(C) 2024 OSTC GmbH (http://www.ostc.de)

HOWTO zu UNIX-Links

(C) 2007-2013 T.Birnthaler/H.Gottschalk <howtos(at)ostc.de>
OSTC Open Source Training and Consulting GmbH
http://www.ostc.de

$Id: unix-links-HOWTO.txt,v 1.8 2019/11/26 19:37:07 tsbirn Exp $

Dieses Dokument beschreibt Eigenschaften und Einsatzzweck von Links unter UNIX.

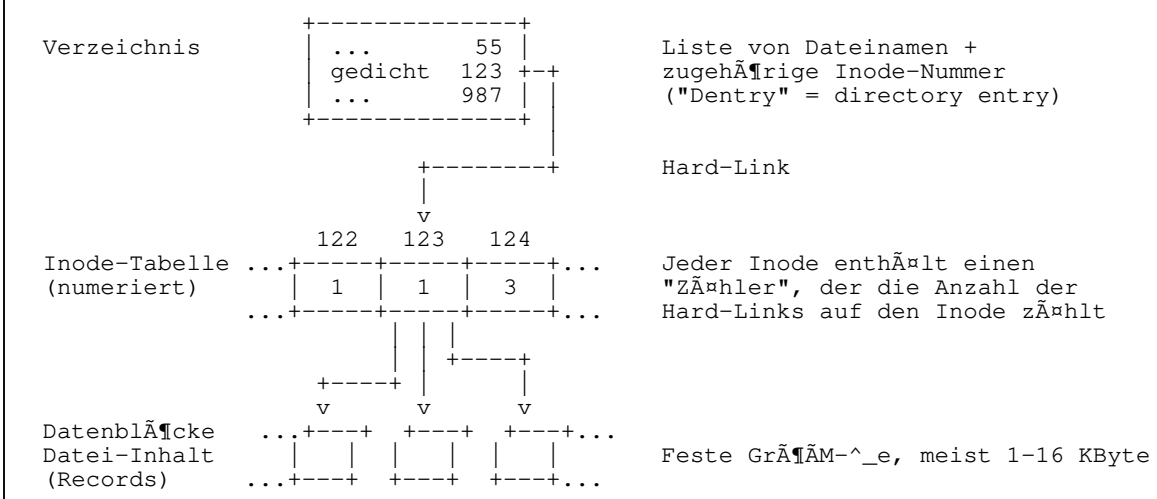
INHALTSVERZEICHNIS

1) Ã‌M-^\bersicht
2a) Hard-Link (Name + I-Node-Nummer)
2b) Inode ("Index-Knoten", mit Datei-Attributen)
2c) Datei-Inhalt
2d) Beispiel
3) Einsatzzweck von Links
4) Die beiden Link-Arten Hard-Link und Symbolischer Link
5) Vergleich mit Windows

---

1) Ã‌M-^\bersicht

Eine Datei setzt sich unter UNIX aus 3 Komponenten zusammen, die auch an 3 verschiedenen Stellen im UNIX-Dateisystem abgelegt werden:

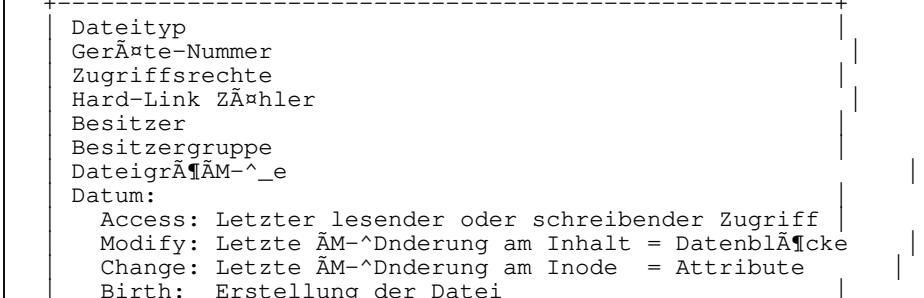


2a) Hard-Link (Name + I-Node-Nummer)

In jedem Verzeichnis ist nur eine Liste sogenannter "Hard-Links" bestehend aus Dateiname + I-Node-Nummer ("Dentry" = directory entry) eingetragen. Diese verbinden den Dateinamen mit der zugehÃ¶rigen Verwaltungsstruktur namens "I-Node" (Index Node). Es kann mehr als einen Hard-Link auf die gleiche Datei geben. In jedem Inode wird intern mitgezÃ¤hlt, wieviele Hard-Links aktuell auf ihn zeigen.

