Ein Beitrag von Vimes

USB-Speichergeräte funktioneren ja unter Linux mittlerweile erfreulich gut. Anstecken, mounten, nutzen. Fertig. Den zweiten Schritt erledigt sogar häufig schon das Betriebssystem.

Manchmal möchte man aber, daß ein bestimmtes Laufwerk immer unter derselben Kennung erreichbar ist. Beispiel gefällig? Wenn ich meinen iPod abgleichen lassen möchte, ist es bequem, das mounten von gtkpod übernehmen zu lassen, also z.B. /dev/sda2 (weil die zweite Partition des iPods die Daten enthält). Funktioniert so auch ohne jede Probleme. Wenn ich aber bereits meine USB-Festplatte angeklemmt habe, dann ist DIE /dev/sda und der iPod erhält /dev/sdb. gtkpod schaut dann in die Röhre.
Anderes Beispiel: meine Backups werden von einem Script erstellt. Dieses Script soll idealerweise die fertigen Archive auch gleich auf meiner externen Festplatte ablegen. Dafür muß der Pfad aber bekannt und absolut sein, nicht relativ. Es genügt nicht, zu wissen, daß es /dev/sd* ist. Ob sda, sdb oder gar sdc ist dem Script nämlich nicht egal.

Ja, es gibt für solche Probleme eine Lösung oder genauer gesagt, zwei Lösungen, die beide bei udev eingebaut sind.
Udev übernimmt die Verwaltung von USB-Geräten (nicht nur Speichergeräten, aber andere lasse ich hier mal außen vor). Mehr dazu: http://de.wikipedia.org/wiki/Udev

Als ersten Schritt schalten wir es aus, daß udev sich alle "Orte" merkt, die udev angelegt hat; hierfür muß der Eintrag "RC_DEVICE_TARBALL" von "no" auf "yes" gesetzt werden. Unter gentoo steht diese Variable unter /etc/conf.d/rc ; für andere Distributionen empfehle ich, danach zu grepen; für debian/Ubuntu kann man diesen Schritt überspringen, da weder die Datei noch ein Eintrag dieser Form existieren.

```
discworld ~ # grep -irH 'RC_DEVICE_TARBALL' /etc/
```

Die Ausgabe sieht dann so aus:

```
/etc/conf.d/rc:RC_DEVICE_TARBALL="no"
```

wobei - logisch - "/etc/conf.d/rc:" den Speicherort der Datei angibt.

Jetzt gibt es, wie schon gesagt, zwei Wege, einem Laufwerk einen festen Pfad zu verpassen:

1. das feste Schema von udev

Dazu schließen wir zunächst das USB-Gerät, z.B. einen USB-Stick, an. Anschließend tippen wir als root das hier ein:

```
discworld ~ # ls -lR /dev/disk
```

Die Ausgabe, die dann erscheint, ist auf den ersten, zweiten und vermutlich auch dritten Blick sehr verwirrend:

```
./by-id:
total 0
lrwxrwxrwx 1 root root 9 2006-08-27 20:18 usb-SanDisk_Cruzer_Titanium_0000000000000212457 -
> ../../sda
lrwxrwxrwx 1 root root 10 2006-08-27 20:18 usb-SanDisk_Cruzer_Titanium_0000000000000212457-
part1 -> ../../sda1

./by-path:
total 0
lrwxrwxrwx 1 root root 9 2006-08-27 20:18 usb-0000000000000212457:0:0:0 -> ../../sda
lrwxrwxrwx 1 root root 10 2006-08-27 20:18 usb-0000000000000212457:0:0:0-part1 -> ../../sda1
```

(Den Rest der Ausgabe habe ich abgeschnitten, er ist uninteressant)

Wir interessieren uns jetzt für den Eintrag

/by-id:
und dort für den zweiten Abschnitt:
usb-SanDisk_Cruzer_Titanium_0000000000000212457-part1 -> ../../sda1

Der verrät uns, daß das angeschlossene Gerät ein Cruzer Titanium von Sandisk ist und die obige, kryptische und fest verdrahtete ID hat. Das heißt, mit Hilfe dieser (ellenlangen) ID kann man den Speicherstick eindeutig identizieren.

Warum nehmen wir nur den zweiten Abschnitt? Weil dieser die erste (und einzige) Partition des Speichersticks verrät. Der Eintrag darüber sagt nur,

Und jetzt?

