MySQL as a Service: LXC Applicationcontainer

Erkan Yanar

Infrastructure Design
of
Architecture & Standards

6. März 2012

Agenda

Es geht um Virtualisierung!

1. MySQL as a Service *mit MySQL Bordmitteln*
2. Linux: Cgroups & LXC
3. MySQL as a Service *mit LXC Applikationscontainer*
4. Cgroups demystified
5. Application as a Service *mit LXC Applikationscontainer*

Einleitung

Die Welt von Alles-as-a-Service:

Am Beispiel von MySQL wird eine SaaS/RDBaaS/DBoD Lösung vorgestellt.
Die bekannteste MySQL SaaS/RDBaaS/DBoD Lösung ist wohl Amazon RDS.

Einleitung

Die Welt von Alles-as-a-Service:

Am Beispiel von MySQL wird eine SaaS/RDBaaS/DBoD Lösung vorgestellt.
Die bekannteste MySQL SaaS/RDBaaS/DBoD Lösung ist wohl Amazon RDS.

Wofür?

Einleitung

Die Welt von Alles-as-a-Service:

Am Beispiel von MySQL wird eine SaaS/RDBaaS/DBoD Lösung vorgestellt.
Die bekannteste MySQL SaaS/RDBaaS/DBoD Lösung ist wohl Amazon RDS.

- Testumgebung
- Upgrades
- Prototyping
- Betrieb

Einleitung

Die Welt von Alles-as-a-Service:

Am Beispiel von MySQL wird eine SaaS/RDBaaS/DBoD Lösung vorgestellt.
Die bekannteste MySQL SaaS/RDBaaS/DBoD Lösung ist wohl Amazon RDS.

- Testumgebung
- Upgrades
- Prototyping
- Betrieb?

Einleitung

Agenda *Erkan Yanar*

1. RDBaaS mit MySQL Bordmitteln
2. Rettungsanker Virtualisierung?
3. RDBaas (DBoD) mit LXC Applikationscontainer

Multi-Schema

Aufgabe:

Alle Datenbanken der Projekte werden von einer Instanz bedient

Multi-Schema

Aufgabe:

Alle Datenbanken der Projekte werden von einer Instanz bedient

Neuer Kunde:

```
CREATE SCHEMA schema;
GRANT ALL ON schema.* TO kunde@host IDENTIFIED BY password;
```

Übersicht

	Multi Schema
Easy of Use	X
Ressourcenzuweisung zwischen Schemata	-
Applikationssp. Conf	-
Verschiedene MySQL-Version	-
Sep. PITR, Upgrade	-
Serverauslastung	-

Multi Instance

- Verwaltung separater Instanzen
- `mysqld_multi`
- Sep. User pro Instanz
- Sep. Port pro Instanz

my.cnf

```
[mysqld_multi]
mysqld = /some/mysqld_safe
mysqladmin = /some/mysqladmin
user = multi_admin
password = multipass
[mysqld2]
socket = /tmp/mysql.sock2
port = 3307
pid-file = /some/host.pid2
datadir = /some/var2
user = john
[mysqld3]
socket = /tmp/mysql.sock3
port = 3308
pid-file = /some/host.pid3
...
```

Übersicht

	Multi Schema	Multi Instanz
Easy of Use	X	-
Ressourcenzuweisung zwischen Schemata	-	X
Systemressourcen zwischen Instanzen	(-)	-
Applikationssp. Conf	-	X
Verschiedene MySQL-Version	-	(X)
Sep. PITR, Upgrade	-	X
Default Port	X	-
Serverauslastung	-	X
Overhead vieler Instanzen	-	X

Agent Virtualisierung

Wobei kann Virtualisierung helfen?!
Zumindest bei:

- Ressourcenzuweisung
- Isolation
- Port
- ...

Übersicht

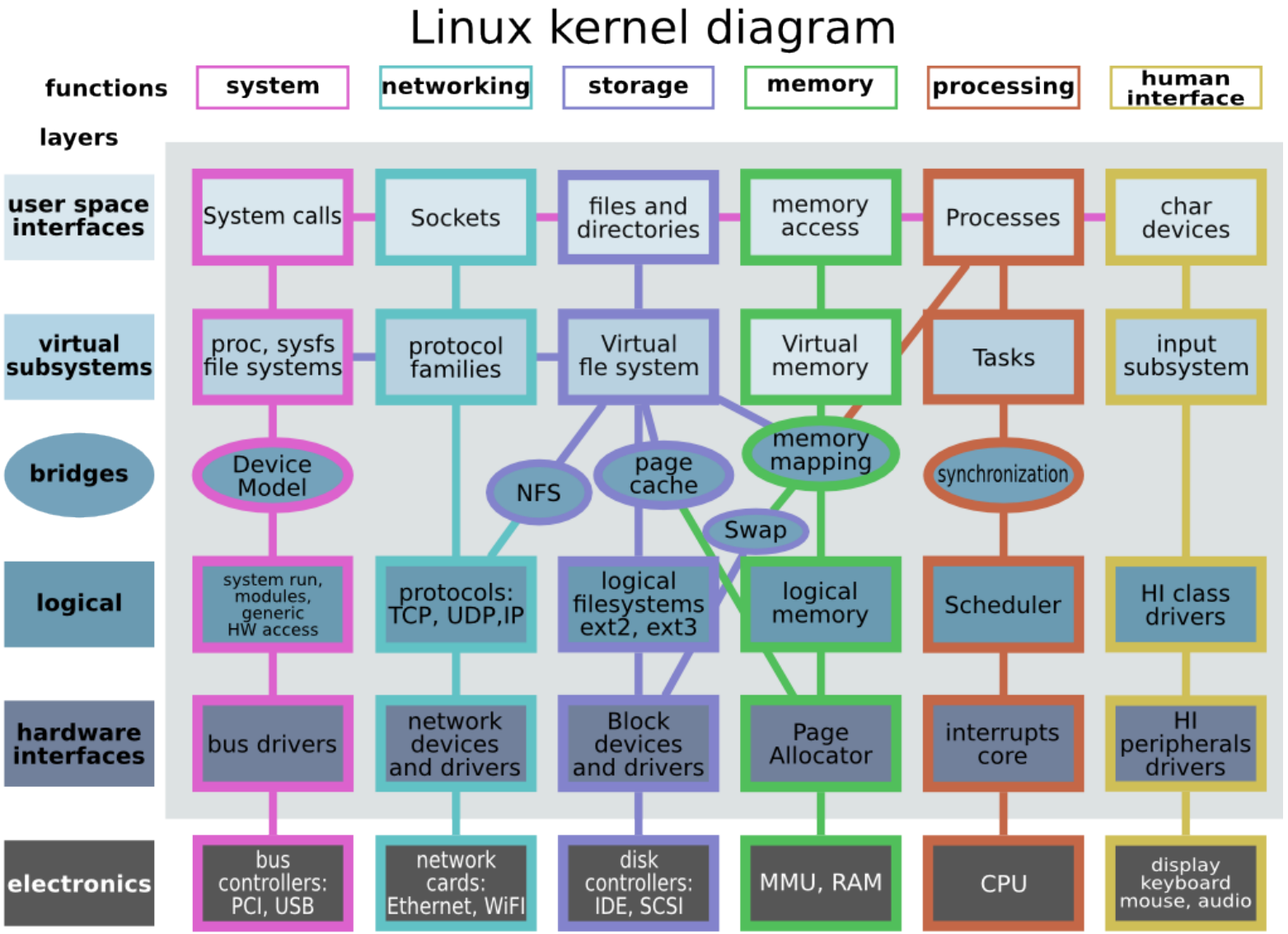
	Multi Schema	Multi Instanz	Virtualisiert
Easy of Use	X	-	(X)
Ressourcenzuweisung zwischen Schemata	-	X	X
Systemressourcen zwischen Instanzen	(-)	-	X
Applikationssp. Conf	-	X	X
Versch. MySQL-Ver.	-	(X)	X
Sep. PITR, Upgrade	-	X	X
Default Port	X	-	X
Serverauslastung	-	X	X
Instanzen Overhead	-	X	X
Accounting	-	-	X