2b) Inode ("Index-Knoten", mit Datei-Attributen)

Der I-Node enthÃ¤lt (bis auf den Datei-Namen und den Datei-Inhalt) alle fÃ¼r eine Datei relevanten Informationen. Er zeigt weiterhin auf die DatenblÃ¶cke, in denen sich entweder der Inhalt der Datei befindet oder die auf weitere DatenblÃ¶cke zeigen (direkte und 1/2/3-fach indirekte Verweise).



```
   | Direkte Verweise auf DatenblÃ¶cke                      |
   | Indirekte Verweise auf DatenblÃ¶cke:                   |
   |   Einfach indirekt                                    |
   |   Zweifach indirekt                                   |
   |   Dreifach indirekt                                   |
   +-------------------------------------------------------+
```

2c) Datei-Inhalt

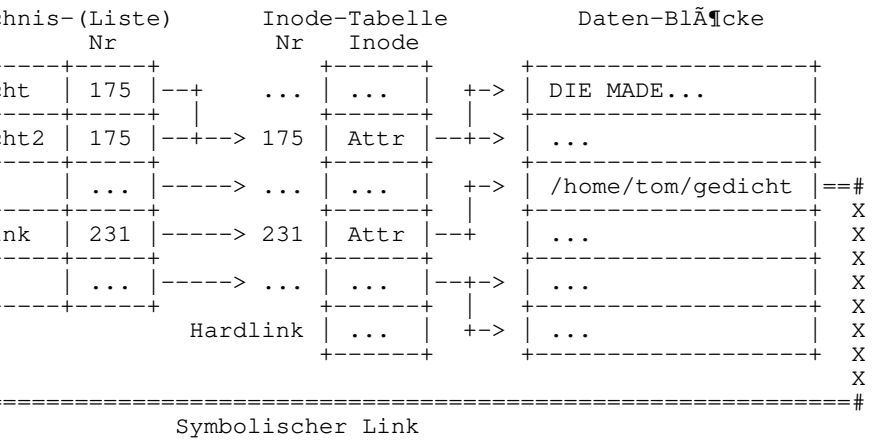
Der Datei-Inhalt ist in DatenblÃ¶cken abgespeichert, die im Dateisystem verstreut liegen.

2d) Beispiel
---

* Aktuelles Verzeichnis ist "/home/tom/".

* Im Verzeichnis gibt es 2 Hard-Links namens "gedicht" und "gedicht2" auf die gleiche Datei.

* Weiterhin gibt es darin 1 Symbolischen Link namens "symlink", der auf "gedicht" zeigt.



3) Einsatzzweck von Links
---

* Immer den gleichen Namen fÃ¼r ein GerÃ¤t zur VerfÃ¼gung stellen, egal welche konkrete Hardware dahinter steckt (Bsp: "/dev/mouse").

* FÃ¼r den schnellen Zugriff in Benutzer-Verzeichnissen eine "Kopie" einer (tief) im Dateisystem gespeicherten Datei ablegen.

* Plattenplatz sparen, wenn die gleiche Datei mehrfach im Dateisystem vorkommen soll (Verweise statt Kopien).

* Schreibfehler oder NamensÃ¤nderungen wegen FHS (File Hierarchy Standard) abfangen (z.B. YaST = yast = zast, conf.modules = modules.conf).

* Konsistenz erhalten bei Versionierung, Updates, Konfigurationsdateien (Bsp: /usr/src/linux -> linux-2.2.5.SuSE).

