science + computing

an atos company

Moby Dock im Windkanal

Chancen und Einsatzzwecke von Docker im Computing

CeBIT Open Source Forum 2015

Dipl.-Inform. (FH) Holger Gantikow
science + computing ag
IT-Dienstleistungen und Software für anspruchsvolle Rechnernetze
Tübingen | München | Berlin | Düsseldorf

Holger Gantikow

Dipl.-Inform. (FH)

System Engineer
science + computing ag STANDARD

72070 Tübingen
Deutschland

Persönliches

Ich suche	neue Kontakte
Ich biete	Abgeschlossene Diplomarbeit ("Virtualisierung im Kontext von Hocherfügbarkeit"), IT-Know-How, Erfahrung mit Linux, speziell Debian&Red Hat, Windows, Mac OS X, Solaris, *BSD, HP-UX, AIX, Netzwerkadministration, Netzwerktechnik, Hardware, Asterisk, VoIP-Systeme, Server Administration, Cluster Computing, Hochverfügbarkeit, Virtualisierung, HA, HPC, Autor von Fachartikeln zum Thema Cloud Computing, RHCT, RHSA, Python Programmierung, Neugierde, Flexibilität
Interessen	IT-spezifisch momentan: Virtualisierung (Xen, ESX, ESXi, KVM), Cluster Computing (HPC, HA), Cloud Computing (speziell: IaasS, HPCaaS), OpenNebula, OpenSolaris, ZFS, XMPP, SunRay ThinClients - ansonsten: Freie Software, Musik, Gitarre, Fotografie

https://www.xing.com/profile/Holger_Gantikow

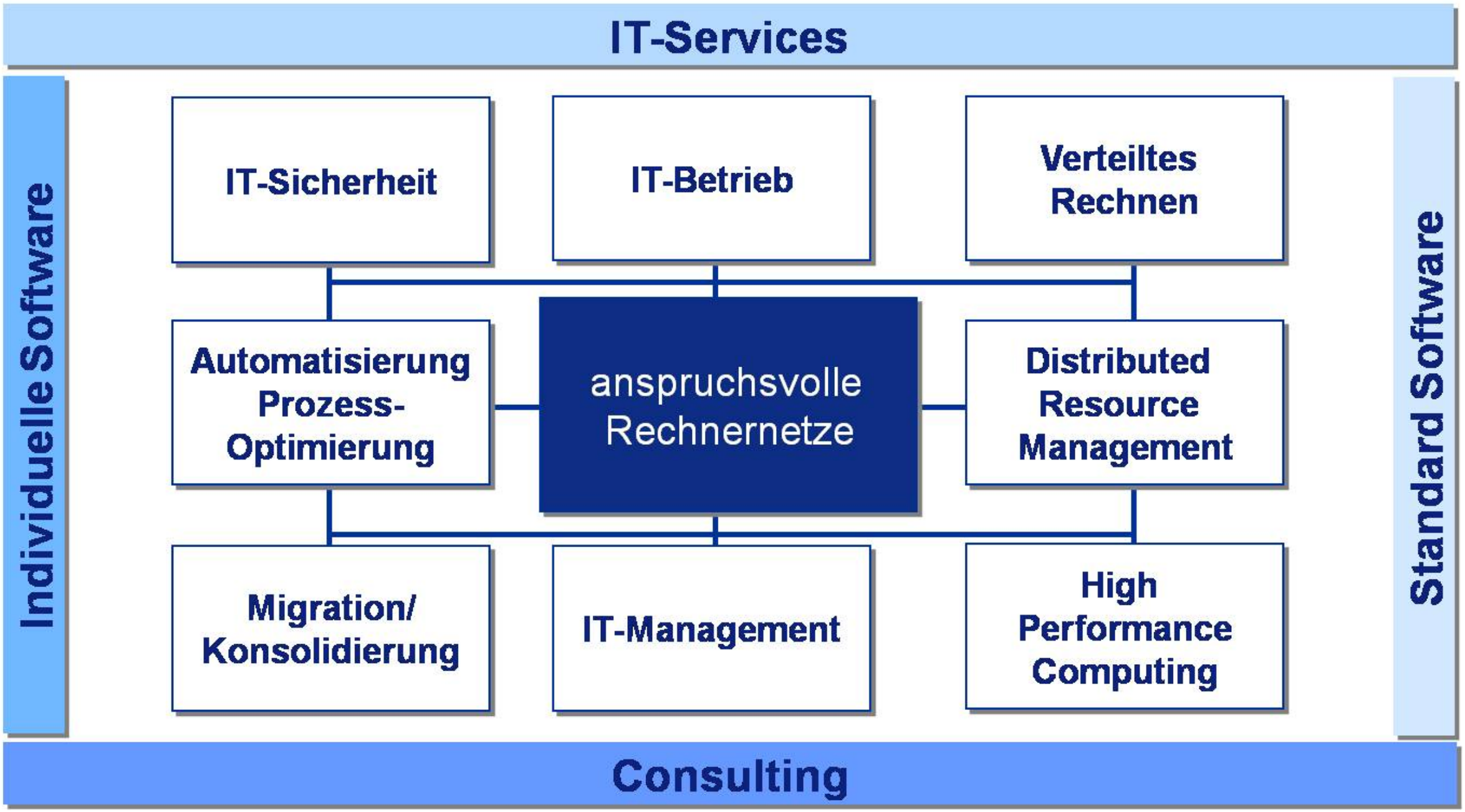
science+computing ag

Unser Fokus:
IT-Dienstleistungen und Software für technische Berechnungsumgebungen

Gegründet	**1989**
Büros	**Tübingen** **München** **Berlin** **Düsseldorf** **Ingolstadt**
Mitarbeiter	**~320**
Besitzer	**Bull S.A. (100%)** **seit 10/2008**
Jahresumsatz (2013)	**30,7 Mio. Euro**

Kernkompetenzen

science + computing
an atos company



Agenda

science + computing
an atos company

Teil I: Docker im HPC

Teil II: Benchmarks

Teil III: Security-Aspekte

science + computing
an atos company

Teil 0:
Einleitung

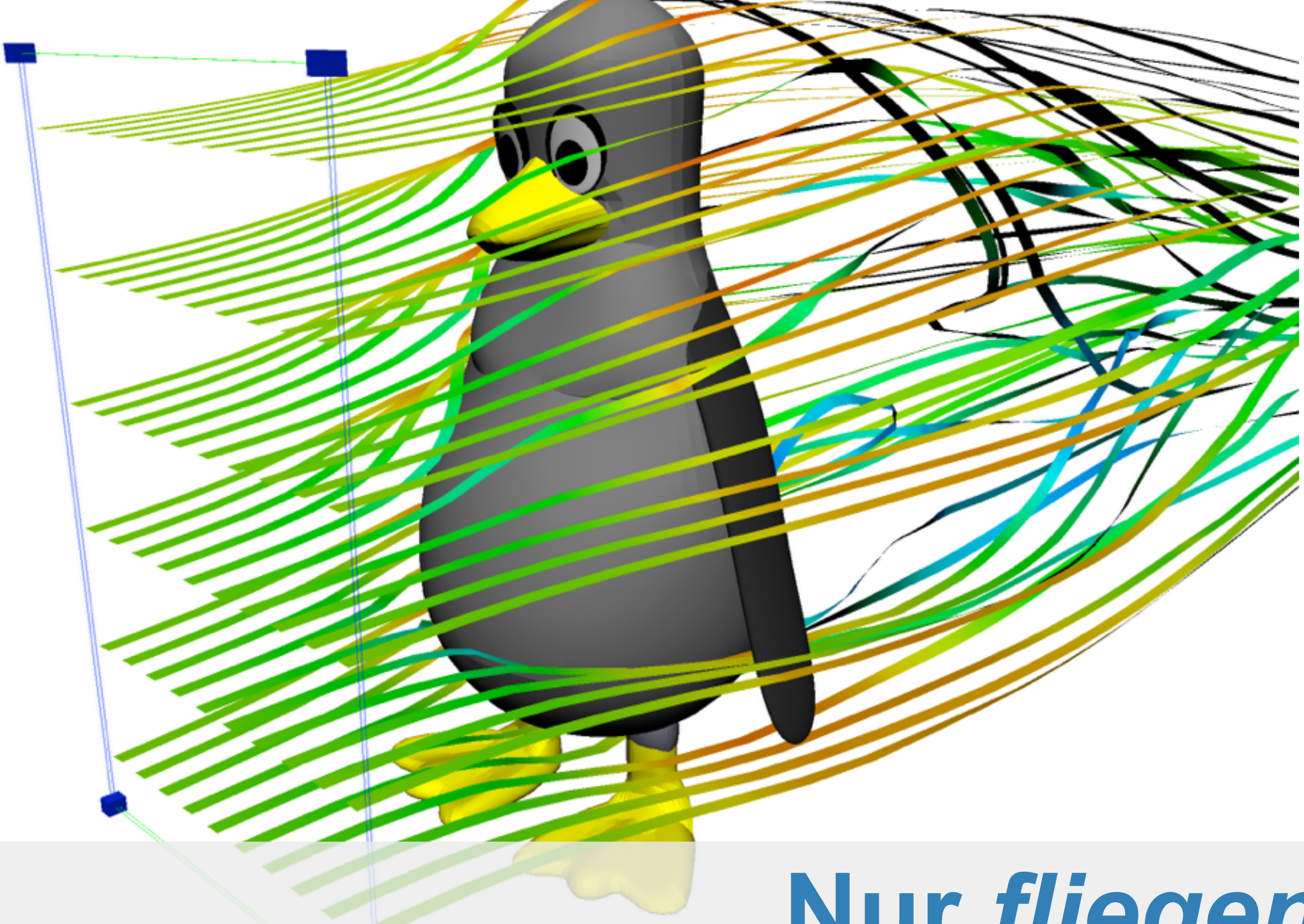
science + computing

an atos company

Was ist dieses *HPC*?

