# Hard-/ und Softlinks

Unterschiede und Anwendungen in modernen Computersystemen

Stefan Kühnel
Department of Computer Science & Mathematics
University of Applied Sciences
Munich Bavaria Germany
stefan.kuehnel@hm.edu

## Abstrakt

Hard-/ und Softlinks spielen eine große Rolle in modernen Computersystemen. Sie werden meist auf Administratorebene eingesetzt und sind daher oft nur der technisch versierten Fachöffentlichkeit bekannt. Ziel dieser wissenschaftlichen Arbeit ist es deshalb, Unterschiede und Anwendungen von Hard-/ und Softlinks herauszuarbeiten sowie Eigenschaften und Limitierungen der beiden Ansätze aufzuzeigen. Der zentrale Aspekt interner Verknüpfungen wird darüber hinaus durch die Erstellung eines Hardlinks unter Windows 10 verdeutlicht und hierdurch ein Bezug zur Praxis hergestellt.

## Keywords

Hardlink, Softlink, Inodes, Dateisysteme, File-Record Number

## 1 Einführung

Die Verwendung eines Computers, sei es für private oder berufliche Zwecke, geht mit einer intensiven Nutzung des Dateisystems einher. Egal, ob man gerade mit einer neuen Projektarbeit beschäftigt ist, sich Musik aus dem Internet herunterlädt oder ein Programm öffnet. Ohne das Dateisystem würden all diese Operationen nicht funktionieren. Doch obwohl das Dateisystem eine so omnipräsente Rolle im alltäglichen Umgang mit Computern spielt, kennt ein Großteil der Nutzer keine detaillierten Einzelheiten über dessen Funktionalität.

Die nachfolgenden Kapitel sollen deshalb einen Überblick über die Funktionalität des Dateisystems geben, wobei ein Schwerpunkt auf den Teilbereich der Hard-/ und Softlinks gelegt wird.

## 2 Erstellung einer Textdatei

Der wohl einfachste und offensichtlichste Weg eine Textdatei unter Windows zu erstellen, ist über das Kontextmenü des Dateiexplorers. Fährt man nach dem Aufruf des Menüs mit der Maus über den Reiter „Neu“, so erscheint die Option „Textdokument“. Vergibt man der Textdatei dann noch einen entsprechenden Namen, so ist der Task „Textdokument erstellen“ erfolgreich abgeschlossen und man kann mit der Bearbeitung beginnen.

Doch nun könnte man sich die Frage stellen, wie ein Dateisystem überhaupt die zuvor erstellte Datei auf der Festplatte wiederfinden kann. Nutzt das Dateisystem auch Dateinamen oder steckt dahinter ein völlig anderes System?

Wenn man den Prozess der Erstellung eines Textdokuments noch einmal gedanklich durchgeht, so könnte aufgrund der Abfrage des Dateinamens durchaus der Eindruck erweckt werden, dass das Dateisystem auf Dateinamen zum Auffinden von Dokumenten angewiesen ist. Dies ist jedoch nicht zutreffend, da das Dateisystem auch ohne die Verwendung von Dateinamen funktionieren würde. [11]

Erstellt man auf einem Computer eine neue Datei, so weist der Dateisystemhandler dieser im ersten Schritt eine eindeutige Nummer, fachsprachlich auch „Inode“ oder „File-Record Number“ genannt, zu. Diese kann dann im Anschluss verwendet werden, um den Inhalt der Datei auf der Festplatte wiederzufinden. [6]

Doch bis zu diesem Zeitpunkt fehlt noch die Verbindung zum Dateinamen. Das Dateisystem hat durch die Inode keine Probleme mehr, die Datei auf der Festplatte zu finden, doch dem Dateiexplorer ist die Datei noch unbekannt. [11]

Im nächsten Kapitel wird daher auf die Verknüpfung von Dateiname und Dateiinhalt auf der Festplatte eingegangen.

## 3 Hardlink

Bis zum jetzigen Zeitpunkt kann lediglich das Dateisystem die Datei auf der Festplatte auffinden und den Dateiinhalt auslesen. Der Dateiexplorer hat diese Möglichkeit noch nicht. Der nächste Schritt erfordert deshalb die Bekanntmachung der Datei mit dem Dateiexplorer.

Dazu wird in einem bestehenden Verzeichnis ein neuer Verzeichniseintrag erstellt, der Namen und Inode einer Datei miteinander verknüpft. (vgl. Honarmand 2017, S. 4) [8] In der Fachsprache wird eine solche Verknüpfung auch als Hardlink bezeichnet. (vgl. ebd., S. 7) [8] Da nun der Dateiname über die Inode mit dem entsprechenden Speicherbereich auf der Festplatte verknüpft ist (vgl. Abb. 1), kann der Dateiexplorer den Inhalt der Datei auch anzeigen.

