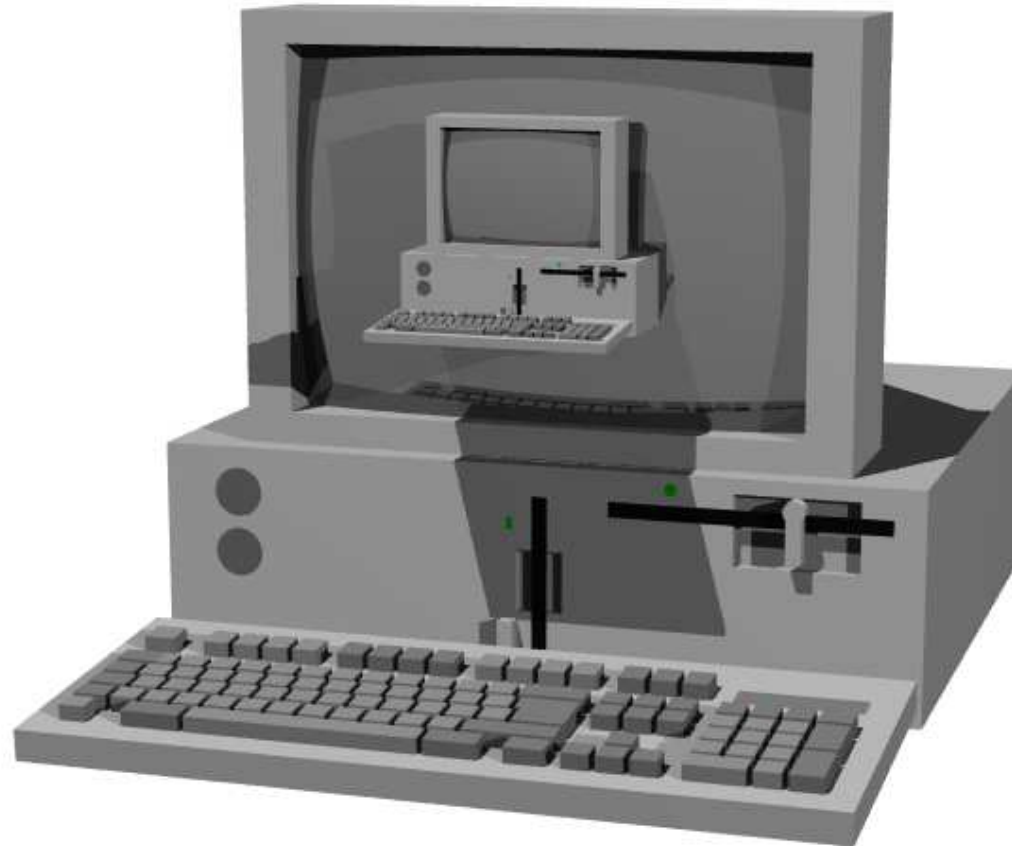


QEMU – Der Rechner im Rechner



Vortragsübersicht

- Was ist QEMU?
- Einsatzszenarien
- QEMU installieren / Beispielaufruf
- Virtuelles Netzwerk
- Snapshot-Modus
- Gastsystem einfrieren / speichern
- Inoffizielle Erweiterungen

Was ist QEMU?

- Vollständiger virtueller Rechner mit:
 - Pentium II CPU (optional SMP)
 - PCI NE2000 Netzwerkkarte(n)
 - PCI Cirrus Logic GD 5446 Grafikkarte
 - ISA Soundblaster 16 oder PCI ES1370 Soundkarte
 - Bis zu 4 IDE-Festplatten
 - IDE-CDROM-Laufwerk
 - Serieller Port (eingeschränkt)
 - Parallelport (eingeschränkt)
- Quelloffen, steht unter GPL/LGPL
- Braucht keine Kernelmodule oder root-Rechte
Optionales Beschleuniger-Kernelmodul [kqemu](#) (closed source)

Was ist QEMU? (2)

- Autor: Fabrice Bellard
- Bisher lauffähige Gastsysteme:
 - Linux 1.2/2.0/2.2/2.4/2.6
 - Free/Open/NetBSD
 - MS-DOS, FreeDOS
 - Win 95/98/ME, Win 2000, Win XP
 - Diverse Exoten :-)
- QEMU läuft unter Linux/*BSD, Windows und Mac OS X
- Aktuelle Version: 0.8.0

Was ist QEMU? (3)

Vergleich mit anderen VM-Lösungen:

