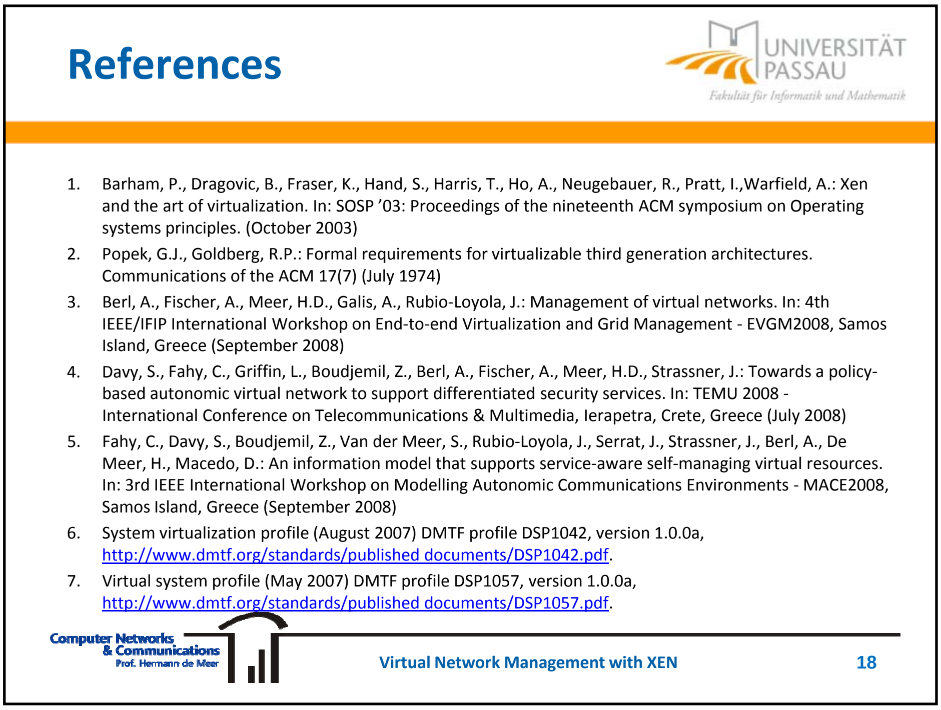
> Verification of the applicability of XEN as solution for virtual networks

> Definition of an appropriate interface

- Finding the right granularity of management and monitoring functions – low complexity, high functionality

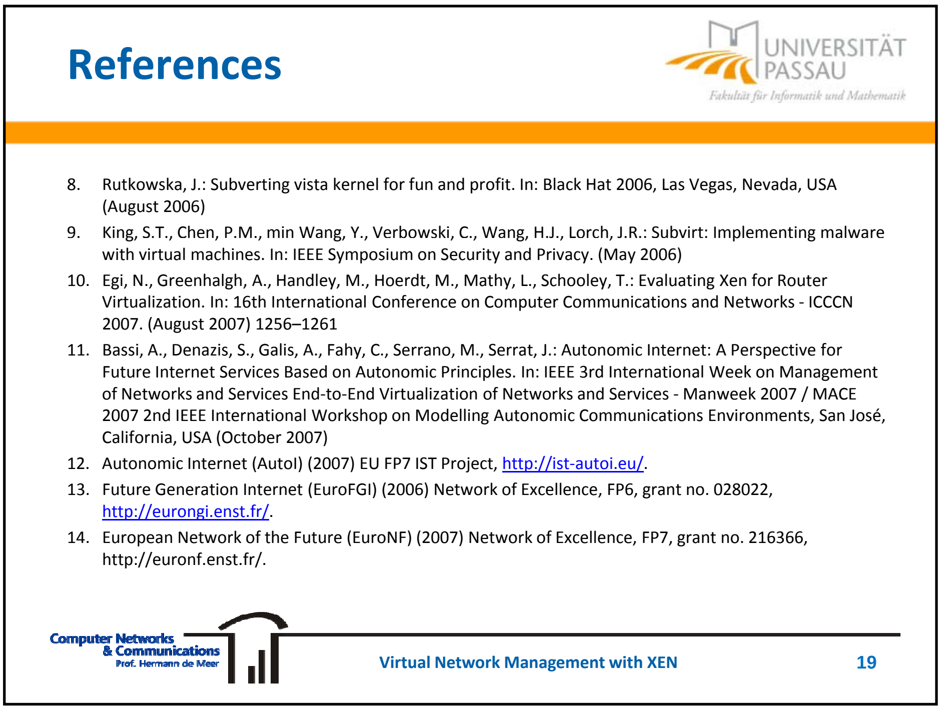
> Determination of security requirements