High Performance Computing

Foto: Dieter Both, Bull GmbH



Grafik: Dr. Martin Schulz

Nur *fliegende* Pinguine?

Enterprise IT vs. HPC

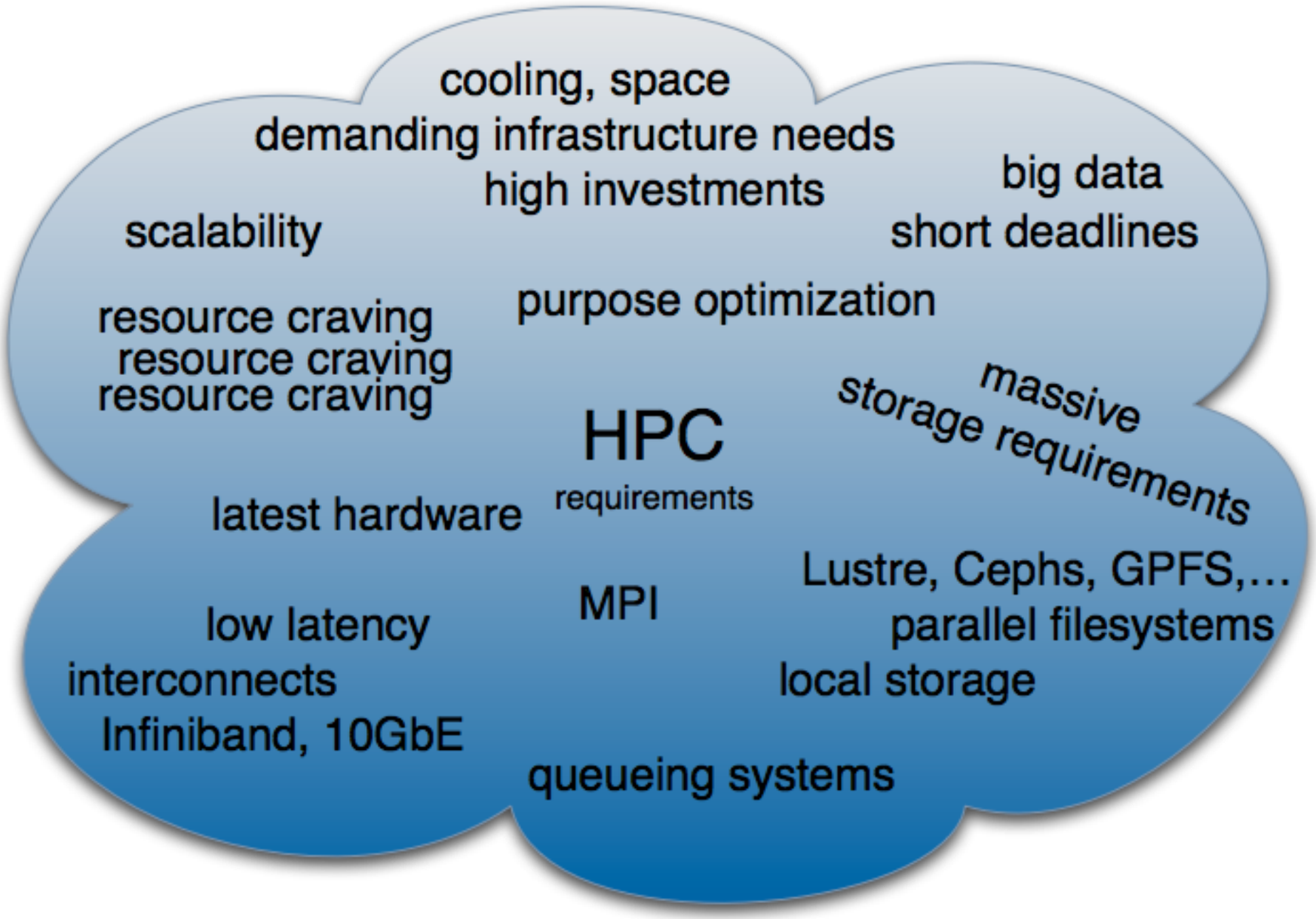
Enterprise IT vs. HPC

science + computing
an atos company

	Enterprise IT	HPC Zentren
Auslastung	< 50%	>>50% (teils bis zu 90%)
Einsatz von Virtualisierung	ja	nein bis selten
Art der Systeme	heterogen	homogen
scale ...	**in** Konsolidierung via Virtualisierung	**out** Berechnung verteilt auf mehrere Hosts
Verbindungs-Anforderungen	gemäßigt	Bedarf an latenzarmen Verbindungen mit hohem Durchsatz
Ressourcen-Verbrauch	endlich - meist von Anwenderanzahl abhängig	quasi unendlich - was vorhanden ist wird genutzt
real. Bezugsart (Cloud)	SaaS (Software as a Service)	IaaS (Infrastructure as a Service)
Dateisysteme	zen	

