cloop - a transparently decompressing block device

Klaus Knopper

in cooperation with

LinuxTag e.V.

Build date: 10.3.2002

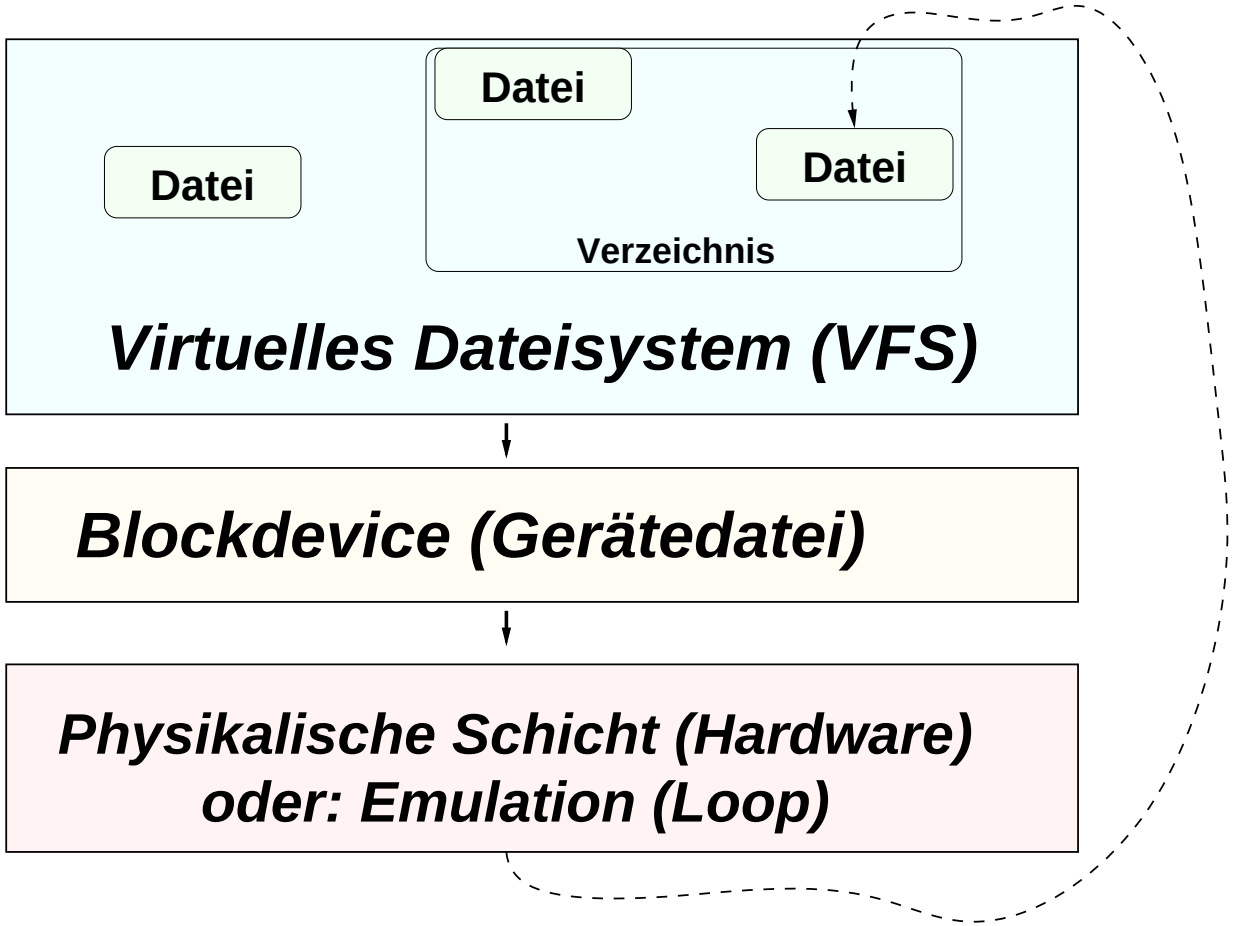
Zusammenfassung

`cloop` - Compressed Loopback Device - ist ein ursprünglich von Paul 'Rusty' Russel (Author of **`ipchains`** & **`iptables`**) für Linuxcare geschriebenes, auf **`loop.c`** von Kernel 2.2 basierendes Blockdevice, das das transparente Lesen blockweise komprimierter Images erlaubt. Das Modul wurde von Klaus Knopper auf Kernel 2.4 portiert und mit diversen Erweiterungen für das komprimierte Dateisystem für die KNOPPIX-CD versehen (wird jedoch auch von anderen Projekten wie der LNX-BBC verwendet). Durch die transparente Kompression können auf einer Standard 650MB-CD bis zu 2 Gigabyte an lauffähiger Software und Dokumentation installiert werden.

cloop Features

- Block-Device (Major 240, Minor 0),
- weitestgehend Dateisystem-unabhängig (Host- und Guest-Dateisystem),
- reines Modul, wird aus dem Quelltext für den jeweiligen Kernel gebaut, kein Kernel-Patch erforderlich [2],
- **gzip/zlib**-Blockkompression,
- in der aktuellen Version (0.63) sehr Lesefehlertolerant,
- Kernel 2.2 und 2.4-tauglich (per `#ifdef`),
- **`cloop`**-Modul & Utilities: GPL, **`libz`**/**`gzip`**: LGPL [3].