Container aka OS-Virtualisierung und die Anderen unter Linux

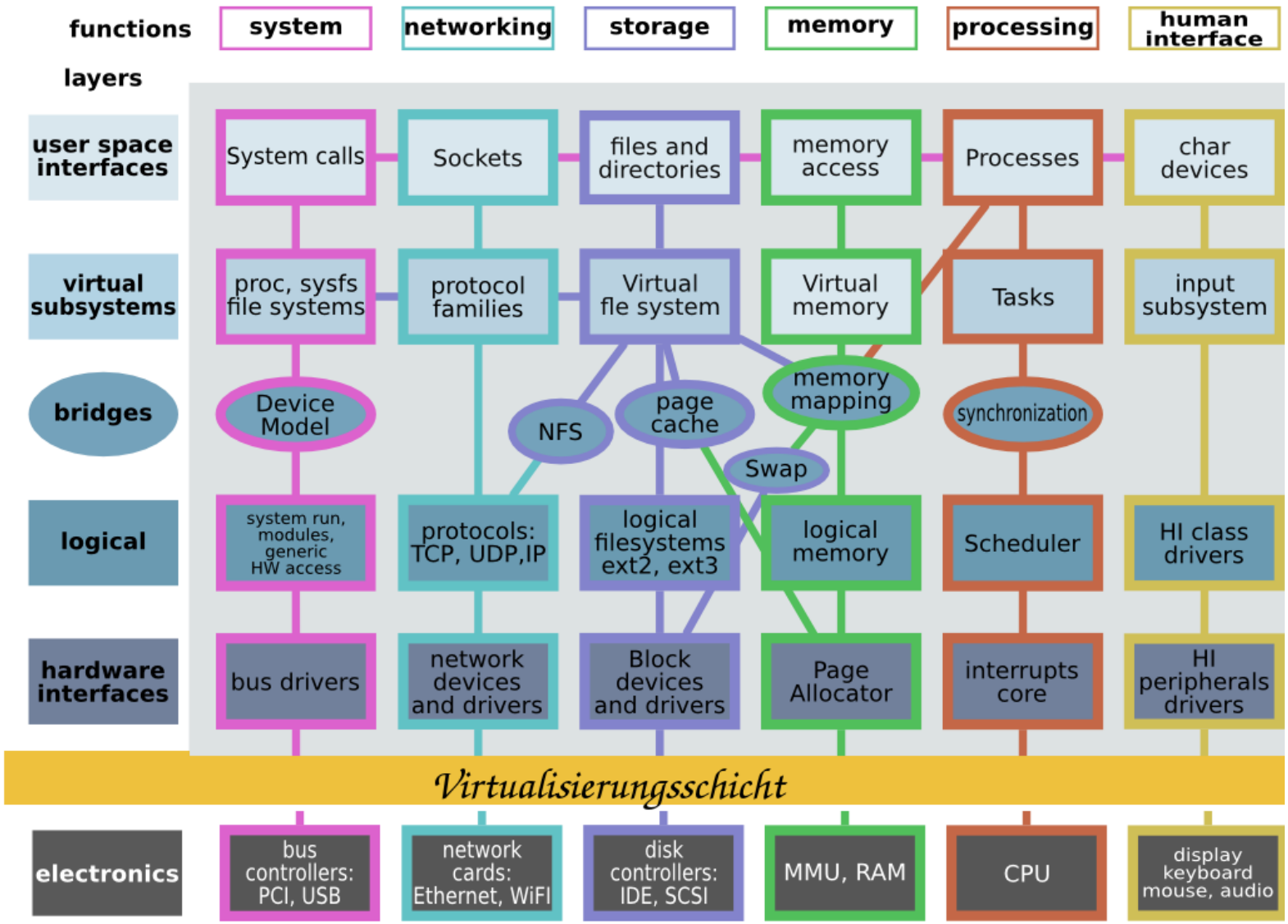
Möglichkeiten:

LXC

Linux

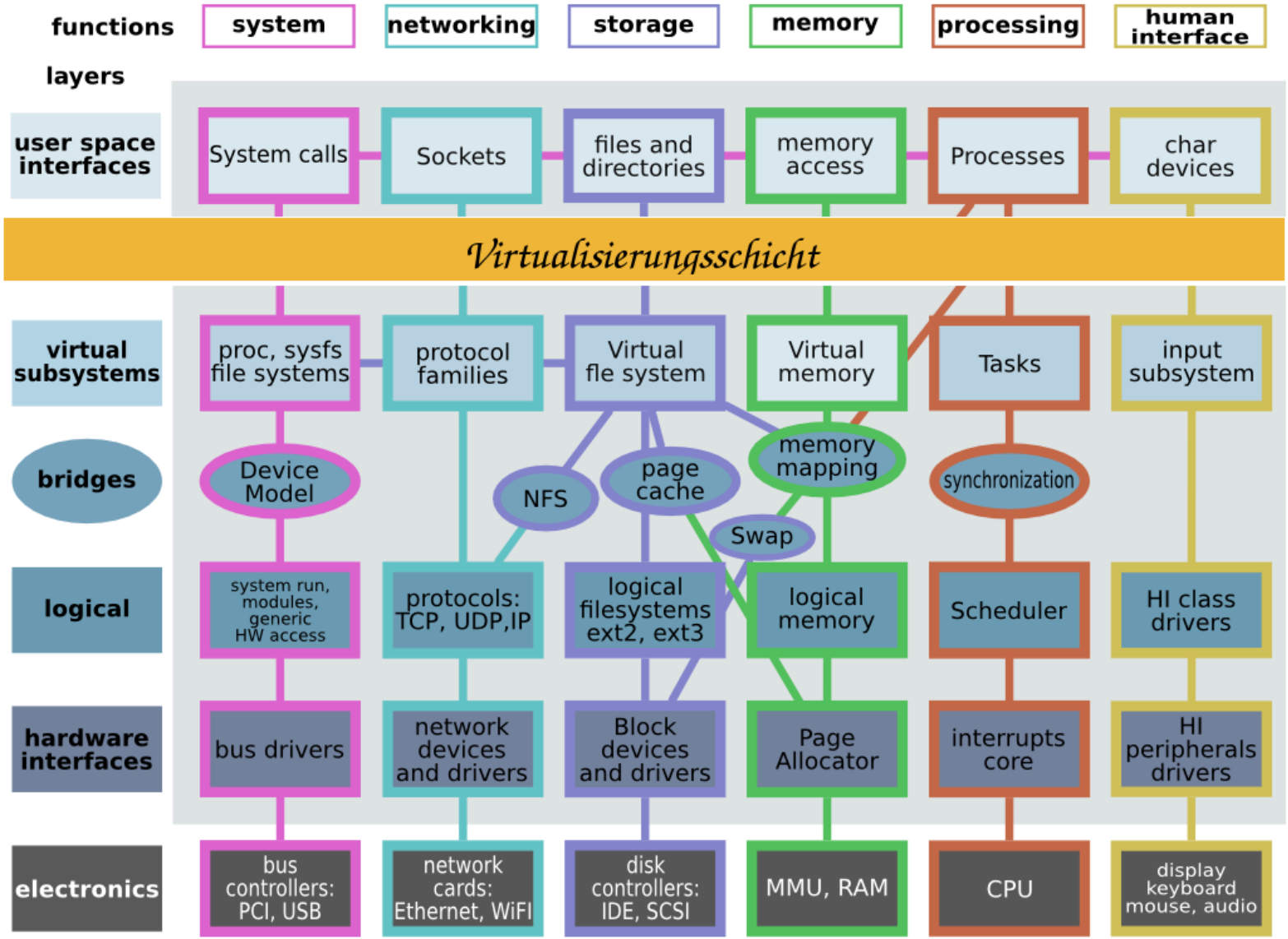

Linux kernel diagram
functions
system
networking
storage
memory
processing
human interface
layers
user space interfaces
virtual subsystems
bridges
logical
hardware interfaces
electronics
System calls
Sockets
files and directories
memory access
Processes
char devices
proc, sysfs file systems
protocol families
Virtual fle system
Virtual memory
Tasks
input subsystem
Device Model
NFS
page cache
memory mapping
synchronization
Swap
system run, modules, generic HW access
protocols: TCP, UDP,IP
logical filesystems ext2, ext3
logical memory
Scheduler
HI class drivers
bus drivers
network devices and drivers
Block devices and drivers
Page Allocator
interrupts core
HI peripherals drivers
bus controllers: PCI, USB
network cards: Ethernet, WiFI
disk controllers: IDE, SCSI
MMU, RAM
CPU
display keyboard mouse, audio
© 2007-2009 Constantine Shulyupin http://www.MakeLinux.net/kernel/diagram

KVM etc.



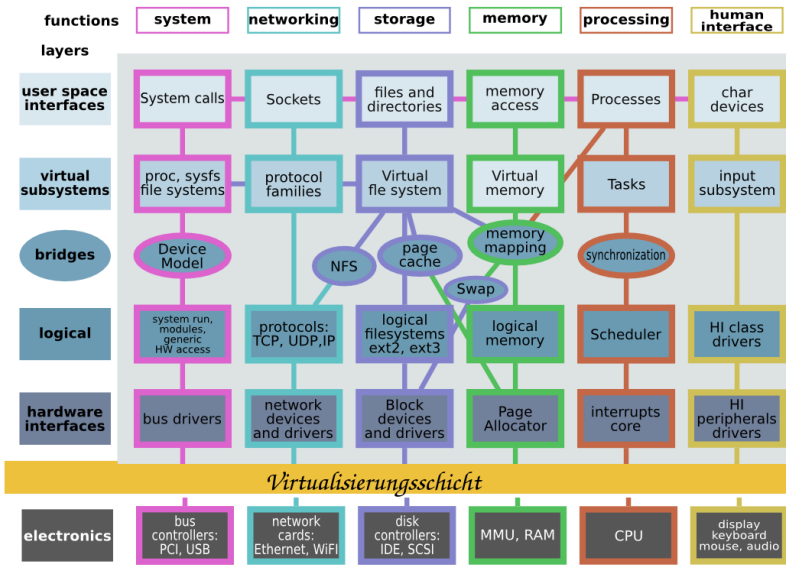
© 2007-2009 Constantine Shulyupin http://www.MakeLinux.net/kernel/diagram