Ganz einfach, jetzt legt man in der /etc/fstab einen Eintrag an, der so aussieht:
/dev/disk/by-id/usb-SanDisk_Cruzer_Titanium_0000000000000212457-part1 /mnt/cruzer auto noauto,user 0 0

Ab sofort kann man den Stick (als Benutzer) mit dem folgenden Befehl mounten:

mount /mnt/cruzer

und dabei ist es egal, ob der Stick als erstes, zweites oder fünftest Gerät angeschlossen wird.

Ok, ich gebe zu, dieser ellenlange Eintrag in der /etc/fstab ist etwas unhandlich. Und sieht auch sehr unschön aus.

Update:
Die Methode, eine Verknüpfung anzulegen, funktioniert nur eingeschränkt - sie wird nach jedem Neustart gelöscht. Unschön, aber nicht zu ändern.

Kommen wir zu 2. Methode:

2. udev-Regeln

udev-Regeln sind eleganter. Allerdings ist es mir bisher nicht gelungen, sie mit USB-Sticks zu verwenden, sondern nur mit einer externen Festplatte und meinem iPod.
Die Regeln gelten für die udev-Version 0.87.

udev-Regeln werden im Verzeichnis

/etc/udev/rules.d/

gespeichert (das gilt zumindest für gentoo und Ubuntu).

Werfen wir mal einen Blick hinein:

05-udev-early.rules 50-udev.rules

05-udev-early.rules enthält wichtige Regeln, die als allererstes ausgeführt werden. Diese läßt man in Ruhe.
50-udev.rules enthält einen ganzen Haufen Regeln für alle denkbaren Geräte. Auch diese Datei läßt man besser in Ruhe; wer sie löscht, steht vor dem Problem, daß das Betriebssystem die allermeisten Geräte nicht mehr zuordnen kann.

Die Nummern vor den Dateien (05 bzw. 50) geben die Reihenfolge an, in der sie abgearbeitet werden.

Update:
Unter debian (und vermutlich auch Ubuntu) sieht die Liste doch ein klein wenig anders aus:

010_permissions.rules z45_persistent-net-generator.rules
020_permissions.rules z50_run.rules
udev.rules z55_hotplug.rules
z20_persistent-input.rules z60_alsa-utils.rules
z20_persistent.rules z60_hdparm.rules
z25_persistent-cd.rules z60_xserver-xorg-input-wacom.rules
z25_persistent-net.rules z75_cd-aliases-generator.rules

Wenn wir jetzt eine Datei names

10-udev.rules

anlegen, dann wird diese nach der 05, aber vor der 50 abgearbeitet. Und genau das tun wir jetzt.

Update: Für debian/Ubuntu siehe oben. Hier gibt es keine 05-Datei, es genügt, eine Datei mit dem Titel 010_permission.rules (oben fett) anzulegen und die Regeln hineinzuschreiben. Wichtig ist, daß die Zahl die niedrigste ist (damit die Regeln als erste abgearbeitet werden) und daß die Datei auf .rules endet. Sonst wird sie nicht als Regel-Datei erkannt und ignoriert.

Zuerst einmal muß man wissen, wie man angeschlossene USB-Geräte überhaupt erkennen kann.

Hilfreich ist hier die Seite

http://www.reactivated.net/writing_udev_rules.html

aus der ich mich bedient habe.

Aber zurück zum Thema. Wenn ich einem USB-Gerät über eine Regel einen festen Gerätepfad zuweisen möchte, muß ich es irgendwie eineindeutig (kein Tippfehler) identifizieren können.

Hier kommt das Programm udevinfo ins Spiel.

Mit der Eingabe (als root) von

discworld ~# udevinfo -ap /sys/block/sda

bekommt man einen Haufen Kram angezeigt, den wir uns am Beispiel meiner USB-Festplatte mal anschauen. (Wenn man das ganze so schreibt:
discworld ~# udevinfo -ap /sys/block/sda >> /home/$Username/usb-geraet.txt
dann wird die Ausgabe in eine Textdatei geschrieben, die im eigenen Home-Verzeichnis liegt - bitte aber vorher den eigenen Usernamen eintragen! - und usb-geraet.txt heißt. Ist augenfreundlicher)
(Wichtig: das USB-Gerät muß angeschlossen sein und im obigen Falle muß es das erste angeschlossene Speichermedium sein. Ist also bereits ein USB-Stick angeschlossen, dann stecke ich meine Festplatte an und möchte diese über udev identifzieren, dann muß ich oben sdb schreiben, weil es bereits das zweite Gerät ist.)

