LXC – Linux Containers

Virtuelle Systeme ohne Hypervisor

Virtualisierung im System

- Operating system-level virtualization
- Mehrere isolierte Systeminstallationen unter einem gemeinsamen Host
- Für alle Instanzen nur ein Kernel, aber unterschiedliche Dateisysteme, init's, Prozeduren
- Use cases: Sicherheit, Mobilität, Multihosting, cross-distro-build, Serverkonsolidierung, Prozesstracking, Ressourcenmanagement, Clusterstabilität
- Container sind grün ;-)

Implementierungen

- Linux
  - OpenVZ
  - Linux Vserver
  - LXC
  - Virtuozzo
  - FreeVPS
- Andere OS
  - Zones (Solaris)
  - Jails (FreeBSD)
  - WPARs (AIX)
  - iCore (Windows XP)

LXC - Linux Containers

- Benutzt nur Features des offiziellen Kernels
- Usertools bereits in vielen Distros vorhanden, also einfach „apt-get install lxc“ :)
- Application container: einzelne Applikationen in einem Container, z.B. bash, sshd, apache
- System container: ein System im Container
- checkpoint/restart & freeze/unfreeze

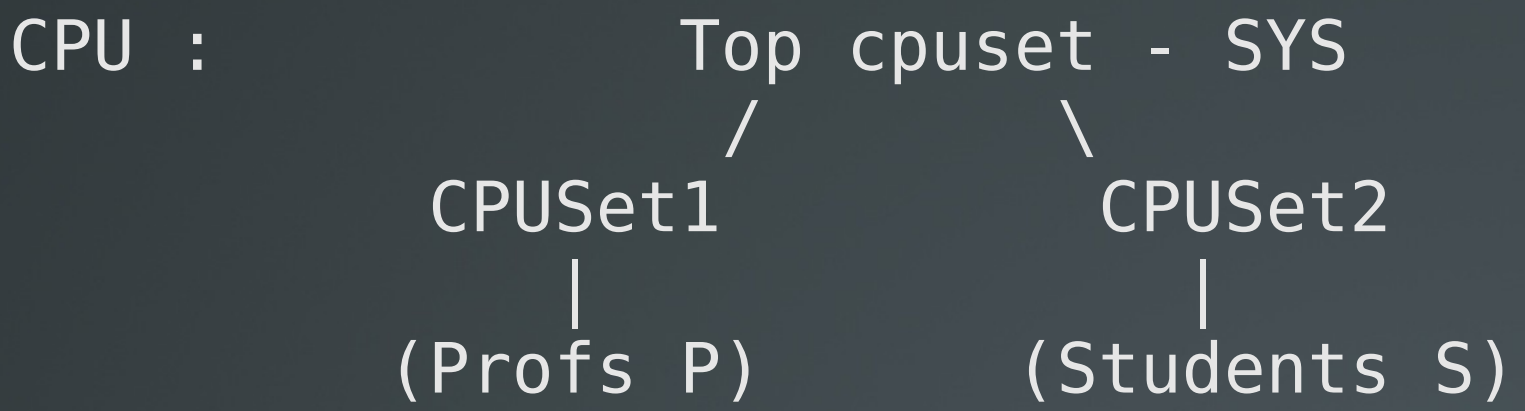
Eigenschaften eines Containers

1. **Aggregation**: Prozesse werden unter einem gemeinsamen Merkmal gruppiert und bekommen Systemressourcen zugewiesen
2. **Isolation**: Ressourcen, die einer Gruppe von Prozessen zugeteilt wurden, können nicht von anderen Prozessen genutzt werden

Aggregation – control groups

- Implementiert via cgroupfs
- mount -t cgroup cgroup /cgroup
- cgroup kann durch neue Subsysteme leicht erweitert werden
- Subsysteme: Freezer, memory, cpu accounting, cpuset, device whitelist, network bandwidth, (IO quota)

Control groups - Szenario



- Speicher: P 50%, S 30%, SYS 20%
- Platte : P 50%, S 30%, SYS 20%
- Netzwerk: WWW 20% (15%/5%), NFS 60%, 20%

Isolation - namespaces

- Systemressourcen werden über namespaces (NS) angesprochen
- Die wesentlichen NS sind im mainstream Kernel seit Version 2.6.29
- Ein NS per Kernel Subsystem: mount points, network, IPC(posix, sysV), Tasks(Prozesse, Threads), utsname, proc
- In der Queue (2.6.35): user, time, sysfs

Isolation – pid NS

```
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:00 init [3]
  248 ?        Sl     0:00 /usr/sbin/rsyslogd -c3
  262 ?        Ss     0:00 /usr/sbin/sshd
  302 ?        S      0:00 /bin/sh /usr/bin/mysqld_safe
  519 ?        Ss     0:00 /usr/sbin/cron
  533 ?        Ss     0:00 /usr/sbin/apache2 -k start
  555 console  Ss+    0:00 /sbin/getty 38400 console
  556 tty1     Ss+    0:00 /sbin/getty 38400 tty1 linux
  557 tty2     Ss+    0:00 /sbin/getty 38400 tty2 linux
  559 tty3     Ss+    0:00 /sbin/getty 38400 tty3 linux
  560 tty4     Ss+    0:00 /sbin/getty 38400 tty4 linux
 1691 ?        Ss     0:00 sshd: root@pts/0
 1693 pts/0    Ss     0:00 -bash
 1699 pts/0    R+     0:00 ps x
```

Container Tools

- lxc-create/lxc-destroy: erzeugen und löschen eines Containers
- lxc-start/lxc-stop: starten und stoppen :)
- lxc-ps: prozesse eines Containers auflisten
- lxc-netstat: jepp, netstat
- lxc-checkconfig: Systemcheck
- Alternative: libvirt/virsh

Konfigurationsdatei

```
lxc.utsname = OST01
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = virbr1
lxc.network.name = eth0
lxc.network.ipv4 = 192.168.122.22/24

lxc.mount.entry = /home /home none bind 0 0

# /dev/null and zero
lxc.cgroup.devices.allow = c 1:3 rwm
lxc.cgroup.devices.allow = c 1:5 rwm
# consoles
lxc.cgroup.devices.allow = c 5:1 rwm
lxc.cgroup.devices.allow = c 5:0 rwm
```

Erstellen mit Template

- Templates sind Scripts, die ein neues System standardisiert erstellen
- In /usr/lib/lxc/lxc/templates: lxc-busybox lxc-debian lxc-fedora lxc-sshd lxc-ubuntu
- Erstellen eines neuen Containers:

  lxc-create -n OST01 -f OST01.conf -t mydebian

Links

- LXC
  - http://lxc.sourceforge.net
  - http://lxc.teegra.net
  - http://lxc.sourceforge.net/doc/
  - https://help.ubuntu.com/community/LXC
- cgroup's
  - http://en.opensuse.org/Cgroup
  - http://fedoraproject.org/wiki/Features/ControlGroups