Container

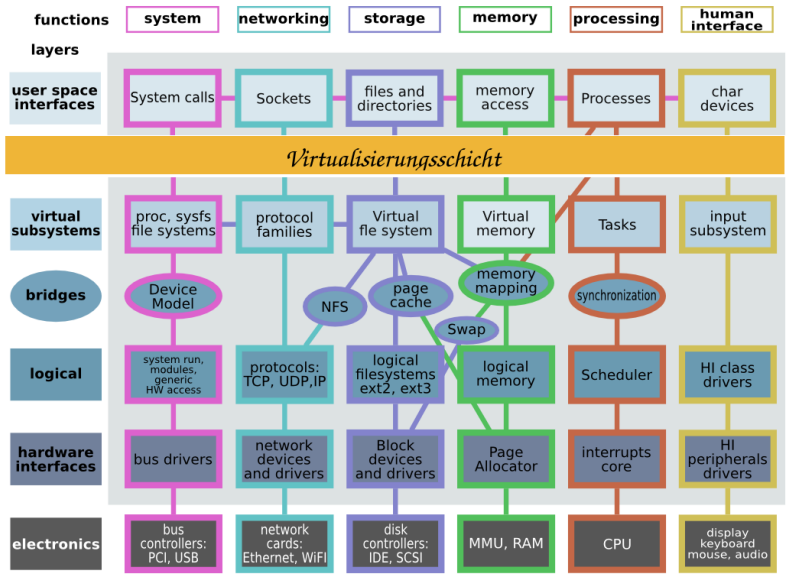


© 2007-2009 Constantine Shulyupin http://www.MakeLinux.net/kernel/diagram

Two Worlds



© 2007-2009 Constantine Shulyupin http://www.MakeLinux.net/kernel/diagram



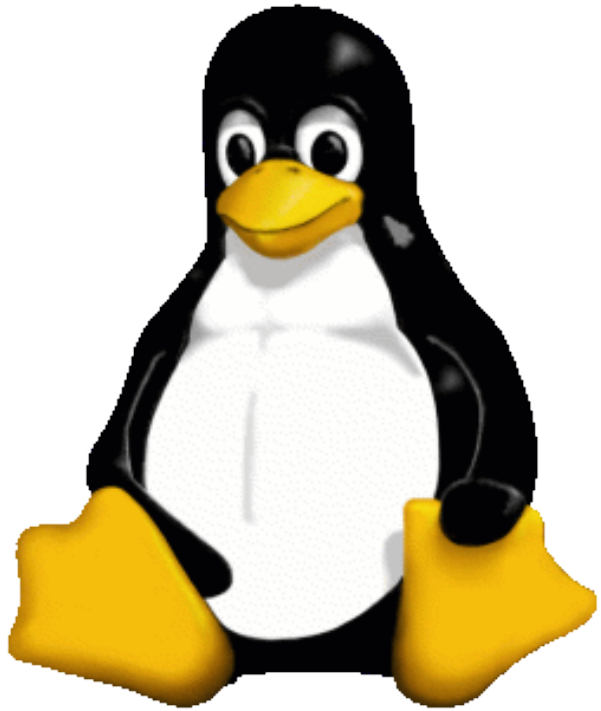
© 2007-2009 Constantine Shulyupin http://www.MakeLinux.net/kernel/diagram

Hardware Virtualisierung kann mehr!

	Hardware Virt.	Betriebssystem Virt.
	KVM, Xen, VMWare	LXC, OpenVZ
Separates OS	X	-
Seperater Kernel	X	-
Andere Hardwarearchitekturen	qemu	-
Server Auslastung	-	X

Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler

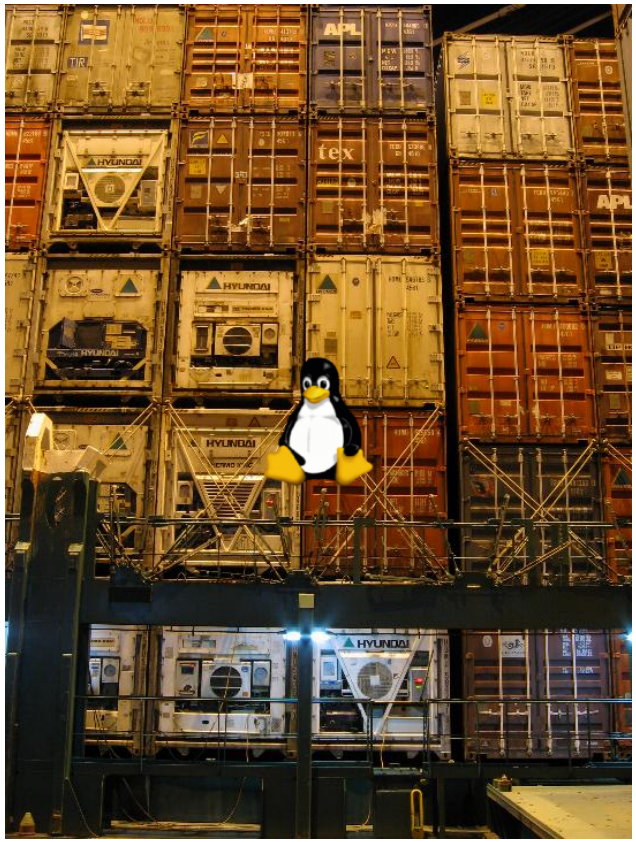


Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler

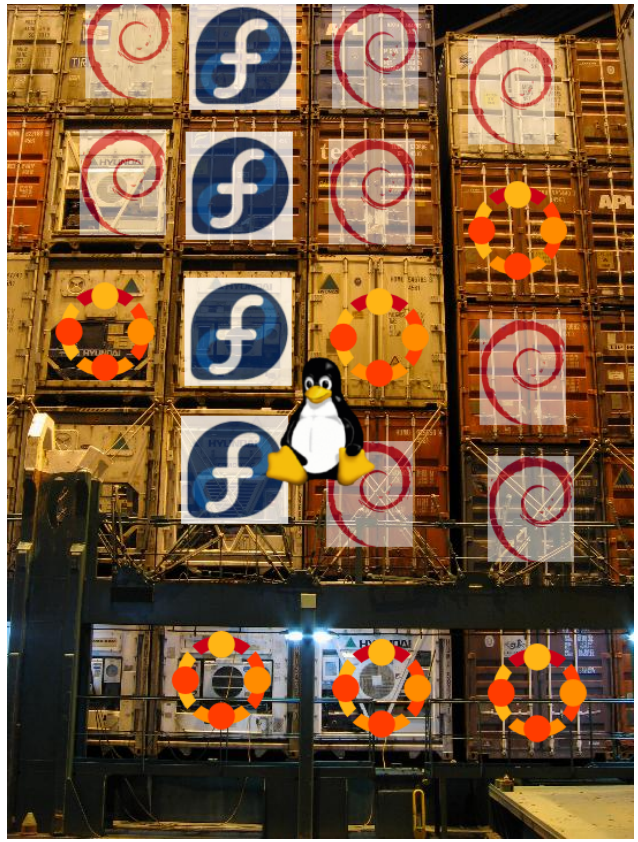
Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler



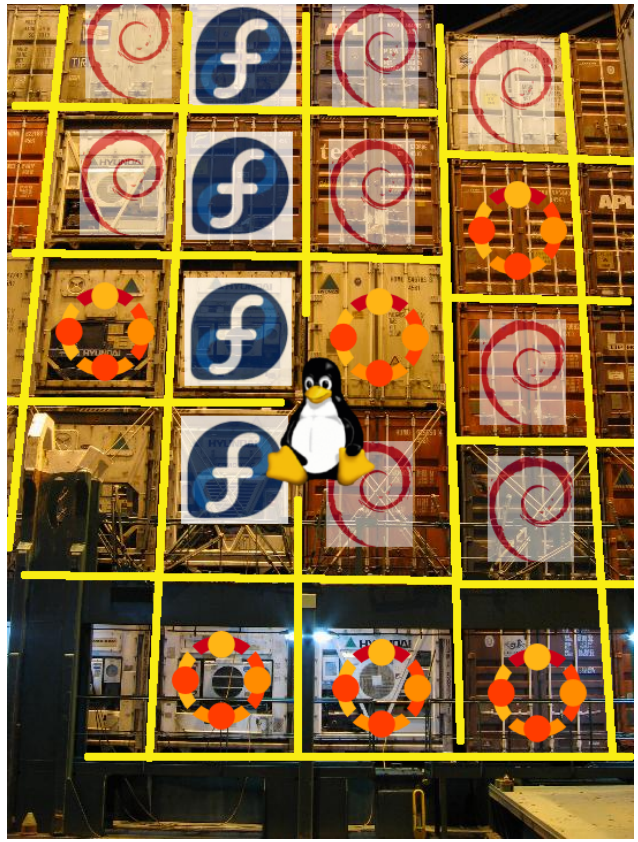
Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler



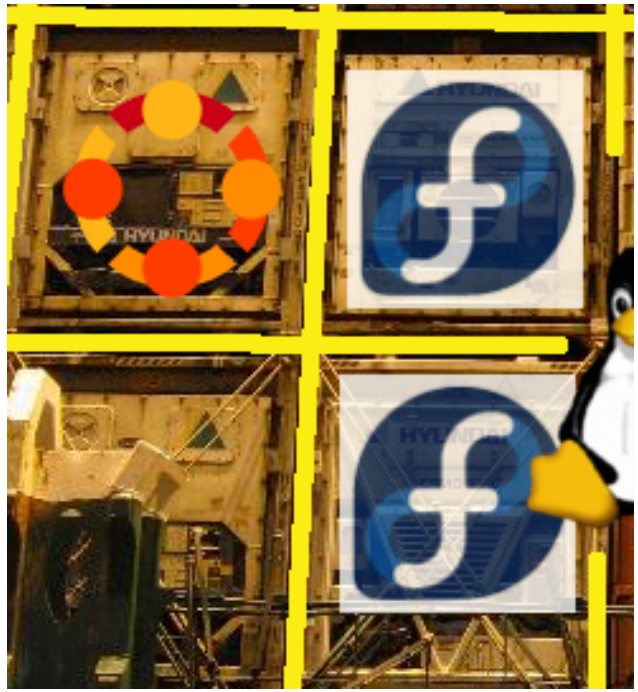
Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler



Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler



Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- **Prozessvirtualisierung**
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler

Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler

Container sind:

- Virtualisierung im OS
- Container / Verzeichnisse
- Hostkernel übernimmt die Verwaltung
- → Nur ein OS-Typ
- Dünne Virtualisierungsschicht
- Prozessvirtualisierung
- Dynamische Zuweisung von Ressourcen
- CPU und I/O Scheduler

Überblick

Look@Container ala LXC

Überblick

Look@Container ala LXC

1. cgroups: Ressourcenmanagement
2. LXC: (Applikations)Container on top
3. OpenVZ vs. LXC kurzer Überblick