Als Ausgabe erhalte ich für meine USB-Festplatte das hier:

udevinfo starts with the device the node belongs to and then walks up the
device chain, to print for every device found, all possibly useful attributes
in the udev key format.
Only attributes within one device section may be used together in one rule,
to match the device for which the node will be created.

looking at device '/block/sda':
KERNEL=="sda"
SUBSYSTEM=="block"
SYSFS{stat}==" 39 364 410 780 0 0 0 0 0 330 780"
SYSFS{size}=="586072368"
SYSFS{removable}=="0"
SYSFS{range}=="16"
SYSFS{dev}=="8:0"

looking at device '/devices/pci0000:00/0000:00:1d.7/usb1/1-4/1-4:1.0/host1/target1:0:0/1:0:0:
0':
ID=="1:0:0:0"
BUS=="scsi"
DRIVER=="sd"
SYSFS{ioerr_cnt}=="0x0"
SYSFS{iodone_cnt}=="0x2e"
SYSFS{iorequest_cnt}=="0x2e"
SYSFS{iocounterbits}=="32"
SYSFS{timeout}=="30"
SYSFS{state}=="running"
SYSFS{rev}=="0000"
SYSFS{model}=="HD300LD "
SYSFS{vendor}=="SAMSUNG "
SYSFS{scsi_level}=="3"
SYSFS{type}=="0"
SYSFS{queue_type}=="none"
SYSFS{queue_depth}=="1"
SYSFS{device_blocked}=="0"
SYSFS{max_sectors}=="240"

looking at device '/devices/pci0000:00/0000:00:1d.7/usb1/1-4/1-4:1.0/host1/target1:0:0':
ID=="target1:0:0"
BUS==""
DRIVER==""

looking at device '/devices/pci0000:00/0000:00:1d.7/usb1/1-4/1-4:1.0/host1':
ID=="host1"
BUS==""
DRIVER==""

looking at device '/devices/pci0000:00/0000:00:1d.7/usb1/1-4/1-4:1.0':
ID=="1-4:1.0"
BUS=="usb"
DRIVER=="usb-storage"
SYSFS{modalias}=="usb:v04B4p6830d0240dc00dsc00dp00ic08isc06ip50"
SYSFS{bInterfaceProtocol}=="50"
SYSFS{bInterfaceSubClass}=="06"
SYSFS{bInterfaceClass}=="08"
SYSFS{bNumEndpoints}=="03"
SYSFS{bAlternateSetting}==" 0"
SYSFS{bInterfaceNumber}=="00"

looking at device '/devices/pci0000:00/0000:00:1d.7/usb1/1-4':
ID=="1-4"
BUS=="usb"
DRIVER=="usb"
SYSFS{configuration}==""
SYSFS{serial}=="DEF10BD9ECC1"

SYSFS{product}=="Cypress AT2LP RC7"
SYSFS{maxchild}=="0"
SYSFS{version}==" 2.00"
SYSFS{devnum}=="5"
SYSFS{speed}=="480"
SYSFS{bMaxPacketSize0}=="64"
SYSFS{bNumConfigurations}=="1"
SYSFS{bDeviceProtocol}=="00"
SYSFS{bDeviceSubClass}=="00"
SYSFS{bDeviceClass}=="00"
SYSFS{bcdDevice}=="0240"
SYSFS{idProduct}=="6830"
SYSFS{idVendor}=="04b4"
SYSFS{bMaxPower}==" 2mA"
SYSFS{bmAttributes}=="c0"
SYSFS{bConfigurationValue}=="1"
SYSFS{bNumInterfaces}==" 1"

looking at device '/devices/pci0000:00/0000:00:1d.7/usb1':
ID=="usb1"
BUS=="usb"
DRIVER=="usb"
SYSFS{configuration}==""
SYSFS{serial}=="0000:00:1d.7"
SYSFS{product}=="EHCI Host Controller"
SYSFS{manufacturer}=="Linux 2.6.17.6 ehci_hcd"
SYSFS{maxchild}=="8"
SYSFS{version}==" 2.00"
SYSFS{devnum}=="1"
SYSFS{speed}=="480"
SYSFS{bMaxPacketSize0}=="64"
SYSFS{bNumConfigurations}=="1"
SYSFS{bDeviceProtocol}=="01"
SYSFS{bDeviceSubClass}=="00"
SYSFS{bDeviceClass}=="09"
SYSFS{bcdDevice}=="0206"
SYSFS{idProduct}=="0000"
SYSFS{idVendor}=="0000"
SYSFS{bMaxPower}==" 0mA"
SYSFS{bmAttributes}=="e0"
SYSFS{bConfigurationValue}=="1"
SYSFS{bNumInterfaces}==" 1"

looking at device '/devices/pci0000:00/0000:00:1d.7':
ID=="0000:00:1d.7"
BUS=="pci"
DRIVER=="ehci_hcd"
SYSFS{modalias}=="pci:v00008086d000024DDsv00001297sd0000FB65bc0Csc03i20"
SYSFS{local_cpus}=="1"
SYSFS{irq}=="3"
SYSFS{class}=="0x0c0320"
SYSFS{subsystem_device}=="0xfb65"
SYSFS{subsystem_vendor}=="0x1297"
SYSFS{device}=="0x24dd"
SYSFS{vendor}=="0x8086"

looking at device '/devices/pci0000:00':
ID=="pci0000:00"
BUS==""

