FUSE – Dateisysteme

Eine schriftliche Ausarbeitung von Jens Spiekermann
im Rahmen des Proseminars "Speicher- und Dateisysteme"

Jens Spiekermann
Proseminar: "Speicher- und Dateisysteme"
Thema: "FUSE-Dateisysteme"

Inhaltsverzeichnis

Jens Spiekermann
Proseminar: "Speicher- und Dateisysteme"
Thema: "FUSE-Dateisysteme"

1. Einleitung

Der Inhalt der folgenden Ausarbeitung befasst sich mit dem Thema Filesystem in Userspace. Filesystem in Userspace (auch kurz: FUSE) ist ein Modul für Unix-Systeme, welches dem Benutzer erlaubt, eigene Dateisysteme einzubinden. Dabei handelt es sich um eine Vielzahl verschiedener einbindbarer Dateisysteme wie beispielsweise Festplatten oder Laufwerke. Eine weitere Besonderheit des Prinzips ist die Tatsache, dass die Einbindung auch von unprivilegierten Benutzern vorgenommen werden kann. Eingebundene Dateisysteme können dann wie ein übliches navigierbares Dateisystem benutzt werden.

Das Ziel, welches diese Ausarbeitung erfüllen soll, ist, dass der Leser auch ohne Vorkenntnisse die Grundlagen, Funktionsweise und die Möglichkeiten der Einbindung von Dateisystemen verstehen soll. Im Folgenden soll der Aufbau der Arbeit vorgestellt und erläutert werden.

2. Aufbau

Um das Prinzip von Filesystem in Userspace auch Lesern ohne Vorkenntnissen verständlich zu machen, sollen an erster Stelle in Abschnitt 3 die Grundlagen erklärt werden, die zum Verstehen des Themas relevant sind. Diese umfassen Prinzipien und Funktionsweisen der Dateisysteme, genauere Erklärung des Aufbaus des Betriebssystems, sowie das Erläutern von Benutzerrechthandhabung unter Unix.

Nachdem die Grundlagen erläutert wurden, werden in Abschnitt 4 zuerst allgemeine Daten zu FUSE erläutert, welche die Entstehung und Betriebssystemunterstützung umfassen. Darauf erfolgt eine Darstellung der Funktionsweise von FUSE, die auf die Grundlagen über den Betriebssystemkern aufbaut.

In Verbindung zu der Erklärung der Vorgehensweise werden in Abschnitt 5 die Eigenschaften von Filesystem in Userspace erläutert. Dies umfasst die Darstellung der Vor- und Nachteile, die durch die Nutzung von FUSE entstehen.

Die einbindbaren Dateisysteme lassen sich ihrer Funktions- und Nutzungsweise entsprechend kategorisieren. In Abschnitt 6 werden einige der Kategorien aufgezählt und erläutert. Darauf folgend werden diverse Dateisysteme genauer erläutert. Zum Abschluss wird beispielhaft die Einbindung des SSHFS-Dateisystems anhand von Linux-Konsolenbefehlen erfolgen.

3. Grundlagen

3.1 Dateisysteme

Inhaltlich zusammenhängende Daten, die zu einer einzelnen Einheit zusammengefasst wurden, werden als Dateien bezeichnet. Dies können zum Beispiel Textdateien, Audiodateien, Videodateien oder Bilddateien sein. Diese Daten müssen auf einem Datenträger abgelegt und gespeichert werden können. Dabei muss beachtet werden, die Dateien auch über einen längeren Zeitraum speichern zu können. Zusätzlich können Dateien Metadaten besitzen, die genauere Angaben über die Dateien, wie zum Beispiel Dateigröße oder Zugriffsrechte, beinhalten. Auch muss ermöglicht werden, dass zuvor abgelegte Dateien an genau dem Ort, an dem sie abgelegt wurden, wiedergefunden werden können. Es ist also ein System notwendig, mit welchem man diese Anforderungen effizient erfüllen und Daten verwalten kann.