Fazit:
- HPC unterscheidet sich deutlich von Enterprise IT

HPC-*Buzzword*-Cloud



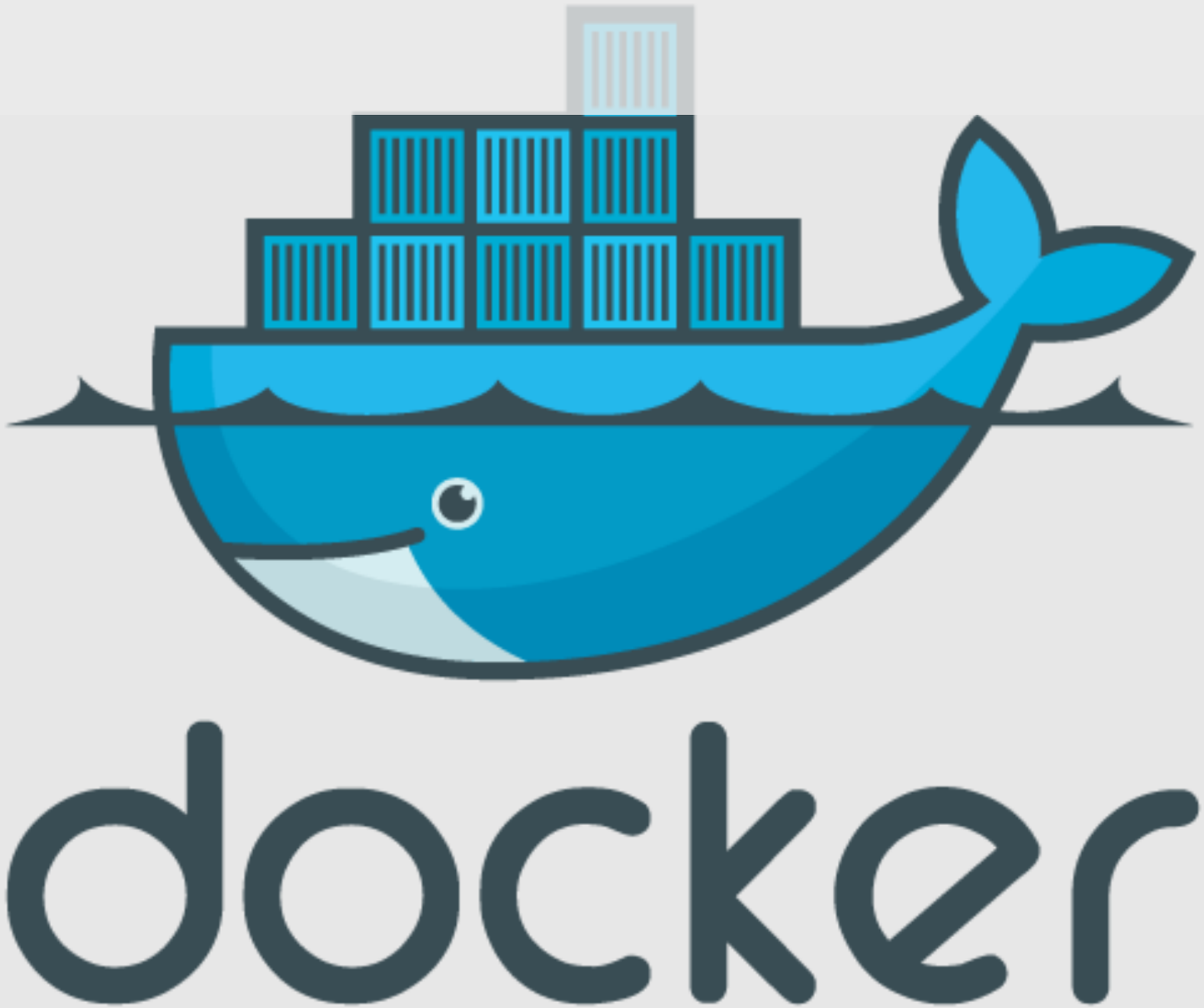
science + computing

an atos company

Docker

Quelle:
http://cdn.meme.am/instances/500x/59600465.jpg

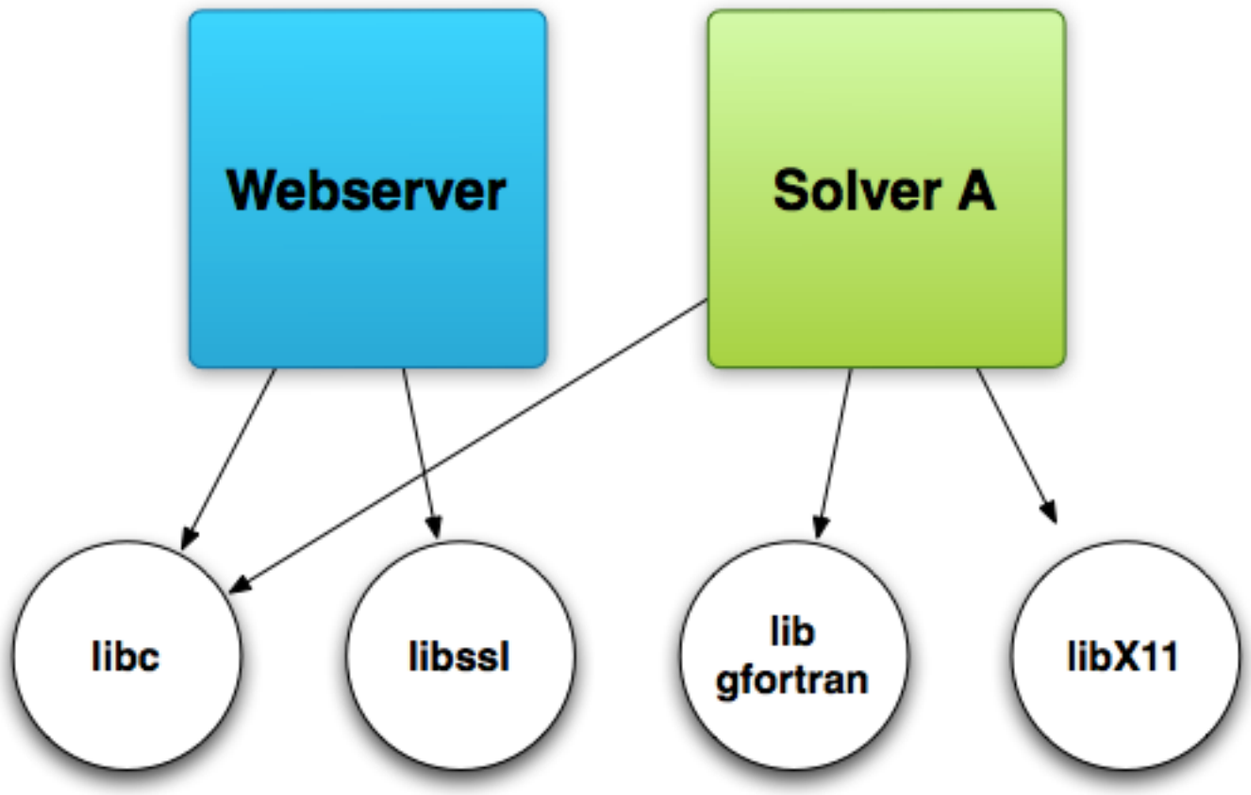
Docker 101



Quelle:
http://blog.docker.com/wp-content/uploads/2013/06/Docker-logo-011.png

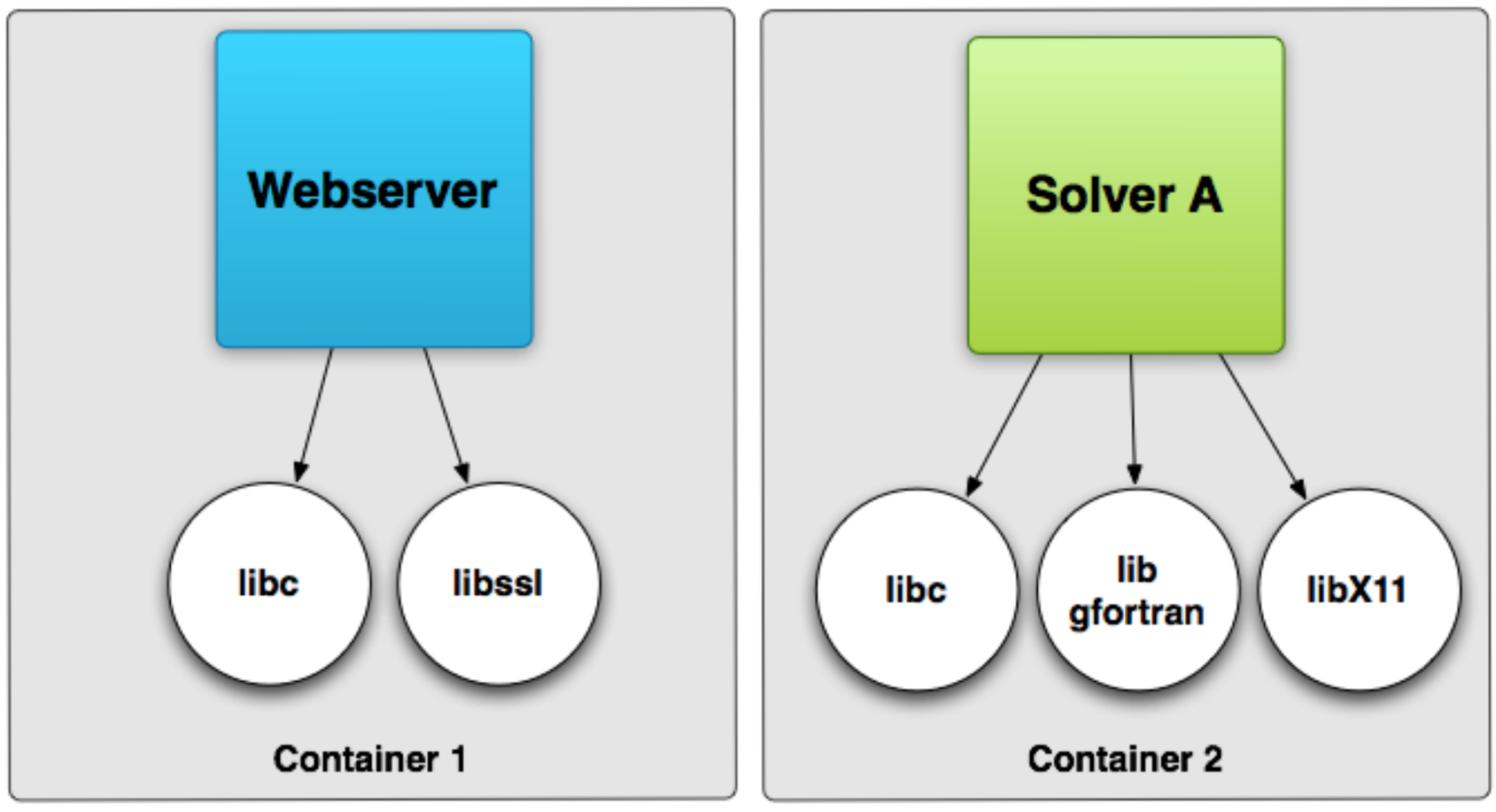
Aufbau *ohne* Docker

science + computing
an atos company

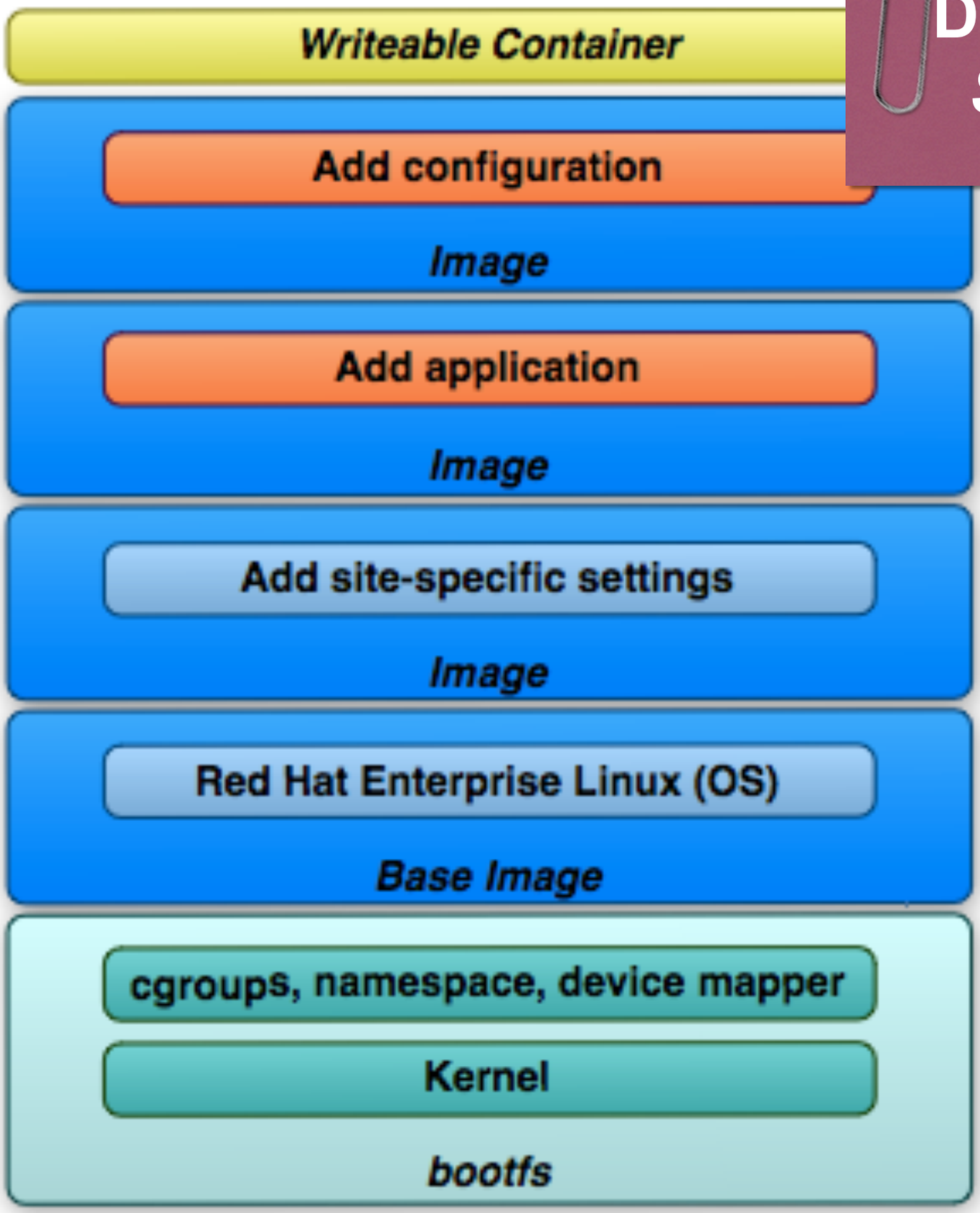


Aufbau *mit* Docker

science + computing
an atos company



Dateisystem *Schichten*

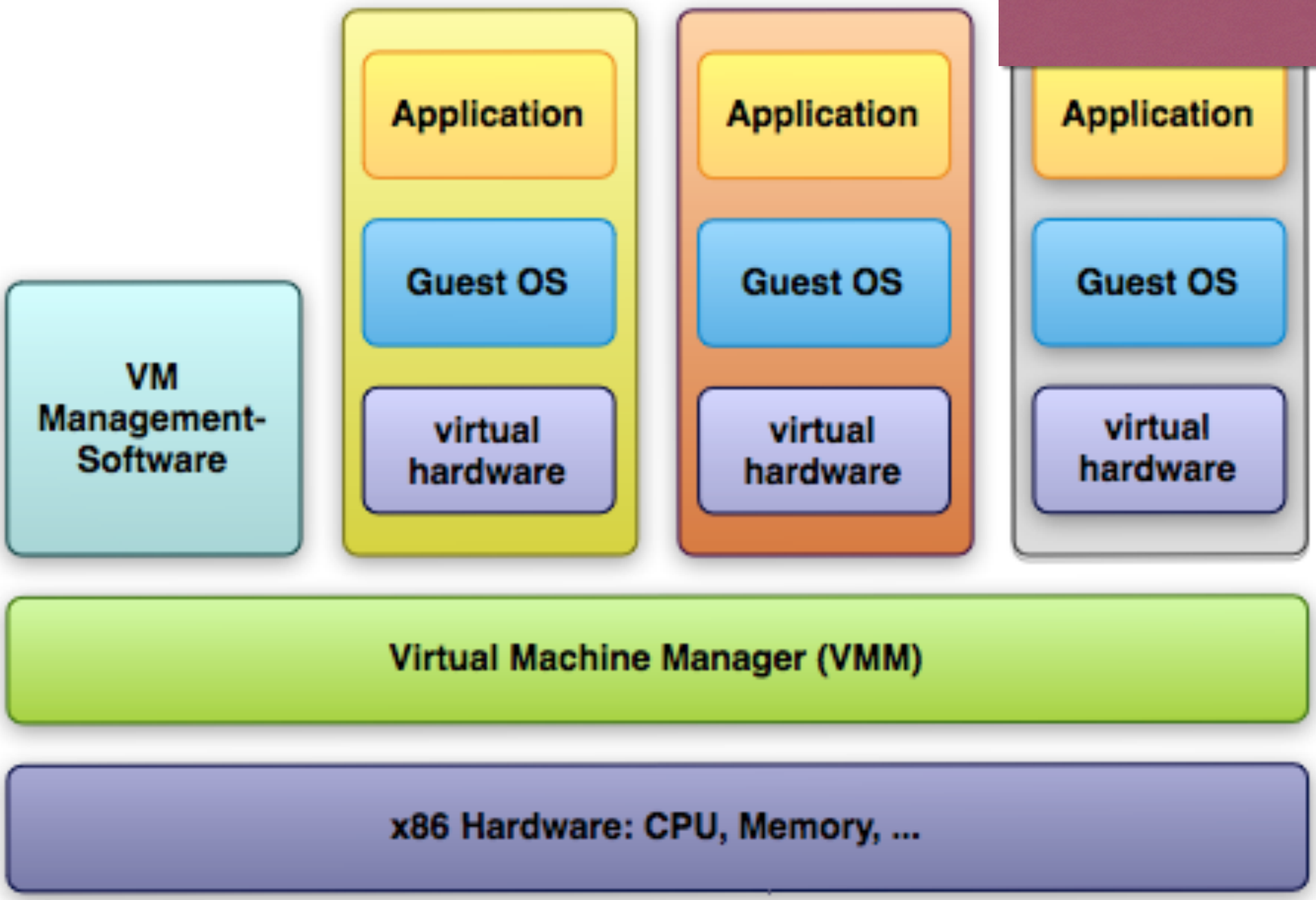


science + computing

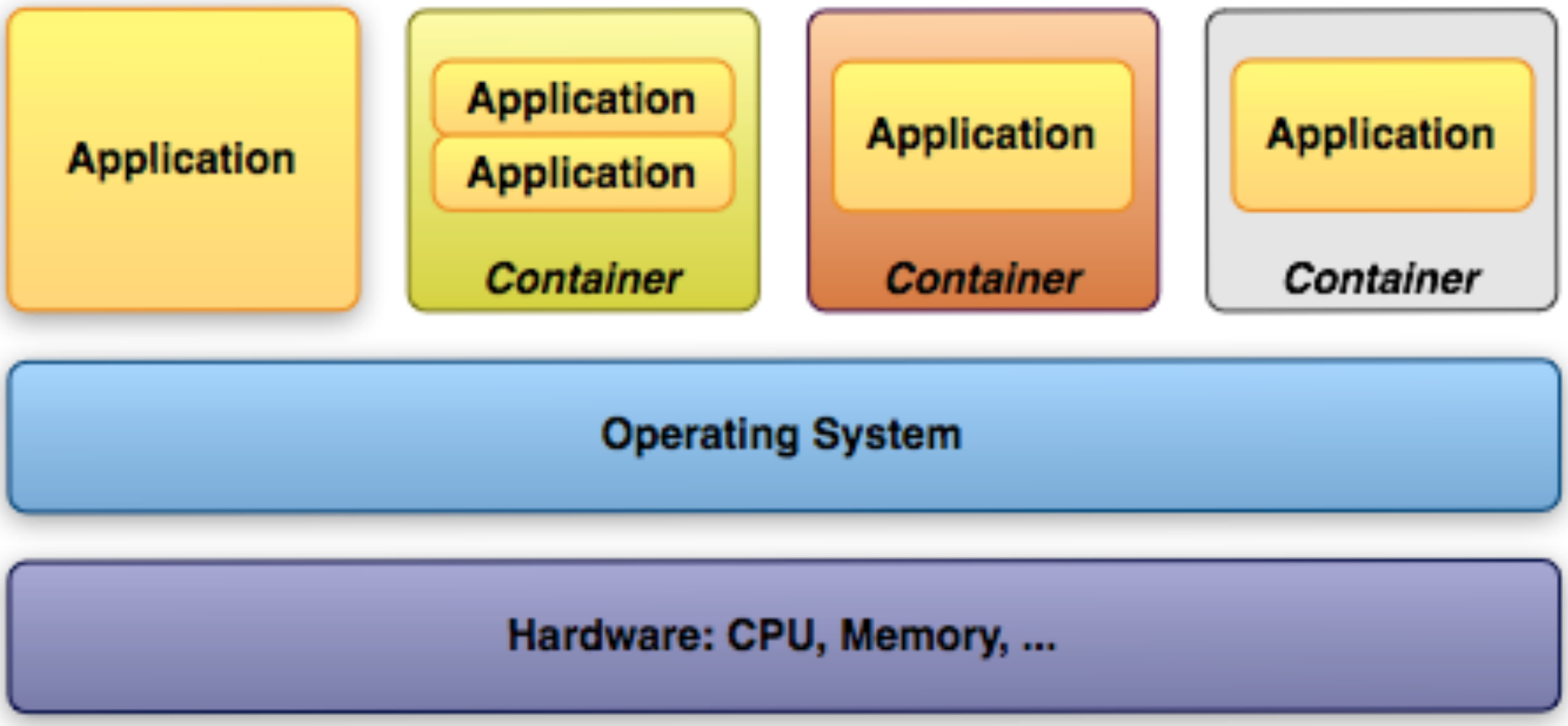
an atos company

Virtualisierung 2.0???

Typ 1
Virtualisierung



Container Virtualisierung



science + computing

an atos company

Und sonst so?

science + computing

an atos company

Teil I:

Docker im HPC

science + computing

an atos company

Wo drückt's?

science + computing

an atos company

Saubere Trennung

science + computing

an atos company

Deployment!

science + computing

an atos company

Heterogene Cluster homogenisieren

science + computing

an atos company

Umgebung weitergeben?

science + computing

an atos company

weitere Ressourcen?

science + computing

an atos company

Warum nicht *klassische Virtualisierung?*

science + computing

an atos company

Teil II:

Benchmarks

Benchmarks...

FINAL-closer2015.pdf (Seite 1 von 7)

Container-Based Virtualization for HPC

Holger Gantikow[1], Sebastian Klingberg[1], Christoph Reich[2]