- Anders als VMware emuliert QEMU die CPU komplett
Dadurch langsamer, aber nicht auf x86-Plattformen beschränkt
- Für XEN müssen die Gastsysteme gepatcht werden, für QEMU nicht
Wesentlich mehr Gastsysteme sind lauffähig
- Gegenüber Bochs ist QEMU 5-10mal schneller

Einsatzszenarien

- Testen von bootfähigen ISO-Images
- Virtueller Netzaufbau
- Ausprobieren von neuen Distributionen
- Was wäre wenn. . .
- Einsperren von Netzdiensten
- Kernel-Debugging

QEMU installieren

- Distributionspakete
 - Pakete existieren für alle gängigen Distributionen, nicht immer aktuell
- Vorkompilierte Version von <http://www.qemu.org/>
 - Tarball mit allen nötigen Dateien
 - Statisch gelinktes Binary für x86-Linux
- Kompilieren aus Sourcen
 - libSDL inkl. Headerdateien benötigt
 - gcc 3.x benötigt (gcc 4.x macht Probleme)

QEMU installieren (2)

Kompilieren aus Sourcen:

```
tar xzf qemu-0.8.0.tar.gz
cd qemu-0.8.0
./configure --target-list=i386-softmmu --enable-adlib
make
su -c "make install; strip /usr/local/bin/qemu*"
```


Beispielaufruf

Damn Small Linux testen:

```
qemu -m 64 -localtime -cdrom dsl.iso
```

Den virtuellen Rechner mit 64MB RAM starten, Uhrzeit als lokale Zeit vom Hostsystem übernehmen, booten vom ISO-Image `dsl.iso` als virtuelles CDROM-Laufwerk.

Während QEMU läuft, kann mit CTRL-ALT-2 in den eingebauten Monitor geschaltet werden, mit CTRL-ALT-1 kommt man zurück zur Darstellung des Gastsystems. DEMO

Beispielaufruf (2)

Ubuntu installieren:

```
qemu-img create -f qcow ubdisk.qimg 2500M
```

```
qemu -m 192 -localtime -soundhw es1370 -hda ubdisk.qimg \  
-cdrom ubuntu-5.10.iso -boot d
```

Ein 2.5GB Festplatten-Image erstellen, virtuellen Rechner mit 192MB RAM starten, lokale Zeit vom Host übernehmen, Ensoniq 1370 Soundkarte, Image `ubdisk.qimg` als primary master Festplatte (hda), Image `ubuntu-5.10.iso` als virtuelles CDROM-Laufwerk, booten von diesem Laufwerk.

Nach der Installation im erneuten QEMU-Aufruf die Option `-boot c` verwenden, um von der virtuellen Festplatte zu booten.

Virtuelles Netzwerk

- User Networking (default)
Option: `-net nic -net user`
 - „Applikatives NAT“
 - Eingebauter DHCP-Server (10.0.2.0/24)
 - Optional Portweiterleitung mit `-redir`
- tap/tun Networking
Option: `-net nic -net tap,...` DEMO
 - Bessere Steuerungsmöglichkeit, weil der Traffic über ein separates Interface (z.B. `tap0`) läuft
 - Konfiguration über Skript `/etc/qemu-ifup` oder `tunctl`
 - Einsatz von Routing und `iptables` möglich/nötig

Virtuelles Netzwerk (2)