Dateisysteme sorgen dafür, dass genau diese Anforderungen in die Tat umgesetzt werden. Sie regeln die Ablage von Dateien auf einem Datenträger und legen die Operationen fest, die an den Dateien ausgeführt werden können. Die wesentliche Kernaufgabe von Dateisystemen besteht im Speichern und Wiederfinden von Dateien. Verzeichnisstrukturen sind die Gestalt von Dateisystemen, in ihnen können von Benutzern Dateien in beliebigen Pfaden ablegen. Dateisysteme befinden sich im Betriebssystemkern und unterscheiden sich je nach Betriebssystem. So verwendet Windows zum Beispiel NTFS-Dateisysteme, Linux das ext3-Dateisystem und OpenSolaris das ZFS-Dateisystem. Die Struktur, die die Verzeich-nisse aufweisen unterscheidet sich ebenfalls je nach Betriebssystem und lässt sich anhand der nachfolgenden Abbildung aufzeigen.

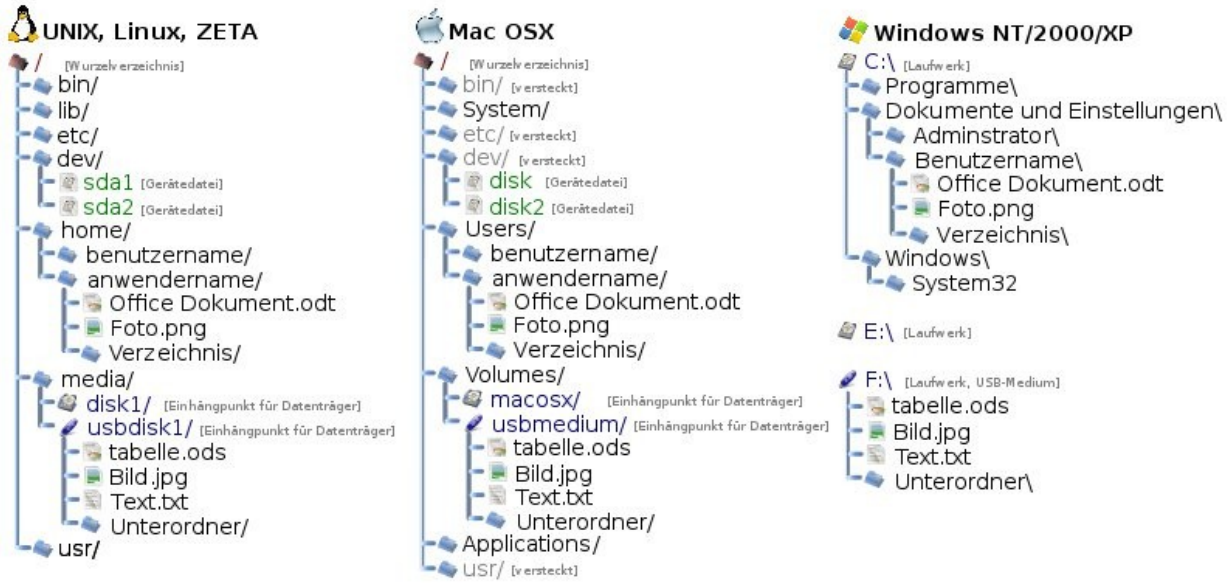

Abbildung 1: Verzeichnisstrukturen

Die Abbildung zeigt verschiedene Verzeichnisstrukturen in Abhängigkeit vom Betriebssystem. Der Aufbau der Verzeichnisse erfolgt in Baumstruktur, so können Ordner weitere Unterordner und/oder Dateien enthalten. Der Unterschied zwischen Windows- und Unix-Dateisystemen besteht darin, dass es unter Windows je nach Laufwerkanzahl mehrere Wurzelknoten in der Verzeichnisstruktur geben kann. In Unix-Dateisystemen können Laufwerke nur an bestimmten Stellen im Baum eingebunden werden.