Keywords: Cont

Abstract: Expe insta comp This Com (Doc analy

1 INTROD

Applications i Computing (HPC) comes to resource put and interconn are traditionally r on physical systen called clusters.

Such a cluster formance, but of up: a) The operati

vmVcontainers.pdf (Seite 1 von 15)

Suchen

RC25482 (AUS1407-001) July 21, 2014
Computer Science

IBM Research Report

An Updated Performance Comparison of Virtual Machines and Linux Containers

Wes Felter, Alexandre Ferreira, Ram Rajamony, Juan Rubio
IBM Research Division
Austin Research Laboratory
11501 Burnet Road
Austin, TX 78758
USA

IBM Docker Paper

RC25482 (AUS1407-001) July 21, 2014
Computer Science

IBM Research Report

An Updated Performance Comparison of Virtual Machines and Linux Containers

Wes Felter, Alexandre Ferreira, Ram Rajamony, Juan Rubio
IBM Research Division
Austin Research Laboratory
11501 Burnet Road
Austin, TX 78758
USA

Quelle: Google: *ibm docker paper* oder:
http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf

IBM Docker Paper - Aufbau

3. Evaluation

In this paper we seek to isolate and understand the overhead introduced by virtual machines (specifically KVM) and containers (specifically Docker) relative to non-virtualized Linux. The fact that Linux can host both VMs and containers creates the opportunity for an apples-to-apples comparison between the two technologies with fewer confounding variables than many previous comparisons. We attempt such a comparison in this paper.