Dateisystem und Blockdevice-Layer



Übersetzen

```
make KERNEL_DIR=/usr/src/linux-2.4.18
```

- Liest Kernel-Optionen aus **autoconf.h** und setzt ggf. SMP-Option.
- übersetzt zlib-1.3 statisch,
- übersetzt **compressed_loop.c** und Utilities,
- erzeugt Binaries **create_compressed_fs** und **extract_compressed_fs**
- bindet zlib mit compressed_loop.o zu **cloop.o**.

Übersetzen (Debian-Paket)

```
apt-get source cloop
cd cloop-0.63
:> knoppix
fakeroot dpkg-buildpackage
```

Installieren

```
mknod -m 444 /dev/cloop b 240 0

install -m 755 cloop.o \
 /lib/modules/`uname -r`/kernel/drivers/block/

install -m 755 *compressed_fs /usr/sbin/
```

Debian:

```
dpkg -i ../cloop-utils_0.63-2_i386.deb
dpkg -i ../cloop-module_0.63-2_i386.deb
dpkg -i ../cloop-src_0.63-2_all.deb
```

Image erzeugen

```
mkisofs -r -l datadir | \
create_compressed_fs - 65536 > isoimg.z
```

Achtung: **`create_compressed_fs`** hält das komplette komprimierte Image bis zur vollständigen Abarbeitung und Schreiben der Header im virtuellen Speicher, damit es als PIPE (z.B. für **`cdrecord`**) verwendet werden kann. ☞ Für ausreichend SWAP sorgen!

Anwenden (1)

Z.B. bei KNOPPIX [1] in **`linuxrc`**:

```
insmod cloop.o file=/cdrom/KNOPPIX/KNOPPIX
mount -r /dev/cloop /mnt/knoppix
```

Anwenden (2)

```
knopper@Koffer:~/cloop-chemnitz2002$ df
Filesystem 1k-blocks    Used Available Use% Mounted on
/dev/hda1     104412   97208      7204  94% /
/dev/hda5    2562252 1520304   1041948  60% /usr
...
/dev/cloop   1848152 1848152         0 100% /mnt/knoppix

knopper@Koffer:~/cloop-chemnitz2002$ ls -l /mnt/knoppix/
-rw-r--r-- 1 knopper users 11776 Feb 13 19:52 powerslide
drwxr-xr-x 2 knopper users  2880 Mar 10 02:42 slides
-rwxr-xr-x 1 knopper users    57 Mar  9 04:26 start.sh
drwxr-xr-x 2 knopper users   112 Mar  7 01:39 style
drwxr-xr-x 2 knopper users   168 Mar  8 15:11 templates
...
```

Performance-Tuning (1)

- **iso9660** mit Rockridge (-R), evtl. abgeschalteten Kompatibilitätsflags (-U) ist ein leseoptimiertes Dateisystem, welches als Gast-Dateisystem einen sehr schnellen Zugriff bietet.
- Durch den Blockdevice-Cache werden **dekomprimierte** Blöcke im RAM gehalten, während (v.a. bei Kernel 2.4.x) durch geeignete Mapping-Routinen „doppeltes Caching“ der **komprimierten** Blöcke vermieden wird.
- Der zuletzt gelesene **komprimierte** Block wird Modul-intern vollständig gecacht ☞ Vermeiden von unnötigen physikalischen Lesevorgängen.

Performance-Tuning (2)

- Die Dateien sind für KNOPPIX in einer **`mkisofs.sortlist`** mit Lese-Timestamp vorsortiert, so dass sie einigermaßen in der richtigen Reihenfolge physikalisch auf der CD vorhanden sind und „am Stück“ gelesen werden können ☞ Reduzierung der Laserabtasterkopfbewegung.
- Höherer `gzip`-Kompressionsfaktor: bringt bei 64k-Blöcken nichts.
- `bzip2` statt `gzip`: Komprimiert eine CD mit gemischten Daten durchschnittlich ca. 5% besser, ist aber beim Dekomprimieren Faktor 4 langsamer als der **`gzip`**-Algorithmus.

Komprimiertes Image - Format

Alle Nummern und Pointer in network byte order:

1. 128 Bytes Header mit kleinem Shellskript als Selbst-Lader (**`insmod cloop.o file=/path/to/image`**),
2. Header mit Information *uncompressed* Blocksize und Anzahl der Blöcke,
3. Block Index:
   (a) Adresse des ersten komprimierten Blocks der Datei,
   (b) Adresse des 2. komprimierten Blocks der Datei,
   (c) ...
   (d) end-of-file Adresse.
4. Daten:
   (a) Komprimierter Block # 1,
   (b) Komprimierter Block # 2,
   (c) ...

Geplante Erweiterungen

- Unterstützung von Mehrfach-Loops (evtl. mit Hilfe von **`losetup`**-`ioctl()`s).
- Optimierung des Blockindex-Suchalgorithmus, redundanten Code entfernen.
- Schreibunterstützung: Problematisch, da komprimierte Blockgröße variabel und „Löschen“ von Blocks nicht möglich (gehört eigentlich in den VFS-Layer).
- Dekompressor in Assembler für häufigste Architekturen.
- **`devfs`**-Unterstützung.

Links

[1] http://www.knoppix.de/
Abkürzungs-URL zur KNOPPIX-CD.

[2] http://www.knopper.net/knoppix/sources/
Sourcen von **cloop**.

[3] http://www.gnu.org/
Homepage der GNU-Software und der GPL.

[4] http://www.lnx-bbc.org/
Homepage des Linux-Bootable-Businesscard Projekt.

[5] Kontakt/ Entwickler-Liste:
cloop@knopper.net
debian-knoppix@linuxtag.org