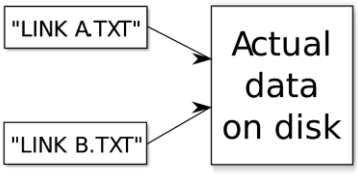


*Abb. 1: Vereinfachte Illustration von Hardlinks*

## 3.1 Eigenschaften

In der nachfolgenden Grafik wird deutlich, dass es auch mehrere Hardlinks zu einer entsprechenden Inode geben kann. [7]

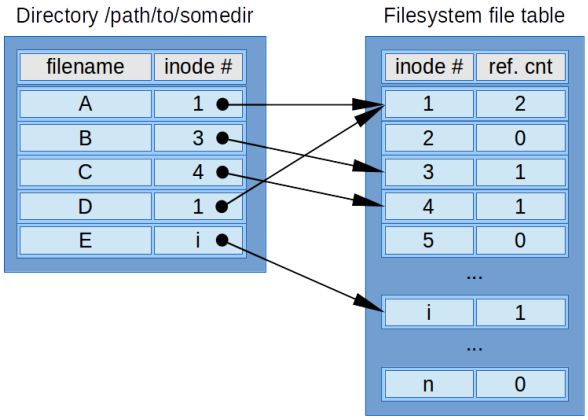


*Abb. 2: Funktion von Hardlinks unter UN*X Dateisystemen*

Ein Beispiel hierfür stellt Inode 1 dar. Sowohl Datei A als auch Datei D verweisen auf diese Inode, wodurch auch der Referenzzähler „ref. cnt" entsprechend den Wert 2 besitzt.

Dieses Charakteristikum führt nun unweigerlich dazu, dass eine Änderung von Datei D auch zu einer Änderung in Datei A führt, da beide Dateien über Inode 1 auf denselben Speicherbereich verweisen.

Unter Windows kann diese Eigenschaft folgendermaßen getestet werden.

1. Schritt: Erstellen Sie auf Ihrem Desktop eine neue Textdatei mit dem Namen F1.txt.
2. Schritt: Öffnen Sie Ihre Kommandozeile über die Tastenkombination [ + R → cmd] und wechseln Sie über den `cd` Befehl auf den Desktop.
3. Schritt: Führen Sie folgenden Befehl aus:

```
$ fsutil hardlink create F2.txt F1.txt
```

4. Schritt: Änderungen von F1.txt sollten auch in F2.txt sichtbar sein. [2]

Eine weitere Besonderheit von Hardlinks umfasst das Entfernen von Dateien von der Festplatte. Erst wenn der Referenzzähler einer Inode den Wert 0 erreicht hat, wird der Speicherbereich durch das Dateisystem zum Überschreiben oder Löschen freigegeben. [7] Dies erklärt auch, warum es im Grunde gesagt keine direkte Löschfunktion gibt, da Dateien lediglich überschrieben werden. (vgl. Honarmand 2017, S. 7) [8]

## 3.2 Limitierungen

Eine wichtige Einschränkung von Hardlinks ist die Beschränkung der Funktionalität auf ein Dateisystem. Dies bedeutet, dass Hardlinks nicht quer über Dateisysteme hinweg erstellt werden können, sondern nur auf einem einzigen Dateisystem funktionieren. [3]

Ferner existiert die Einschränkung, dass „meist" keine Hardlinks auf Verzeichnisse erstellt werden können, obwohl diese selbst auch Dateien sind. (vgl. Honarmand 2017, S. 4) [8] Der Grund liegt dabei in der Bewahrung der Multi-Punkt Notation `..` für übergeordnete Verzeichnisse. Wäre es nämlich möglich, einen Hardlink von einem Unterverzeichnis auf ein übergeordnetes Verzeichnis zu erstellen, so könnten Programme, die das Dateisystem durchlaufen, in eine Endlosschleife geraten. (vgl. Bach 1986, S. 128) [1]

# 4 Softlink

Softlinks sind unter Windows meist unter dem Begriff „Shortcut" bekannt. [9] Dabei handelt es sich lediglich um eine Textdatei, welche als symbolische Verknüpfung markiert ist und dabei einen Pfad zu einer anderen Datei oder einem anderen Verzeichnis enthält. Durch diese Art der Verknüpfung kann einerseits Speicherplatz gespart werden [10], ein Löschen der verlinkten Datei führt andererseits aber auch dazu, dass der Softlink ins Leere läuft. (vgl. Sorin 2016, S. 20) [4]

Ein Feature von Softlinks, welches dessen Pendant nicht unterstützt, ist die Möglichkeit, dateisystemübergreifende Verknüpfungen zu erstellen. Dies erlaubt es nun, auf dem Laufwerk `D:\` einen Shortcut für ein Programm zu definieren, das eigentlich auf dem Laufwerk `C:\` installiert ist. (vgl. ebd.)

Weiterhin ist es auch möglich, große Verzeichnispfade in einem kurzen Pfad zusammenzufassen, was dem Nutzer Zeit bei der Eingabe ersparen kann. (vgl. ebd.)