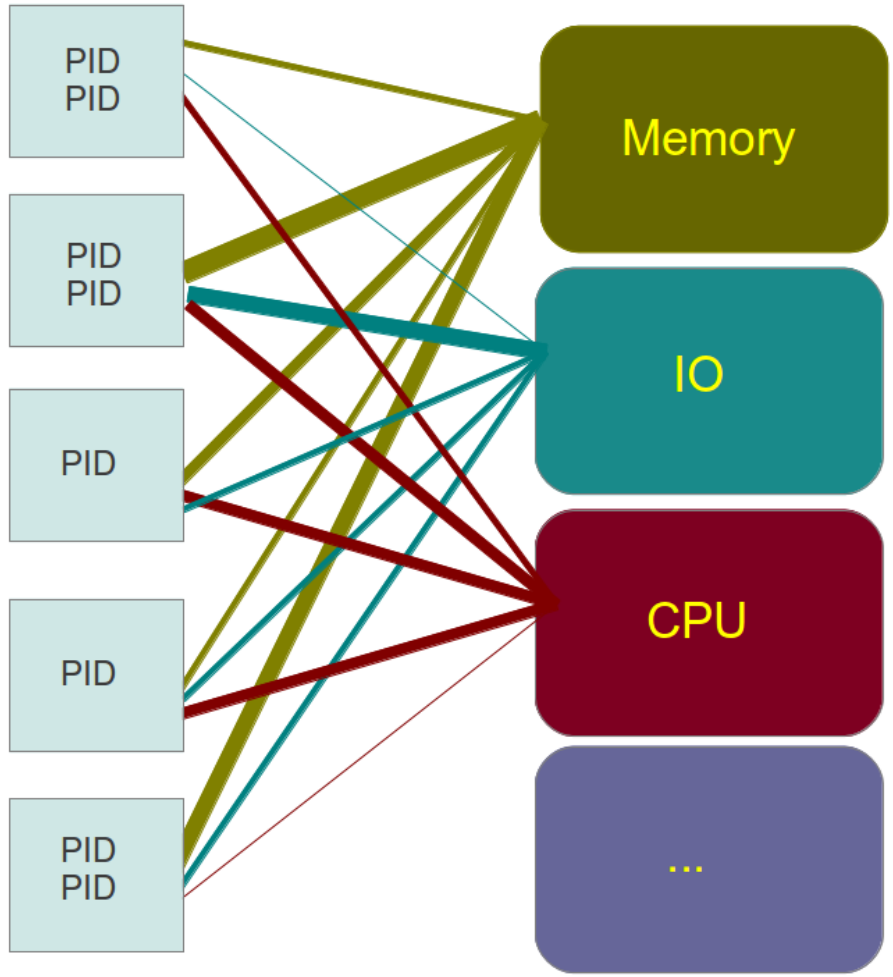
Ressourcenmanagement mit cgroups

Control Groups

- Gruppieren von Prozessen
- Gemeinsame Ressourcen
- Childs bleiben in der Gruppe

Control Groups

- VFS
- $\geq$ Kernel 2.6.24
- unabhängig von LXC
- mount:
  `cgroup /cgroups cgroup defaults 0 0`



Subsysteme/Controlgroups

```
cat /proc/cgroups
 #subsys_name hierarchy num_cgroups enabled
cpuset 1 4 1
cpu 2 4 1                                                    CPU
cpuacct 3 4 1
memory 4 4 1
devices 5 4 1
freezer 6 4 1
net_cls 7 1 1
blkio 8 4 1
```

Subsysteme/Controlgroups

```
cat /proc/cgroups
 #subsys_name hierarchy num_cgroups enabled
cpuset 1 4 1
cpu 2 4 1
cpuacct 3 4 1
memory 4 4 1                                    Speicher
devices 5 4 1
freezer 6 4 1
net_cls 7 1 1
blkio 8 4 1
```

Subsysteme/Controlgroups

```
cat /proc/cgroups
 #subsys_name hierarchy num_cgroups enabled
cpuset 1 4 1
cpu 2 4 1
cpuacct 3 4 1
memory 4 4 1
devices 5 4 1                                    mknod
freezer 6 4 1
net_cls 7 1 1
blkio 8 4 1
```

Subsysteme/Controlgroups

```
cat /proc/cgroups
 #subsys_name hierarchy num_cgroups enabled
cpuset 1 4 1
cpu 2 4 1
cpuacct 3 4 1
memory 4 4 1
devices 5 4 1
freezer 6 4 1                                   FROZEN/THAWED
net_cls 7 1 1
blkio 8 4 1
```

Subsysteme/Controlgroups

```
cat /proc/cgroups
 #subsys_name hierarchy num_cgroups enabled
cpuset 1 4 1
cpu 2 4 1
cpuacct 3 4 1
memory 4 4 1
devices 5 4 1
freezer 6 4 1
net_cls 7 1 1
blkio 8 4 1
```

Markieren

Subsysteme/Controlgroups

```
cat /proc/cgroups
 #subsys_name hierarchy num_cgroups enabled
cpuset 1 4 1
cpu 2 4 1
cpuacct 3 4 1
memory 4 4 1
devices 5 4 1
freezer 6 4 1
net_cls 7 1 1
blkio 8 4 1
```

CFQ

```
  WOHL FÜR DEN TECHTALK WEGNEHMEN
# ls /cgroups #2.6.38 (Auszug) --- To be deleted --
blkio.throttle.read_bps_device  cpuset.memory_pressure
blkio.throttle.read_iops_device cpuset.memory_pressure_enabled
blkio.throttle.write_bps_device cpuset.mems
blkio.throttle.write_iops_device cpuset.sched_load_balance
blkio.weight                    cpuset.sched_relax_domain_level
cgroup.clone_children           cpu.shares
cgroup.procs                    devices.allow
cpuacct.stat                    memory.limit_in_bytes
cpuacct.usage                   memory.memsw.limit_in_bytes
cpuacct.usage_percpu            memory.oom_control
cpuset.cpu_exclusive            memory.stat
cpuset.cpus                     memory.swappiness
cpuset.mem_exclusive            memory.usage_in_bytes
cpuset.mem_hardwall             net_cls.classid
cpuset.memory_migrate           tasks
```

LXC

LXC
LinuXContainer

LXC

LinuXContainer

Ein chroot macht auf virtuell

LXC
LinuXContainer

Grundprinzipien und Stolpersteine

LinuXContainer

LXC: better cgroups?

- Spätestens seit 2.6.26 im Kernel (Network-Namespace)
- Erzeugt mit Hilfe von Namespaces Container.
- cgroups dienen zur Resourcenverwaltung.
- LXC übernimmt die Verwaltung der Prozessgruppen
- Modulares Design!