We do not evaluate the case of containers running inside VMs or VMs running inside containers because we consider such double virtualization to be redundant (at least from a performance perspective). To measure overhead we have configured our benchmarks to saturate the resources of the system under test. Docker containers were not restricted by cgroups so they could consume the full resources of the system under test. Likewise, VMs were configured with 32 vCPUs and adequate RAM to hold the benchmark's working set. We use microbenchmarks to individually measure CPU, memory, network, and storage overhead. We also measure two real server applications: Redis and MySQL.

All of these tests were performed on an IBM®System x3650 M4 server with two 2.4-3.0 GHz Intel Sandy Bridge-EP Xeon E5-2665 processors for a total of 16 cores (plus HyperThreading) and 256 GB of RAM. The two processors/sockets are connected by QPI links making this a non-uniform memory access (NUMA) system. This is a mainstream server configuration that is very similar to those used by popular cloud providers [illegible] 3.10 (Saucy)

formation to VMs, so the guest OS believes it is running on a 32-socket system with one core per socket. This is a double-edged sword; abstracting the hardware can improve portability but it also eliminates some opportunities for optimization. [13]

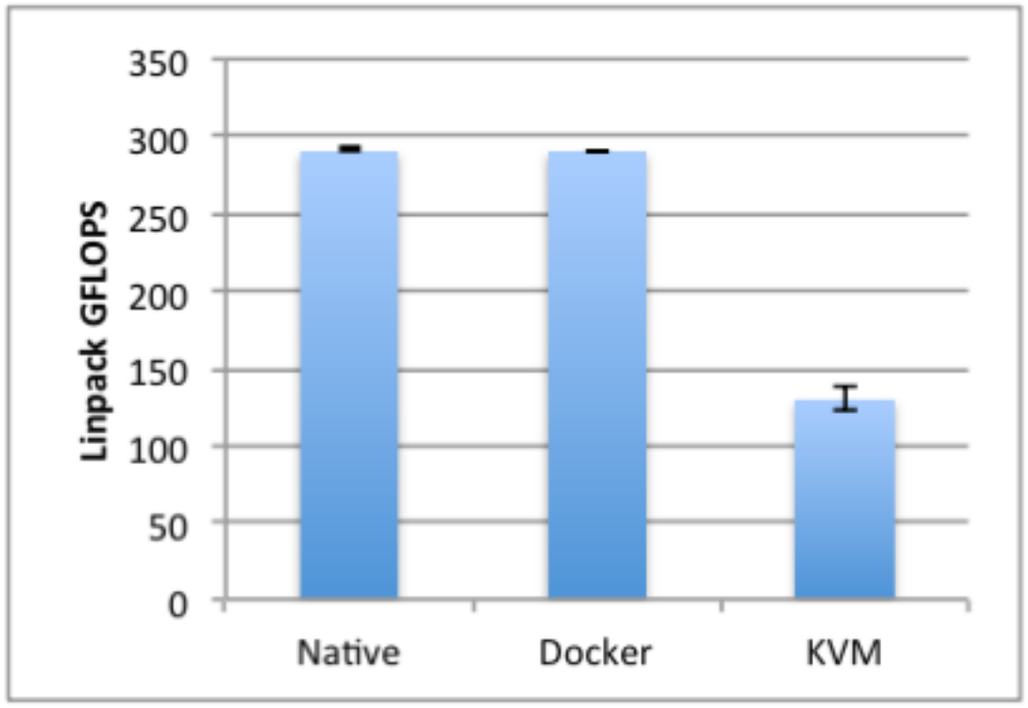


Figure 1. Linpack performance on two sockets (16 cores). Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained over all runs.