3.2 Betriebssystemkern

Es gibt verschiedene Arten von Betriebssystemkernen, im Folgenden wird nur auf den monolithischen Betriebssystemkern eingegangen.
Der Aufbau von Betriebssystemen lässt sich in Benutzermodus und Betriebssystemkern unterteilen. Benutzer agieren im Benutzermodus und können selber nicht auf den Betriebssystemkern zugreifen. Im Benutzermodus befinden sich Anwendungen, die der Benutzer des Computers selbst ausführen kann. Dies sind beispielsweise Anwendungsprogramme wie Office. Zusätzlich sind im Benutzermodus Librarys (Bibliotheken) enthalten, die von Unix-Systemen benötigt werden, um mit dem Betriebssystemkern zu kommunizieren. Der Betriebssystemkern beinhaltet Treiber und Dateisysteme und besitzt, anders als der Benutzermodus, direkten Zugriff auf die Hardware des Rechners. Die Kommunikation der beiden Modi erfolgt beispielsweise durch Systemaufrufe, die von Anwendungsprogrammen verursacht werden. Die folgende Abbildung des Aufbaus von Unix-Betriebssystemen zeigt die Inhalte der beiden Modi anschaulich.

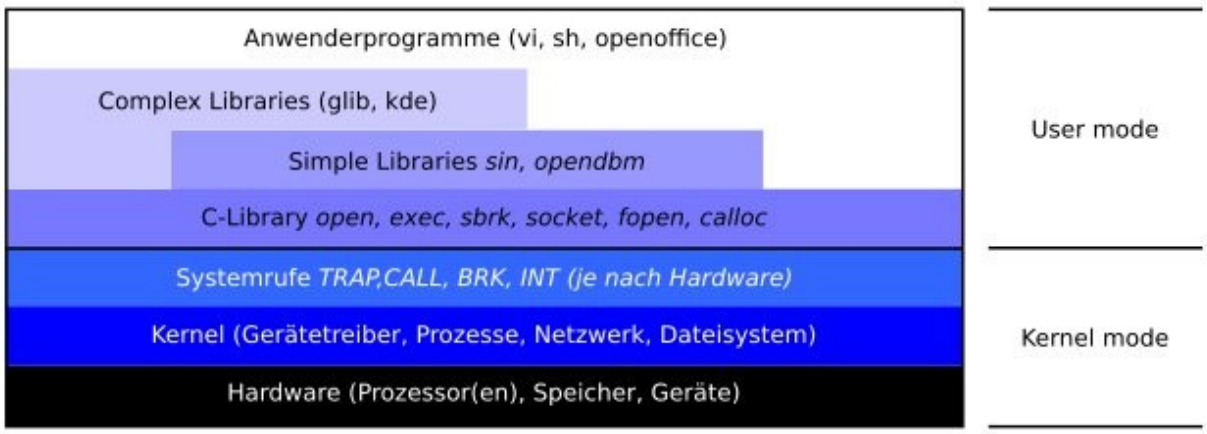

Abbildung 2: Betriebssystemschichten

Generell lässt sich über den Betriebssystemkern sagen, dass er als Schnittstelle zwischen dem Benutzermodus und der eigentlichen Hardware fungiert.

3.3 Benutzerrechte

Eine Eigenschaft von Filesystem in Userspace ist, dass unprivilegierte Benutzer ihre eigenen Dateisysteme einbinden können. Dafür soll kurz die Benutzerrechthandhabung unter Unix-Systemen erklärt werden.
Benutzerrechte im Mehrbenutzerbetrieb lassen sich umsetzten, indem man den Benutzern eines Computers eigene Konten zuweist und diesen Rechte gewährt. Unter Linux gibt es einen Administrator ("Root"), der in der Lage ist, anderen Benutzerkonten Rechte zu gewähren oder zu entziehen. Wenn ein Benutzer den Computer benutzt, muss er immer unter seinem Benutzernamen angemeldet sein, damit das System zu jedem Zeitpunkt weiß, wer das System benutzt. Jeder Benutzer besitzt ein Heimatverzeichnis, in dem er, je nach Rechten, die ihm zugewiesen worden sind, arbeiten kann. Die Heimatverzeichnisse sind unabhängig voneinander, wenn beispielsweise ein Benutzer in seinem Heimatverzeichnis ein Programm installiert, hat dies keine Auswirkungen auf andere Benutzer und deren Verzeichnisse. In Bezug auf FUSE bedeutet dies, dass Benutzer FUSE nutzen können, ohne selber Administratorrechte zu besitzen.