Namespaces Die Seele der Virtualisierung

utsname	hostname	[Modular]
Pid	private PIDs	[Automatisch]
User	private UIDs	[Automatisch]
Network	privates Interface	[Modular]
IPC	privates IPC	[Automatisch]

LXC virtualisiert chroot() Umgebungen

Konfiguration

/var/lib/lxc/$CONTAINER Konfigurationsverzeichnis des Containers

/var/lib/lxc/$CONTAINER/config Konfigurationsdatei des Containers

Wo ist das chroot Verzeichnis?

(lxc.)rootfs Filesystem des Containers

LXC startet dieses „System“

LXC-Tools

Auszug

lxc-create Erstellt einen Container

lxc-destroy Löscht rootfs und das Configverzeichnis

LXC-Tools

Auszug

lxc-create Erstellt einen Container

lxc-destroy Löscht rootfs und das Configverzeichnis

Unnötiges Commando?

lxc-create -n name [-f config_file] [-t template]

- Schreibe mit config_file nach /var/lib/lxc/$name/config
- Nutze Template (Skript) zum Erstellen eines Containers
- Mehr zu Templates? Probleme?

Container Filesystem

Erstelle einen Container:

- debootstrap, febootstrap ..
- udevd ausschalten
  Daher für i.e. tty, null, zero etc. mknod nutzen
- hwclock entfernen
- ..

lxc-debian

```
/usr/sbin/update-rc.d -f checkroot.sh remove
/usr/sbin/update-rc.d -f umountfs remove
/usr/sbin/update-rc.d -f hwclock.sh remove
/usr/sbin/update-rc.d -f hwclockfirst.sh remove
/usr/sbin/update-rc.d -f module-init-tools remove
```

`/usr/lib/lxc/templates`

Container Filesystem

Erstelle einen Container:

- debootstrap, febootstrap ..
- udevd ausschalten
  Daher für i.e. tty, null, zero etc. mknod nutzen
- hwclock entfernen
- ..

lxc-debian

```
/usr/sbin/update-rc.d -f checkroot.sh remove
/usr/sbin/update-rc.d -f umountfs remove
/usr/sbin/update-rc.d -f hwclock.sh remove
/usr/sbin/update-rc.d -f hwclockfirst.sh remove
/usr/sbin/update-rc.d -f module-init-tools remove
```

`/usr/lib/lxc/templates`

Container zeigen

Man erstelle eine Konfigurationsdatei

Konfiguration

lxc.rootfs	chroot
lxc.mount.entry	Ein Mountpunkt im fstab-Format
lxc.mount	Pfad zu einem File mit Mountp. im fstab Format
lxc.tty	Virtuelle Consolen: lxc-console
lxc.pts	Pseudo ttys
lxc.cap.drop	man capabilities

```
lxc.tty = 4
lxc.rootfs = /lxc/debian/rootfs
lxc.mount = /lxc/debian/fstab
```

Network

lxc.network.type

Kein Eintrag	Interfaceeinstellungen des Hosts
empty	loopback
veth	Virtual Ethernet (bridge)
macvlan	MAC-Address based Vlan
phys	physisches Interface

```
lxc.network.type = veth
lxc.network.flags= up
lxc.network.link = br0
lxc.network.ipv4 =
192.168.1.69/24
lxc.network.name = eth0
lxc.network.veth.pair =
this-veth
```

/var/lib/lxc/$CONTAINER/config

```
lxc.utsname = zeig
lxc.tty = 4
lxc.pts = 1024

#Vom Host gemounted
lxc.mount = /lxc/debian/fstab

#rootfs
lxc.rootfs = /lxc/debian/rootfs

#Netzwerk:
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = br0
lxc.network.hwaddr = 08:00:12:34:56:78
lxc.network.ipv4 = 192.168.1.69/24
lxc.network.name = eth0

...
```

LXC-*Tools*

Auszug:

lxc-ls	Zeigt alle konfigurierten und laufenden Container
lxc-start/stop	Starten/Stoppen eines Containers
lxc-ps	Wrapper um ps mit Containername
lxc-console	Konsolenverbindung zum Container
lxc-execute	Startet einen Prozess im ContainerEnvironment

LXC-*Tools*

Auszug:

lxc-ls	Zeigt alle konfigurierten und laufenden Container
lxc-start/stop	Starten/Stoppen eines Containers
lxc-ps	Wrapper um ps mit Containername
lxc-console	Konsolenverbindung zum Container
lxc-execute	Startet einen Prozess im ContainerEnvironment
	Applikationscontainer

Show me that stuff!

- Container start
- Host Zugriff
- Privater Prozessspace
- Applikationskontainer mit Ressourcemanagement

Security

Capabilities

remind the fstab

lxc.cap.drop

root im Container zu mächtig

- module sys_module
- mount sys_admin

echo b > /proc/sysrq-trigger

- SELinux
- Smack

lxc.mount.entry=proc $lxc.rootfs/proc proc nodev,noexec,nosuid,ro 0 0

Applikationscontainer

lxc-execute

- Schlüssel zum Applikationscontainer
- braucht *kein* lxc.rootfs
- Config kann mehrmals verwendet werden (– – *name* , -f)
- Modularität Ausnutzen

OpenVZ vs. LXC

Topic	LXC	OpenVZ
Kernelintegration	X	-
Livemigration	-	X
Host Konfigtools (vzctl)	-	X
Sicheres Netz (venet)	-	X
Sichere Container	-	X
Applikationscontainer	X	-
diskspace	-	X
Cgroups	X	-
Quota	-	X
Distro-Support	X	-
Produktionsreif	(-)	X
libvirt-Integration	X	(X)
Modular	X	-

DBoD mit LXC

DBoD mit LXC

Grundidee

- Out of the Box-Technik
- Zukunftstechnologie
- Keine Verschwendung von Rechnerressourcen
- Keine weiteren Lizenzkosten