* Eine von mehreren Versionen einer Datei oder eines Programms als Standard auswÃ¤hlen (z.B. /usr/bin/bash -> /usr/local/bin/bash-4.0)

* UNIX-Besonderheit: Das gleiche Programm/Skript kann unter verschiedenen Namen referenziert werden und verhÃ¤lt sich dann je nach Name, unter dem es aufgerufen wird, verschieden (Bsp: mv = cp = ln, mkfs.ext2 = mke2fs).

Beispiele fÃ¼r mehrere Namen fÃ¼r die gleiche Datei sind:

UNIX	Kommandos	Grund
SuSE-Linux	Yast2 yast2 zast yast1 yast	Schreibfehler abfangen
Solaris	cp mv ln	Programmierung erleichtern
Linux	gunzip gzip zcat	Programmierung erleichtern
Solaris	/usr/ucb/ps /usr/bin/ps	BSD + SYSV-Verhalten gleichz.

4) Link-Arten Hard-Link und Symbolischer Link

(C) 2024 OSTC GmbH (http://www.ostc.de)

UNIX unterscheidet 2 Arten von Links, die aber beide fÃ¼r den gleichen Zweck eingesetzt werden, nÃ¤mlich zusÃ¤tzliche Namen fÃ¼r eine einzelne Datei zur VerfÃ¼gung zu stellen:

1) Hard-Link: Physikalischer Link (per I-Node)
2) Soft-Link: Symbolischer Link (per Name)

1) Hard-Links gibt es --- wie oben beschrieben --- fÃ¼r jede Datei mindestens einen, beliebig viele weitere kÃ¶nnen folgendermaÃM-^_en angelegt werden:

```
ln DATEI HARDLINK      # Analog "cp DATEI KOPIE"
```

Ihre Verwendung ist in einigen Punkten eingeschrÃ¤nkt (z.B. nur innerhalb einer Partition einsetzbar).

2) Symbolische Links (auch Soft-Links) werden folgendermaÃM-^_en angelegt:

```
ln -s DATEI SOFTLINK   # Analog "cp DATEI KOPIE"
```

und verweisen per (Pfad)Name auf eine andere Datei. Der Pfadname kann relativ oder absolut sein.

Man kann sich das so vorstellen, dass eine Datei als Wert den (Pfad)Namen einer anderen Datei enthÃ¤lt. Erst beim Zugriff auf einen Symbolischen Link wird festgestellt, ob seine Zieldatei wirklich existiert --- und wenn ja --- diese benutzt. Sie sind mit HTML-Links oder Windows-VerknÃ¼pfungen zu vergleichen (Verweis per URL = Name auf eine andere HTML-Seite).

Die beiden Link-Typen kÃ¶nnen folgendermaÃM-^_en charakterisiert werden:

	Hard Physical	Soft Symbol.
Referenz per	I-Node	Name
Anzahl Links auf Datei bekannt	Ja	Nein
Ã M-^\ber Partitionen hinweg mÃ¶glich	Nein	Ja
Ziel kann ein Verzeichnis sein	Nein	Ja
Rekursion mÃ¶glich (Verz. enthÃ¤lt sich selbst)	Nein	Ja
Ziel muÃM-^_ existieren (kann nicht ins Leere zeigen)	Ja	Nein
Geschwindigkeit	schnell	langsam
Unter WINDOWS vorhanden (VerknÃ¼pfung)	Nein	Ja

5) Vergleich mit Windows

Symbolische Links gibt es unter Windows nur in Form der "Desktop-Icons" = Verweise auf Programme, Daten oder Verzeichnisse. Allerdings kÃ¶nnen diese wirklich nur auf dem Desktop liegen, innerhalb von Verzeichnissen sind sie nicht mÃ¶glich. D.h. dieser Link-Typ ist unter Windows nur eingeschrÃ¤nkt verfÃ¼gbar.

Hard-Links gibt es unter Windows erst mit dem Dateisystem NTFS. Zu ihrer Verwaltung gibt es aber keine grafische MÃ¶glichkeit im Windows Explorer, sondern sie sind nur per Kommandos nutzbar.