- Socket Networking

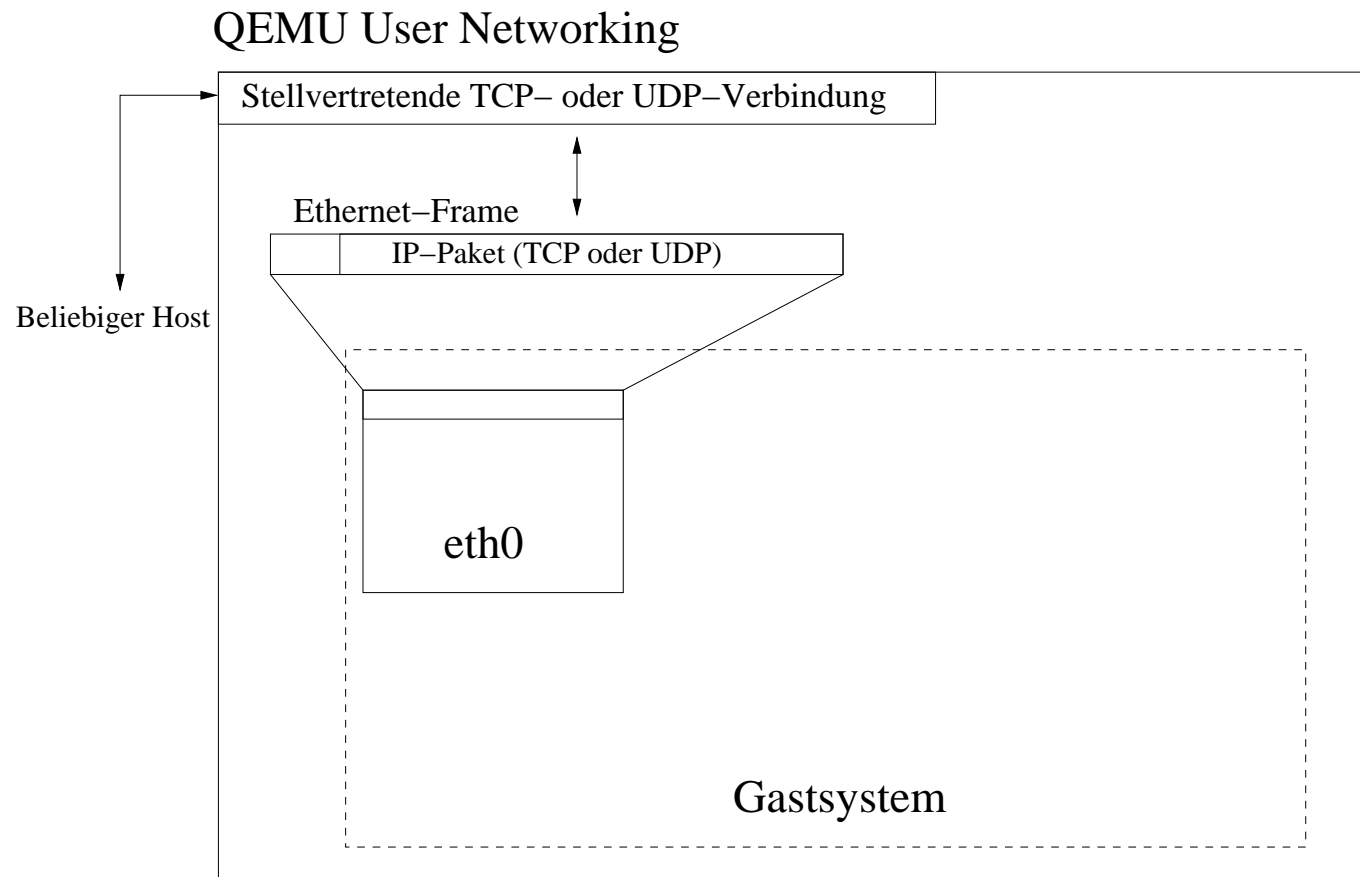
Option: `-net nic -net socket,listen...`