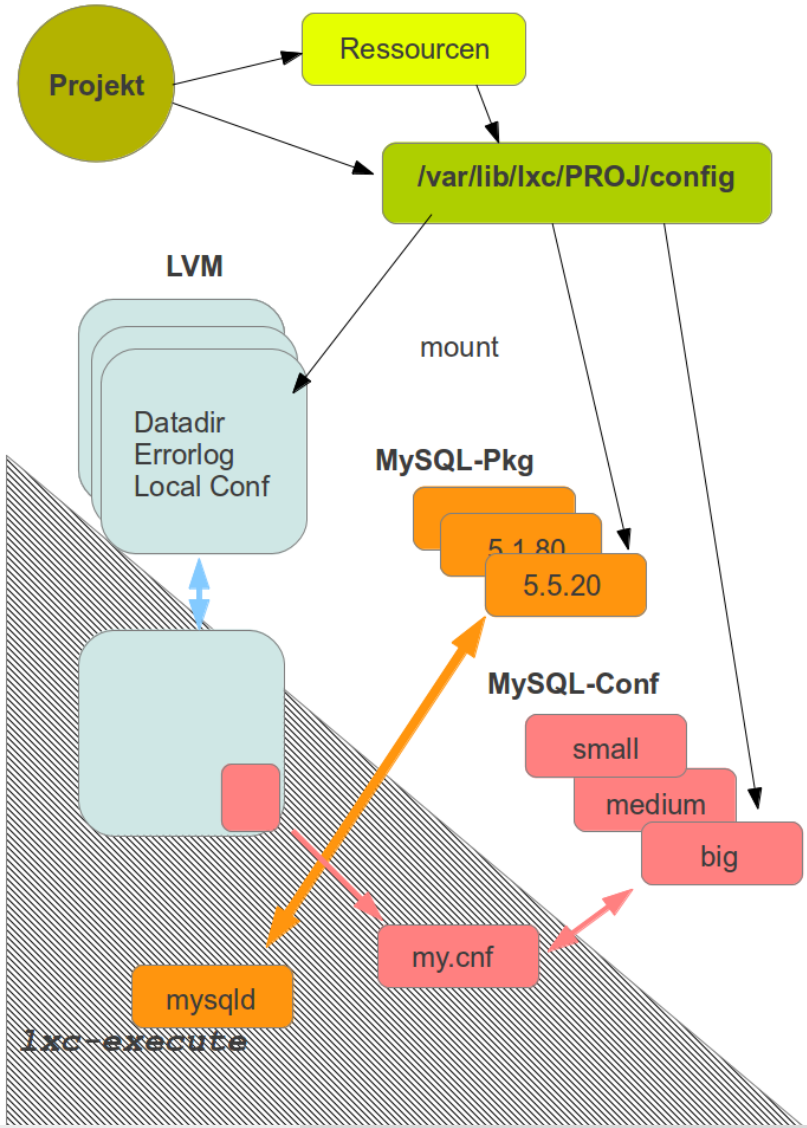
Applikationscontainer mit lxc-execute

lxc-execute

- Schlüssel zum Applikationscontainer
- braucht *kein* lxc.rootfs
- Config kann mehrmals verwendet werden ($--$ *name* , -f)
- Modularität Ausnutzen

Realisierung

- lxc-execute
- LVM
  Für alle veränderlichen Daten
  - datadir
  - tmpdir
  - Errorlog
  - Application specific Config
- Multi-Package
  Auswahl an MySQL-Versionen
- Schema dbod
  Datenbankmanagement



```
lxc.utsname = dbod_8012
#lxc.tty = 1

# Bind-Mounts
lxc.mount.entry =
/opt/app/dbod/mysql//5.5.13/conf/medium /opt/app/mysql/conf none bind 0 0

lxc.mount.entry =
/opt/app/dbod/mysql//5.5.13/pkg/ /opt/app/mysql/product/mysql none bind 0 0

lxc.mount.entry = /data/dbod/dbod_8012 /data none bind 0 0


# Ressourcen
lxc.cgroup.cpu.shares = 2046
lxc.cgroup.memory.limit_in_bytes = 1G
lxc.cgroup.memory.memsw.limit_in_bytes = 1G
```

Tools

- dbod_mysql_create.pl
- dbod_mysql_list.pl
- dbod_mysql_start.pl
- dbod_mysql_stop.pl
- dbod_mysql_upgrade.pl

```
# dbod_mysql_create.pl --port 6005 --number 6005 --mysqlversion 5.5.20
                       -t small  --gb 5 --database kunde
OptionCheck is not finished!!!
Logical volume "dbod_6005" created
passwort for admin on dbod_6005 is yuzPgTRZ92k/U
#
```

```
# dbod_mysql_start.pl -n 6005 && dbod_mysql_list.pl
 Instance            MySQL         Port     Status      Memory    cur. Mem max
dbod_5101              5.1.58      5101    -------           -             -
dbod_5102              5.1.58      5102    -------           -             -
dbod_5103              5.1.58      5103    -------           -             -
dbod_5104              5.1.58      5104    -------           -             -
dbod_5301        Maria-5.3.0       5301    RUNNING      524288        156032
dbod_5305        Maria-5.3.0       5305    -------           -             -
dbod_5501              5.5.13      5501    RUNNING     1048576        462996
dbod_5503              5.5.13      5503    -------           -             -
dbod_6000              5.5.20      6000    -------           -             -
dbod_6001              5.5.20      6001    -------           -             -
dbod_6002              5.5.20      6002    -------           -             -
dbod_6003              5.5.20      6003    -------           -             -
dbod_6004              5.5.20      6004    -------           -             -
dbod_6005              5.5.20      6005    RUNNING      524288        461668
dbod_8000        Spider5.5.14      8000    -------           -             -
dbod_8001        Spider5.5.14      8001    RUNNING     1048576        468008
dbod_8010              5.5.13      8010    RUNNING     1048576         77184
dbod_8011              5.5.13      8011    RUNNING     1048576        464136
dbod_8012              5.5.13      8011    -------           -             -
dbod_8020              5.5.13      8020    RUNNING     1048576        467200
```

Upcoming

- Schema dbod erweitern
- Replication
- Eigene IP
- HA
- Migration
- Btrfs
- Monitoring
- Integration in unsere Automatisierungsumgebung
- Integration in OpenStack/libvirt

Ende Gelände

Erkan Yanar @
Infrastructure Design
of
Architecture & Standards
...
linsenraum.de/erkules
www.xing.com/profile/Erkan_Yanar