Figure 1 shows the performance of Linpack on Linux, Docker, and KVM. A Linpack execution spends the bulk of its time performing mathematical floating point operations. By basing the code on an optimized linear algebra library, the execution gives rise to fairly regular memory accesses [illegible] the floating point capability of the core. [illegible]ore, the math library is highly adaptive [illegible]ovided information to tune itself to the

Quelle: Google: ibm docker paper oder:
http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf

IBM Docker Paper - Ergebnis

Both VMs and containers are mature technology that have benefited from a decade of incremental hardware and software optimizations. In general, Docker equals or exceeds KVM performance in every case we tested. Our results show that both KVM and Docker introduce negligible overhead for CPU and memory performance (except in extreme cases). For I/O-intensive workloads, both forms of virtualization should be used carefully.

We find that KVM performance has improved considerably since its creation. Workloads that used to be considered very challenging, like line-rate 10 Gbps networking, are now possible using only a single core using 2013-era hardware and software. Even using the fastest available forms of paravirtualization, KVM still adds some overhead to every I/O operation; this overhead ranges from significant when performing small I/Os to negligible when it is amortized over large I/Os. Thus, KVM is less suitable for workloads that are latency-sensitive or have high I/O rates. These overheads significantly impact the server applications we tested.

Although containers themselves have almost no overhead, Docker is not without performance gotchas. Docker volumes have noticeably better performance than files stored in AUFS. Docker's NAT also introduces overhead for workloads with high packet rates. These features represent a tradeoff between ease of management and performance and should be considered on a case-by-case basis.

In some sense the comparison can only get worse for containers because they started with near-zero overhead and VMs have gotten faster over time. If containers are to be widely adopted they mus[illegible]s other than

Quelle: Google: ibm docker paper oder:
http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf

that attempting to exploit NUMA in the cloud may be more effort than it is worth. Limiting each workload to a single socket greatly simplifies performance analysis and tuning. Given that cloud applications are generally designed to scale out and the number of cores per socket increases over time, the unit of scaling should probably be the socket rather than the server. This is also a case against bare metal, since a server running one container per socket may actually be faster than spreading the workload across sockets due to the reduced cross-traffic.

In this paper we created single VMs or containers that consumed a whole server; in the cloud it is more common to divide servers into smaller units. This leads to several additional topics worthy of investigation: performance isolation when multiple workloads run on the same server, live resizing of containers and VMs, tradeoffs between scale-up and scale-out, and tradeoffs between live migration and restarting.

Source code

The scripts to run the experiments from this paper are available at https://github.com/thewmf/kvm-docker-comparison.

[illegible]nts

[illegible]ported in part by the Office of Science, [illegible]rtment of Energy under award number

Zusammenfassung

| an atos company

"In general, Docker equals or exceeds KVM performance in every case we tested. [...]

Even using the fastest available forms of par-avirtualization, KVM still adds some overhead to every I/O operation [...].

Thus, KVM is less suitable for workloads that are latency-sensitive or have high I/O rates.

Container vs. bare-metal:
Although **containers themselves have almost no overhead**, **Docker is not without performance gotchas**. **Docker volumes** have noticeably better performance than files stored in AUFS. **Docker's NAT also introduces overhead** for work- loads with high packet rates.
These features represent a **tradeoff between ease of management and performance** and should be considered on a case-by-case basis.

Docker@HPC

Container-Based Virtualization for HPC

Holger Gantikow[1], Sebastian Klingberg[1], Christoph Reich[2]
[1]*science & computing AG, Tübingen, Germany*
[2]*Cloud Research Lab, Furtwangen University, Furtwangen, Germany*
gantikow@gmail.com, klingber@hs-furtwangen.de, christoph.reich@hs-furtwangen.de

Keywords: Container Virtualization, Docker, High Performance Computing, HPC

Abstract: Experts argue that the resource demands of High Performance Computing (HPC) clusters request bare-metal installations. The performance loss of container virtualization is minimal and close to bare-metal, but in comparison has many advantages, like ease of provisioning.
This paper presents the use of the newly adopted container technology and its advantages for High Performance Computing, compared to traditional bare-metal installations or the use of VMs. The setup based on Docker (Docker, 2015) shows the possible use in private HPC sites or public clouds. Ending with a detailed risk analysis of Docker HPC installations.

1 INTRODUCTION

Applications in the domain of High Performance Computing (HPC) have massive requirements when it comes to resources like CPU, memory, I/O throughput and interconnects. This is the reason why they are traditionally run in a bare-metal setup, directly on physical systems, which are interconnected to so-called clusters.

Such a cluster infrastructure offers the best performance, but of disadvantage is the time for setting up: a) The operating system, usually some Linux flavor, must be installed using automatic mechanisms like PXE and Kickstart to install a basic installation ready to log in and get customized. b) All the applications required for computation and general HPC related libraries, like MPI ((MPI), 2015), have to be installed and fine tuned in the customization phase.

or even different versions of the same one, have conflicting environmental requirements, like a specific Linux version or specific library version (e.g. *libc*). This leads to the risk of putting the consistency of the whole cluster at stake, when adding a new application, or a newer version. Libraries might have to be updated, which might imply a
whole Linux operating system (OS
can l
for th
being

N
comp
might
ages,
ning
high
terfer

Danke!
@Sebastian Klingberg

memory footprint, as containers share a lot of resources with the host system as opposed to VMs starting a complete OS and in terms of storage required. Startup time is reduced from the time booting a full OS to the few seconds it takes till the container is ready to use.

For reducing storage requirements Docker makes use of *layered file system images*, usually *UnionFS* (Unionfs, 2015) as a space conserving mechanism, which is lacking in several other container solutions. This allows file systems stacked as layers on top of each other (see Figure 4), which enables sharing and reusing of base layers. For example the base installation of a certain distribution, with individually modified overlays stacked on top, can provide the required application and configuration for several different tasks.

Figure 4: The layers of the Docker file system

4 EXPERIMENTAL EVALUATION

Performance-wise, without all the overhead added by hypervisor and VMs, containers as a light-weight virtualization mechanism can achieve almost the same performance as native execution on a bare-metal system does, as other benchmarks (Felter et al., 2014), (Xavier et al., 2013) underline.

As we were interested in the amount of overhead generated by containerizing a HPC workload, we decided to benchmark a real-world ABAQUS example in different scenarios, comparing the containerized execution time to the native execution. ABAQUS (Abaqus FEA, 2015) is frequently used for finite element analysis (FEA) and computer aided engineering (CAE) for example in the aerospace and automotive industries, as it provides wide material modeling capability and multi-physics capabilities.

The application was installed to local disks on a CentOS 7 SunFire X2200 server with the following hardware configuration:

- CPU: 4x Dual-Core AMD Opteron (tm) 2220
- RAM: 16GB
- HDD: local conventional disks without RAID
- Infiniband: Mellanox MT25204