4. Filesystem in Userspace

4.1 Allgemeines

Filesystem in Userspace war ursprünglich Teil von AVFS (A Virtual Filesystem), welches als Zielsetzung besaß, ausgewählten Anwendungsprogrammen Zugriff auf Dateien, die sich in Archiven befinden, zu ermöglichen. Seit dem 15.10.2004 ist FUSE allerdings ein eigenes Projekt auf sourceforge.net. Das Betriebsssystemkernmodul von FUSE wurde am 13.09.2005 in den Linux-Kern aufgenommen und ist in C implementiert.
Prinzipiell unterstützen alle unix-ähnlichen Betriebssysteme FUSE. Die wohl bekanntesten Vertreter sind Linux, welches als Open Source frei erhältlich ist, und MacOSX, welches das Betriebssystem für die von Apple produzierten Macintosh-Computer darstellt. Außerdem ist die Verwendung von Filesystem in Userspace unter Oracles (früher SUN Microsystems) OpenSolaris möglich. Als weitere Möglichkeiten lassen sich FreeBSD und NetBSD nennen, die beide ihren Ursprung im Betriebssystem Berkeley Software Distribution haben.

4.2 Funktionsweise

Filesystem in Userspace lässt sich in zwei Bestandteile unterteilen. Zum Einen in die Userspace-Bibliothek, die sich im Benutzermodus befindet, zum Anderen in das Kernelmodul, welches sich im Betriebssystemkern befindet. Wenn ein Benutzer im Benutzermodus eine Operation FUSE betreffend ausführt, greift die Anwendung auf das Kernelmodul von FUSE zu, welches wiederum bewirkt, dass der Aufruf zurück in den Benutzermodus umgeleitet wird. Dies geschieht, indem das Kernelmodul mit der Userspace-Bibliothek kommuniziert. Dadurch, dass der Aufruf wieder in den Benutzermodus umgeleitet wurde, spielt sich auch die gesamte Logik des verwendeten FUSE-Dateisystems im Benutzermodus ab.

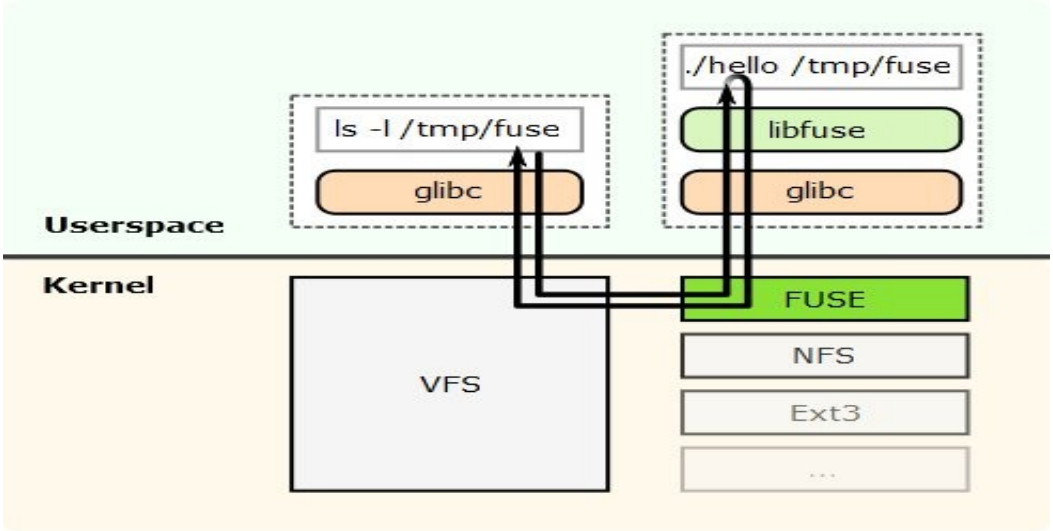

Abbildung 3: FUSE Funktionsweise

Anhand von Abbildung 3 lässt sich die Funktionsweise sowie das Prinzip der Aufrufumleitung bildlich darstellen. Sie stellt dar, wie der Aufruf aus dem Benutzermodus durch das Kernelmodul wieder in den Benutzermodus geleitet wird, damit dort die eigentliche Programmlogik stattfinden kann.