DRIVER==""

Jetzt gilt: Eine udev-Regel darf nur aus Einträgen aus demselben Block bestehen, sonst ist die ungültig. Ein Block beginnt immer mit "looking at device...".

Für meine Festplatte habe ich diesen Block hier genommen:

looking at device '/devices/pci0000:00/0000:00:1d.7/usb1/1-4/1-4:1.0/host1/target1:0:0/1:0:0:0':
ID=="1:0:0:0"
BUS=="scsi"
DRIVER=="sd"
SYSFS{ioerr_cnt}=="0x0"
SYSFS{iodone_cnt}=="0x2e"
SYSFS{iorequest_cnt}=="0x2e"
SYSFS{iocounterbits}=="32"
SYSFS{timeout}=="30"
SYSFS{state}=="running"
SYSFS{rev}=="0000"
SYSFS{model}=="HD300LD "
SYSFS{vendor}=="SAMSUNG "
SYSFS{scsi_level}=="3"
SYSFS{type}=="0"
SYSFS{queue_type}=="none"
SYSFS{queue_depth}=="1"
SYSFS{device_blocked}=="0"
SYSFS{max_sectors}=="240"

Wie wird die Regel jetzt aufgebaut?
Erstmal muß mitgeteilt werden, wie das Gerät überhaupt angesprochen wird. Das besorgt der Eintrag "BUS==".
Im obigen Block lesen wir, daß BUS==scsi.

Also beginnt unsere Regel so:

BUS=="scsi"

Als nächstes müssen wir die Festplatte eindeutig identifzieren. Oben steht ein Eintrag mit dem Wert "HD300LD". Das genügt mir, ich habe sonst kein Gerät, das eine solche Modell-Kennung hat. Also lautet die Regel jetzt:

BUS=="scsi", SYSFS{model}=="HD300LD"

Wie man sieht, werden die einzelnen Bestandteile jeweils mit einem Komma getrennt.
Wichtig ist hier, daß == eine Zuweisung darstellt. Der BUS hat den Wert "scsi". Ein einfaches = genügt nicht!

Als letzten Schritt muß ich udev noch mitteilen, welche Partitionen unter welchem Gerätepfad erkannt werden sollen. Es ist mir mit obiger Webseite nicht gelungen, unter meiner udev-Version gezielt einzelne Partitionen einzubinden. Tante Google verriet mir, daß in meiner udev-Version die Syntax offensichtlich anders ist. Wenn ich alle Partitionen erkennen lassen möchte, genügt dieser Eintrag:

NAME{all_partitions}="/exthd/part"

Damit werden dann alle erkannten Partitionen unter /exthd/partNR durchnummeriert.

Zusammengesetzt sieht die Regel dann so aus:

BUS=="scsi", SYSFS{model}=="HD300LD", NAME{all_partitions}="/exthd/part"

In die /etc/fstab wandert dann dieser Eintrag hier:

/dev/exthd/part1 /mnt/exthd ext3 noauto,user 0 0

bzw. für die zweite Partition auf der Platte:

/dev/exthd/part2 /mnt/exthdnt ntfs noauto,user 0 0

Das Laufwerk läßt sich dann mit einem simplen

mount /mnt/exthd bzw. mount /mnt/exthdnt mounten.

Das war's dann auch schon.

Update: mittlerweile ist es mir auch gelungen, meinen USB-Stick über udev-Regeln einzubinden. Ulkigerweise funktionierte es mit dieser Zeile hier:

BUS=="usb", SYSFS{serial}=="00000000000000212457", NAME{all_partitions}="/cruzer/part"

Anscheinend lassen sich manche Geräte nur über bestimmte Eigenschaften ansprechen, bei obigem USB-Stick war es nicht möglich, ihn über den Modellnamen anzusprechen, während das mit Festplatte und iPod auf Anhieb funktionierte.
Obige Zeile funktioniert sowohl unter gentoo als auch unter debian (und damit wohl auch Ubuntu).

MfG
Vimes