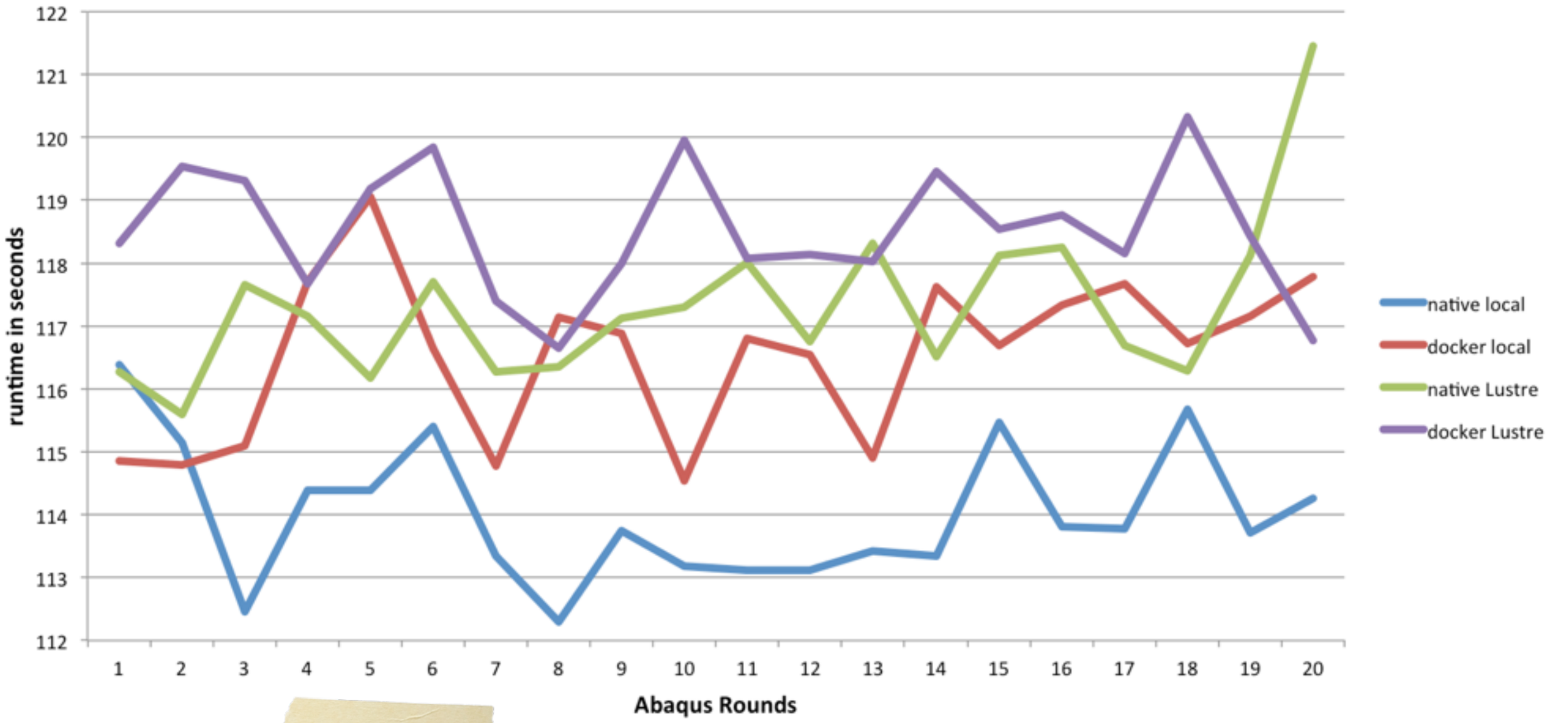
The job used for evaluation is the freely available *s4b* from the *samples.zip* package included in the ABAQUS 6.12-3 installation. It was installed onto local disks and the software licensing was done using a local license file.

As the server provided access to a Lustre parallel

Docker@HPC - Ergebnis

E24 | Zeitverlust

	A	B	C	D	E	F
1	**native local**	docker local	**native Lustre**	docker Lustre		



Zahlen - average runtime (native vs Docker)
local disk: 114s vs 116,5s - overhead: 2,21%
Lustre: 117,3s vs 118,5s - overhead: 1,04%

1,041715506 **Overhead**

Groß-/Kleinschreibung

Summe=0 80%

science + computing
an atos company

Teil III:

Security-Aspekte

science + computing

an atos company

Zitate

Quelle: Surviving the Zombie Apocalyse - Ian Jackson
http://xenbits.xen.org/people/iwj/2015/fosdem-security/

science + computing
an atos company

"Some people make the mistake of thinking of containers as a better and faster way of running virtual machines.
From a security point of view, containers are much weaker."

Dan Walsh,
SELinux architect(?)

science + computing
an atos company

"Virtual Machines might be more secure today,
but containers are definitely catching up."

Jerome Petazzoni,
Senior Software Engineer at Docker

science + computing
an atos company

"You are absolutely deluded, if not stupid, if you think that a worldwide collection of software engineers who can't write operating systems or applications without security holes, can then turn around and suddenly write virtualization layers without security holes."

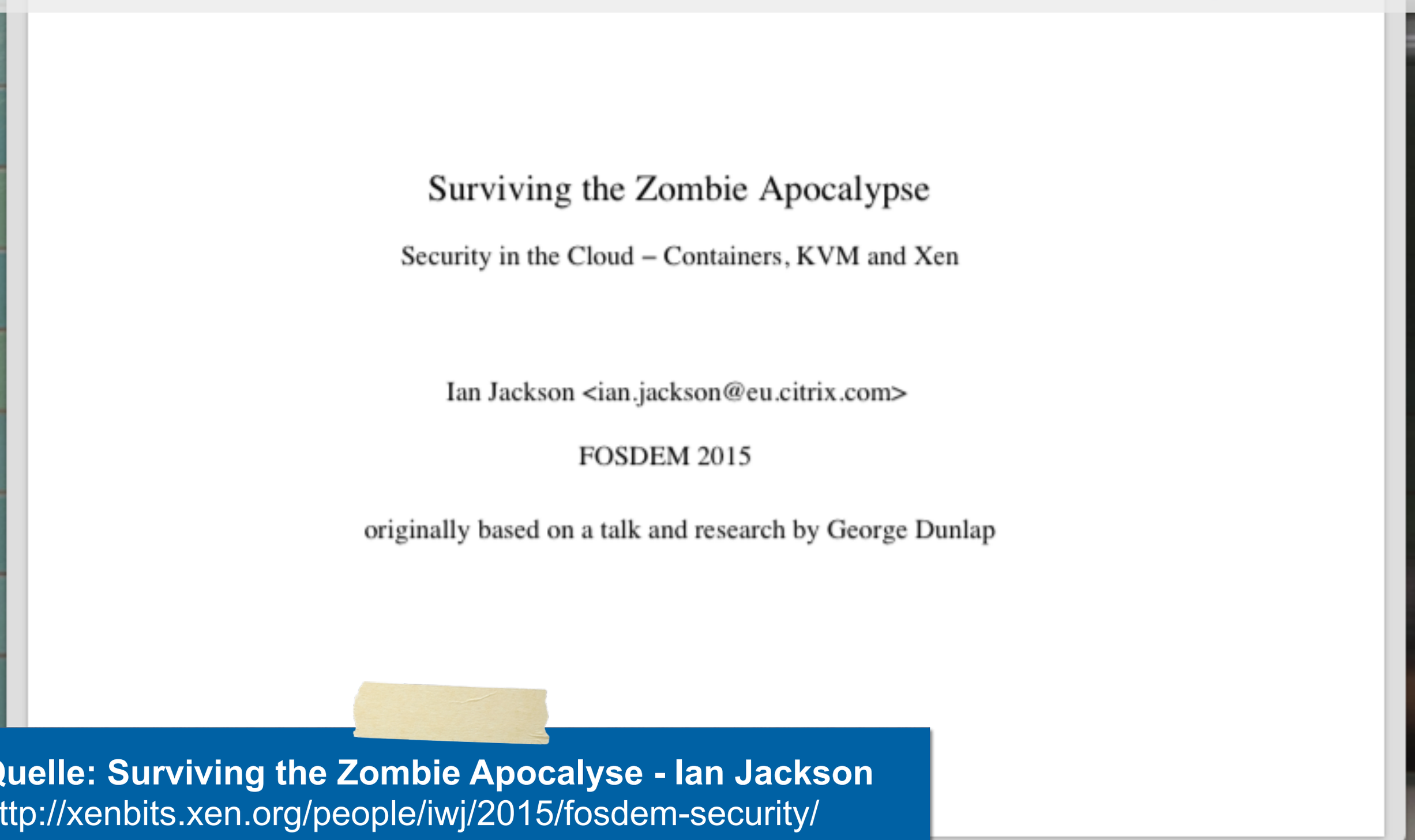
Theo de Raadt,
OpenBSD project lead

science + computing

an atos company

vulnerabilities

Surviving the Zombie Apocalypse



Quelle: Surviving the Zombie Apocalyse - Ian Jackson
http://xenbits.xen.org/people/iwj/2015/fosdem-security/

Zombies? - Findings!

Some Free Software VM hosting technologies
Vulnerabilities published in 2014

	Xen PV	KVM+ QEMU	Linux as general container	Linux app container (non–root)
Privilege escalation (guest–to–host)	0	3–5	7–9	4
Denial of service (by guest of host)	3	5–7	12	3
Information leak (from host to guest)	1	0	1	1
				Hosts only application, not guest OS

Quelle: Surviving the Zombie Apocalyse - Ian Jackson
http://xenbits.xen.org/people/iwj/2015/fosdem-security/

Frustbringer...

Docker - *Missverständnisse*

Jessie Frazelle's Blog

Docker Containers on the Desktop

February 21, 2015

Hello!

If you are not familiar with Docker, it is the popular open source container engine.

Most people use Docker for containing applications to deploy into production or for building their applications in a contained enviornment. This is all fine & dandy, and saves developers & ops engineers huge headaches, but I like to use Docker in a not-so-typical way.

I use Docker to run all the desktop apps on my computers.