# 5 Fazit

Sowohl Hard-/ als auch Softlinks haben gegenseitige Vor-/ und Nachteile. Soll beispielsweise eine laufwerkübergreifende Verlinkung erstellt werden, so ist die Generierung eines Softlink erforderlich. Wenn es allerdings wichtig ist, dass eine Verknüpfung auch dann noch funktioniert, wenn eine andere Datei gelöscht wird, so sind wiederrum Hardlinks zu empfehlen. Somit muss der Nutzer also immer selbst entscheiden, welche Möglichkeit er zum gegenwärtigen Zeitpunkt als sinnvoll erachtet.

## Glossar

- Hardlink: Verknüpfung zwischen Dateiname und eigentlicher Datei auf der Festplatte, die selbst über die Inode referenziert wird. In der Fachsprache kommt auch der Begriff „harter Link" zum Einsatz. [11]
- Softlink: Verknüpfung in einem Dateisystem, die auf eine andere Datei oder ein anderes Verzeichnis verweist. In der Fachsprache existieren hierfür auch die Bezeichnungen „symbolischer Link", „symbolische Verknüpfung" und „Symlink". [10]
- Shortcut: Möglichkeit, ein Programm, Verzeichnis oder eine Datei in einem anderen Verzeichnis mittels symbolischer Verknüpfung aufzurufen.
- Multi-Punkt-Notation: Kommt zum Einsatz, wenn über die Kommandozeile in ein höheres Verzeichnis gewechselt werden soll.

## Referenzen


[1] Bach, Maurice J. (1986): The design of the UNIX operating system. Englewood Cliffs, N.J.: Prentice/Hall.

[2] N.A. (2014): Difference between hardlink and softlink in Rational ClearCase. IBM. Online verfügbar unter https://www.ibm.com/support/pages/difference-between-hardlink-and-softlink-rational-clearcase, zuletzt aktualisiert am 19.02.2019, zuletzt geprüft am 04.08.2020.

[3] Megida, Dillion (2020): Symlink Tutorial in Linux – How to Create and Remove a Symbolic Link. In: *freeCodeCamp.org*, 02.05.2020. Online verfügbar unter https://www.freecodecamp.org/news/symlink-tutorial-in-linux-how-to-create-and-remove-a-symbolic-link/, zuletzt geprüft am 04.08.2020.

[4] Sorin, Daniel J. (2016): ECE590-03 Enterprise Storage Architecture Fall 2016. Online verfügbar unter http://people.duke.edu/~tkb13/courses/ece590-2016fa/slides/08-filesystems.pdf, zuletzt geprüft am 16.07.2020.

[5] Both, David (2017): A user's guide to links in the Linux filesystem. Learn how to use links, which make tasks easier by providing access to files from multiple locations in the Linux filesystem directory tree. Hg. v. OpenSource.com. Online verfügbar unter https://opensource.com/article/17/6/linking-linux-filesystem, zuletzt aktualisiert am 22.06.2017, zuletzt geprüft am 04.08.2020.

[6] Both, David (2017): An introduction to Linux's EXT4 filesystem. Take a walk through EXT4's history, features, and optimal use, and learn how it differs from previous iterations of the EXT filesystem. OpenSource.com. Online verfügbar unter https://opensource.com/article/17/5/introduction-ext4-filesystem, zuletzt aktualisiert am 25.05.2017, zuletzt geprüft am 04.08.2020.

[7] Brown, Christopher; Crabbe, Fredrick; Stahl, David (2010): Class 9: The Unix Filesystem. Online verfügbar unter https://www.usna.edu/Users/cs/wcbrown/courses/IC221/classes/L09/Class.html , zuletzt geprüft am 04.08.2020.

[8] Honarmand, Nima (2017): File Systems Basics. Online verfügbar unter https://compas.cs.stonybrook.edu/~nhonarmand/courses/fa17/cse306/slides/16-fs_basics.pdf, zuletzt geprüft am 04.08.2020.

[9] Zick, Thomas (2018): Was ist ein Shortcut? Einfach erklärt. Hg. v. CHIP Digital GmbH. Online verfügbar unter https://praxistipps.chip.de/was-ist-ein-shortcut-einfach-erklaert_41898, zuletzt aktualisiert am 08.09.2018, zuletzt geprüft am 04.08.2020.

[10] Durr, Yosef (2016): Symlinks in Windows 10! - Windows Developer Blog. Hg. v. Microsoft Corporation. Online verfügbar unter https://blogs.windows.com/windowsdeveloper/2016/12/02/symlinks-windows-10/, zuletzt aktualisiert am 02.12.2016, zuletzt geprüft am 04.08.2020.

[11] Wikipedia (Hg.) (2020): Harter Link. Online verfügbar unter https://w.wiki/WvW, zuletzt aktualisiert am 28.04.2020, zuletzt geprüft am 04.08.2020.


## Abbildungsverzeichnis