5. Eigenschaften

5.1 Vorteile

Ein großer Vorteil, den Filesystem in Userspace bietet, ist die Vielseitigkeit der Verwendungsmöglichkeiten. Durch das in Abschnitt 4 erläuterte Prinzip ist es möglich, eine große Vielfalt an verschiedenen Dateisystemen einzubinden, die ebenfalls eine Vielzahl an verschiedenen Verwendungszwecken bieten.
Die Tatsache, dass Filesystem in Userspace von allen unix-ähnlichen Betriebssystemen unterstützt wird, führt zu einer hohen Portabilität. Man ist so in der Lage, von zwei unterschiedlichen Betriebssystemen auf das selbe Dateisystem zuzugreifen und es zu benutzen.
Für den Mehrbenutzerbetrieb an einem Computer ergibt sich der Vorteil, dass Benutzern, die keine Administratorrechte an einem Rechner besitzen, die Möglichkeit geboten wird, FUSE zu nutzen.

FUSE ist als Open Source frei erhältlich. Dies bewirkt zum Einen, dass der Nutzer FUSE kostenlos erhalten und benutzen kann, zum Anderen, dass sich FUSE leicht erweitern lässt.
Ebenfalls ein Grund für die leichte Erweiterbarkeit stellt die einfache Programmierbarkeit dar. So können FUSE-Dateisysteme beispielsweise in den Sprachen Java oder Python geschrieben werden.

5.2 Nachteile

Als einzigen großen Nachteil für Filesystem in Userspace lässt sich das Leistungsproblem nennen, welches bei der Benutzung entsteht. Dieses wird im Folgenden näher erläutert.
Ein Grund für das Leistungsproblem stellt die Geschwindigkeit der Verbindung von Netzwerk und Internet dar. Sollten Dateisysteme eingebunden werden, die über Netzwerk oder Internet benutzt werden können, kann die Geschwindigkeit der Datenübertragung nur so groß sein, wie es die Verbindung zulässt.
Die Kontextwechsel stellen einen weiteren Grund für das Leistungsproblem von FUSE dar. Generell tritt ein Kontextwechsel immer dann auf, wenn die Bearbeitung eines Prozesses unterbrochen wird um einen anderen Prozess fortzusetzen. Die Rechenzeit, die ein Kontextwechsel in Anspruch nimmt hängt vom verwendeten Prozessor, dem Cache und der eigentlichen Zugriffsgröße ab, so werden pro Kontextwechsel einige Tausend Nanosekunden benötigt. Bei der Verwendung von FUSE treten Kontextwechsel dann auf, wenn in den Dienst, der das eingebundene Dateisystem zur Verfügung stellt, gewechselt wird.
Zusätzlich zu den Kontextwechseln kosten auch Moduswechsel eine gewisse Anzahl an Rechenzeit. Moduswechsel werden verursacht, wenn zwischen Benutzermodus und Betriebssystemkern gewechselt wird. Jeder Aufruf, der durch FUSE verursacht wird, führt zu einem Moduswechsel und gegebenenfalls einem Kontextwechsel.

6 Einbindbare Dateisysteme

6.1 Kategorien

Die Dateisysteme, die mittels Filesystem in Userspace eingebunden werden können, lassen sich nach ihrer Verwendungsart kategorisieren. An dieser Stelle werden beispielhaft sechs verschiedene Kategorien aufgezeigt. Zu beachten ist allerdings, dass diese nur Beispiele darstellen und es noch weitere Kategorien gibt.

Archiv-Dateisysteme haben zur Aufgabe, es Programmen zu ermöglichen auf Dateien zuzugreifen, die sich in Archiven wie etwa zip oder tar befinden.

Datenbank-Dateisysteme sollen die Ablage von Dateien oder Metadaten in relationalen Datenbanken ermöglichen, wodurch die Suche nach Dateien über Tags oder SQL-Abfragen ablaufen kann.

Medien-Dateisysteme ermöglichen den Zugriff auf Mediengeräte, wie zum Beispiel Mobiltelefone oder Digitalkameras. Diese können dann wie ein Massenspeicher verwendet werden.

Filesystem in Userspace ermöglicht außerdem den Zugriff auf nicht ursprüngliche Dateisysteme. Es ist dadurch möglich unter Linux auf Dateisysteme zuzugreifen, die keine Linux-Standards sind. (NTFS)

Verschlüsselte Dateisysteme ermöglichen die sichere Dateilagerung durch entweder Verschlüsselung des kompletten Dateisystems, oder Verschlüsselung einzelner Dateien in einem Dateisystem.