But why wo[illegible]t to run all these apps in containers? Well let me [illegible]e OS X

Quelle: Docker Containers on the Desktop
https://blog.jessfraz.com/posts/docker-containers-on-the-desktop.html

Docker - Leichtsinn

Jessie Frazelle's Blog

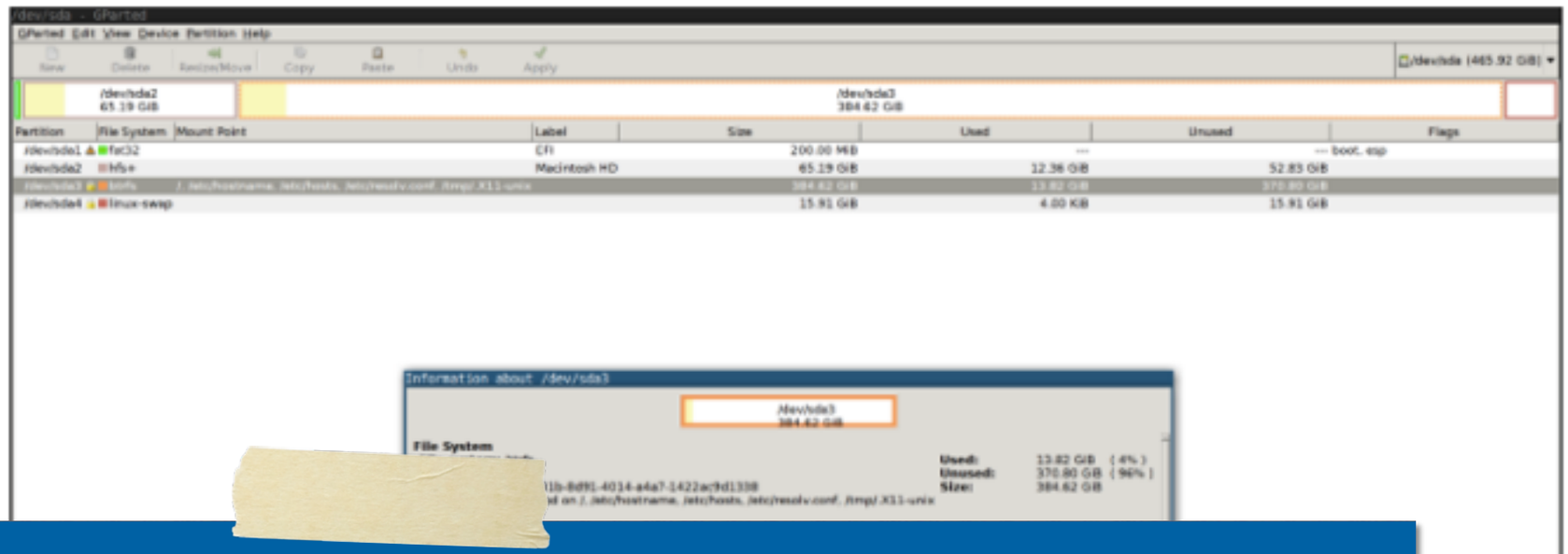
7. Gparted

Dockerfile

Partition your device in a container.

MIND BLOWN.

```
$ docker run -it \
    -v /tmp/.X11-unix:/tmp/.X11-unix \ # mount the X11 socket
    -e DISPLAY=unix$DISPLAY \ # pass the display
    --device /dev/sda:/dev/sda \ # mount the device to partition
    --name gparted \
    jess/gparted
```



Quelle: Docker Containers on the Desktop
https://blog.jessfraz.com/posts/docker-containers-on-the-desktop.html

Docker - Diskussion

X11 is completely unsecure, the "sandboxed" app has full access to every other X11 client.

if you have docker access you have root access […]
`docker run -v /:/tmp ubuntu rm -rf /tmp/*`
Which will remove all the files on your system.

Hacker News new | comments | show | ask | jobs | submit login

▲ Docker containers on the des

▲ alexlarsson 10 days ago

This is not sandboxing. Quite the opposite, this gives the apps root access:

First of all, X11 is completely unsecure, the "sandboxed" app has full access to every other X11 client. Thus, its very easy to write a simple X app that looks for say a terminal window and injects key events (say using Xtest extension) in it to type whatever it wants. Here is another example that sniffs the key events, including when you unlock the lock screen: https://github.com/magcius/keylog

Secondly, if you have docker access you have root access. You can easily run something like:

docker run -v /:/tmp ubuntu rm -rf /tmp/*

Which will remove all the files on your system.

reply

▲ jdub 9 days ago

Just so everyone knows, this is Alex "I have a weird interest in application bundling systems" Larsson, who is doing some badass bleeding edge work on full on sandboxed desktop applications on Linux. :-)

http://blogs.gnome.org/alexl/2015/02/17/first-fully-sandboxe...

http://www.youtube.com/watch?v=t-2a_XYJPEY

Like Ron Burgundy, he's... "kind of a big deal".

(Suffer the compliments, Alex.)

reply

ot part of their
ry to run,

Quelle: Docker containers on the desktop - Discussion
https://news.ycombinator.com/item?id=9086751

science + computing
| an atos company

"Without user namespaces (CLONE_NEWUSER), which Docker currently doesn't use, uid 0 inside a container is the same thing as uid 0 outside it.

If you let Docker run apps as root, which seems to be not uncommon, then it is, in a strong sense, the same as the root user outside the container.

That's why Jessie's gparted process can partition her disk: as long as it can get at the device node, it has full permissions on it.

Quelle: Docker containers on the desktop - Discussion
https://news.ycombinator.com/item?id=9088169

science + computing

an atos company

NFS *anyone?*

science + computing

an atos company

Docker *zähmen*

science + computing

an atos company

Feingranularer Zugriff

science + computing

an atos company

Wrapper

science + computing

an atos company

Application vs System Container

Docker Image Insecurity

Jonathan Rudenberg

is a cofounder of Flynn and an architect of the Tent Protocol.

GitHub
Tent
Twitter
Email
Feed

Docker Image Insecurity

December 23, 2014

Recently while downloading an "official" container image with Docker I saw this line:

```
ubuntu:14.04: The image you are pulling has been verified
```

I assumed this referenced Docker's heavily promoted image signing system and didn't investigate further at the time. Later, while researching the cryptographic digest system that Docker tries to secure images with, I had the opportunity to explore further. What I found was a total systemic failure of all logic related to image security.

Docker's report that a downloaded image is "verified" is based solely on the presence of a signed manifest, and Docker never verifies the image checksum from the manifest. An attacker could provide any image alongside a signed manifest. This opens the door to a number of serious vulnerabilities.

Images are downloaded from an HTTPS server and go through an insecure streaming processing pipeline in the Docker daemon:

```
[decompress] -> [tarsum] -> [unpack]
```

This pipeline is performant but completely insecure. Untrusted input should not be processed before verifying its signature. Unfortunately Docker

ree times before checksum verification is supposed to

Quelle: Docker Image Insecurity
https://titanous.com/posts/docker-insecurity

science + computing
an atos company

Teil IV:
Zusammenfassung

Wa(h)lwerbung?

Quelle:
http://cdn2.spiegel.de/images/image-806145-galleryV9-ygfz.jpg

Zusammenfassung

science + computing
an atos company

Virtualisierung 2.0? Nö ;)

bessere Performance als *klassische Virtualisierung* im HPC

Sicherheit

muss gewährleistet sein

science + computing

| an atos company

Vielen Dank für Ihre Aufmerksamkeit.

Holger Gantikow

science + computing ag
www.science-computing.de

Telefon: 07071 9457 - 503
E-Mail: h.gantikow@science-computing.de

Frage?
Antwort!

science + computing

| an atos company

http://www.science-computing.de
https://www.science-computing.de/jobs

science + computing

an atos company

Quellen

aufgerufen
14.03.2015

Quellen

science + computing
an atos company

Docker ALL THE THINGS

- http://cdn.meme.am/instances/500x/59600465.jpg

Docker Logo

- http://blog.docker.com/wp-content/uploads/2013/06/Docker-logo-011.png

IBM Docker Paper

- Google: *ibm docker paper* oder:
- http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf

Ian Jackson - Surviving the Zombie Apocalyse

- http://xenbits.xen.org/people/iwj/2015/fosdem-security/

Docker Containers on the Desktop

- https://blog.jessfraz.com/posts/docker-containers-on-the-desktop.html

Quellen

science + computing
an atos company

Docker containers on the desktop - Discussion

- https://news.ycombinator.com/item?id=9086751

Docker containers on the desktop - Discussion

- https://news.ycombinator.com/item?id=9088169

A Real Life White Whale that Destroyed Over 20 Whaling Ships and Survived Encounters with Another 80

- http://www.todayifoundout.com/index.php/2011/12/a-real-life-white-whale-that-destroyed-over-20-whaling-ships-and-survived-encounters-with-another-80/

Docker Image Insecurity

- https://titanous.com/posts/docker-insecurity

Wa(h)lplakat

- http://cdn2.spiegel.de/images/image-806145-galleryV9-ygfz.jpg