Option: `-net nic -net socket,connect...`

DEMO

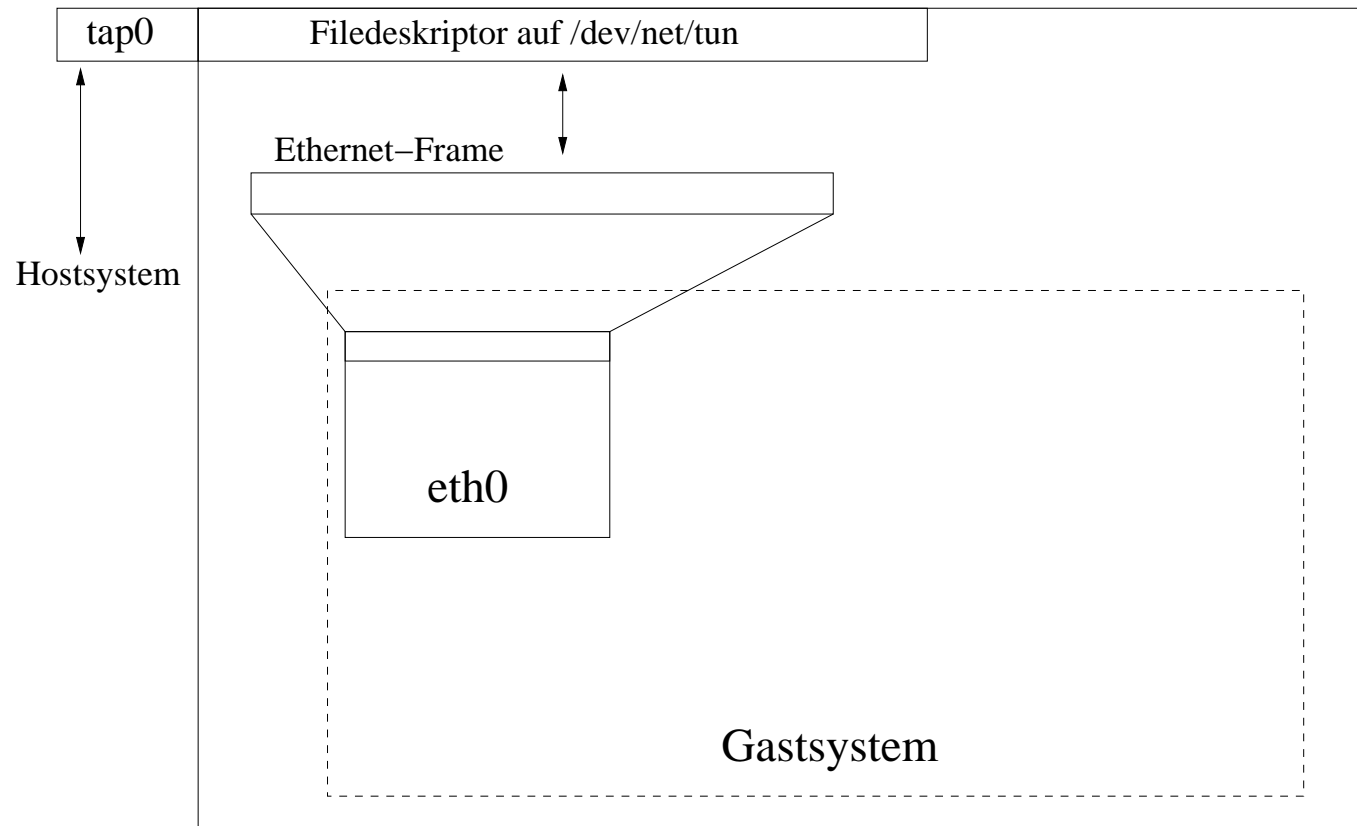
- Zusammenfassen von zwei oder mehr QEMU-Instanzen zu einem virtuellen LAN
- Verbindung auch mit remote laufendem QEMU möglich

Virtuelles Netzwerk (3)



Virtuelles Netzwerk (4)

QEMU tap/tun Networking



Snapshot-Modus

Option: `-snapshot` DEMO

- Änderungen werden *nicht* ins Festplatten-Image geschrieben, sondern nach `/tmp`
- Man kann beliebig im Gastsystem experimentieren, nichts geht dauerhaft kaputt!
- Bei Bedarf sind die Änderungen mit dem QEMU-Monitor-Kommando `commit` ins Image übertragbar

Gastsystem einfrieren / speichern

- Sichern des kompletten Gastsystem-Zustands DEMO
- Kommandos im QEMU-Monitor:
`stop`
`savevm dateiname`
`quit`
- Wiederherstellen des Zustands mit der Option `-loadvm`
- Virtuelle Hardware sollte zwischen `savevm` und `-loadvm` nicht oder nur wenig geändert werden

Gastsystem einfrieren / speichern (2)

Skript zur Automatisierung von `-loadvm`:

```
#!/bin/bash
QIMG=ubdisk.qimg
QOPTS="-m 192 -net nic -net tap,ifname=tap0,script=/bin/true"
if [ -e qdump ] ; then
    if [ qdump -ot $QIMG ] ; then
        qemu $QOPTS -hda $QIMG
    else
        qemu $QOPTS -hda $QIMG -loadvm qdump
    fi
else
    qemu $QOPTS -hda $QIMG
fi
```

Inoffizielle Erweiterungen

- VNC-Patch
VNC-Serverfunktion in QEMU. Das Gastsystem kann remote bedient werden, ohne eigene Netzwerkfunktionen zu besitzen.
- Realtek 8139 Patch, PCnet32 Patch
Weitere virtuelle Netzwerkkarten.
- Portierung für Solaris/SPARC

Links

- QEMU Homepage: <http://www.qemu.org/>
- Diverse Patches: <http://qemu.dad-answers.com/>
- Patches von malc: <http://www.boblycat.org/~malc/code/patches/qemu/>
- VNC-Patch: <http://libvncserver.sourceforge.net/qemu/>
- QEMU für Windows: <http://www.h7.dion.ne.jp/~qemu-win/>