Über Netzwerk-Dateisysteme lassen sich auf Dateisysteme zugreifen, die sich auf entfernten Rechnern befinden. Die Verbindung der beiden Rechner erfolgt dann beispielsweise über SSH.

6.2 Beispiele

In diesem Abschnitt werden drei Dateisysteme , die sich mit Hilfe von FUSE einbinden lassen, etwas näher beschrieben.

NTFS
NTFS ermöglicht es, dass ein nicht ursprüngliches Dateisystem eingebunden werden kann. In diesem Fall ist es das New Technology File System, das durch FUSE auch auf unix-ähnlichen Betriebssystemen benutzt werden kann. Dies wird durch Benutzung des quelloffenen NTFS-3G Treiber realisiert, wodurch Lese- und Schreibzugriff auf das Dateisystem ermöglicht wird. Alternativ funktioniert das Dateisystem mit Captive NTFS, welches auf den Originaltreibern von Windows XP aufbaut. Schreib- und Leseunterstützung werden durch diese Variante ebenfalls unterstützt.

SSHFS
SSHFS ist ein typischer Vertreter der Netzwerkdateisysteme. Das Prinzip basiert auf der Verwendung des Secure Shell File Transfer Protocol, durch das eine sichere und verschlüsselte Verbindung zwischen zwei Rechnern erstellt werden kann. Dadurch ist es möglich, von einem Computer über Netzwerk oder Internet Dateipfade einzubinden, die sich auf entfernten Rechner befinden. Wie genau die Einbindung funktioniert wird im letzten Abschnitt kurz behandelt.

GMailFS
Ähnlich wie SSHFS handelt es sich bei GMailFS ebenfalls um ein Netzwerkdateisystem. Es verwendet einen GMail-Konto als Speichermedium, das etwa 7 Gigabyte an zusätzlichem Speicherplatz bieten kann. Dateien werden durch dieses Prinzip in den Anlagen von E-Mails gespeichert, die Dateinamen finden sich im Betreff der entsprechenden E-Mails wieder. Allerdings wurde die Entwicklung von GMailFS eingestellt, da es gegen die Benutzerbedingungen von Google verstößt, die die automatische Benutzung von GMail-Konten untersagen.

6.3 SSHFS Einbindung

Um SSHFS verwenden zu können, müssen einige Befehle in der Linux-Konsole ausgeführt werden. Als Voraussetzung gilt, die SSHFS Dateien heruntergeladen und installiert zu haben. Im Anschluss daran sollte man als Root einen Benutzer zur FUSE-Gruppe hinzufügen, damit dieser auch FUSE verwenden kann.

"$ usermod -a -G fuse klaus"

Fügt den Benutzer Klaus zur Gruppe FUSE hinzu und erlaubt ihm somit FUSE zu nutzen.

"$ mkdir sshfstest"

Erstellt ein neues, leeres Verzeichnis.

"$ sshfs host:/path ~/localpath"

Bezeichnet den Host und dessen Dateipfad, der dann im lokalen Pfad eingebunden wird. Es könnte zum Beispiel das zuvor neu erstellte Verzeichnis als Pfad benutzt werden. Wenn man das Dateisystem wieder ausgehangen werden soll, kann das mit dem folgenden Befehl umgesetzt werden.

"$ fusermount -u ~/localpath"

Quellen

Abb. 1: http://upload.wikimedia.org/wikipedia/de/1/1f/Filesystem.svg
Abb. 2: http://upload.wikimedia.org/wikipedia/commons/b/b3/Linux_Schichten.svg
Abb. 3: http://upload.wikimedia.org/wikipedia/commons/0/08/FUSE_structure.svg
http://en.wikipedia.org/
http://de.wikipedia.org/
http://www.selflinux.org
http://fuse.sourceforge.net/
http://www.linfo.org/context_switch.html
http://blog.tsunanet.net/2010/11/how-long-does-it-take-to-make-context.html
http://www.techrepublic.com/blog/opensource/use-fuse-To-mount-remote-sshdirectories/225
http://www.cs.rochester.edu/u/cli/research/switch.pdf
http://www.linuxjournal.com/article/8904