- Access rights to management functions
- Privacy issues with monitoring values



References

1. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I.,Warfield, A.: Xen and the art of virtualization. In: SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles. (October 2003)
2. Popek, G.J., Goldberg, R.P.: Formal requirements for virtualizable third generation architectures. Communications of the ACM 17(7) (July 1974)
3. Berl, A., Fischer, A., Meer, H.D., Galis, A., Rubio-Loyola, J.: Management of virtual networks. In: 4th IEEE/IFIP International Workshop on End-to-end Virtualization and Grid Management - EVGM2008, Samos Island, Greece (September 2008)
4. Davy, S., Fahy, C., Griffin, L., Boudjemil, Z., Berl, A., Fischer, A., Meer, H.D., Strassner, J.: Towards a policy-based autonomic virtual network to support differentiated security services. In: TEMU 2008 - International Conference on Telecommunications & Multimedia, Ierapetra, Crete, Greece (July 2008)
5. Fahy, C., Davy, S., Boudjemil, Z., Van der Meer, S., Rubio-Loyola, J., Serrat, J., Strassner, J., Berl, A., De Meer, H., Macedo, D.: An information model that supports service-aware self-managing virtual resources. In: 3rd IEEE International Workshop on Modelling Autonomic Communications Environments - MACE2008, Samos Island, Greece (September 2008)
6. System virtualization profile (August 2007) DMTF profile DSP1042, version 1.0.0a, http://www.dmtf.org/standards/published documents/DSP1042.pdf.
7. Virtual system profile (May 2007) DMTF profile DSP1057, version 1.0.0a, http://www.dmtf.org/standards/published documents/DSP1057.pdf.



References

8. Rutkowska, J.: Subverting vista kernel for fun and profit. In: Black Hat 2006, Las Vegas, Nevada, USA (August 2006)
9. King, S.T., Chen, P.M., min Wang, Y., Verbowski, C., Wang, H.J., Lorch, J.R.: Subvirt: Implementing malware with virtual machines. In: IEEE Symposium on Security and Privacy. (May 2006)
10. Egi, N., Greenhalgh, A., Handley, M., Hoerdt, M., Mathy, L., Schooley, T.: Evaluating Xen for Router Virtualization. In: 16th International Conference on Computer Communications and Networks - ICCCN 2007. (August 2007) 1256–1261
11. Bassi, A., Denazis, S., Galis, A., Fahy, C., Serrano, M., Serrat, J.: Autonomic Internet: A Perspective for Future Internet Services Based on Autonomic Principles. In: IEEE 3rd International Week on Management of Networks and Services End-to-End Virtualization of Networks and Services - Manweek 2007 / MACE 2007 2nd IEEE International Workshop on Modelling Autonomic Communications Environments, San José, California, USA (October 2007)
12. Autonomic Internet (AutoI) (2007) EU FP7 IST Project, http://ist-autoi.eu/.
13. Future Generation Internet (EuroFGI) (2006) Network of Excellence, FP6, grant no. 028022, http://eurongi.enst.fr/.
14. European Network of the Future (EuroNF) (2007) Network of Excellence, FP7, grant no. 216366, http://euronf.enst.fr/.