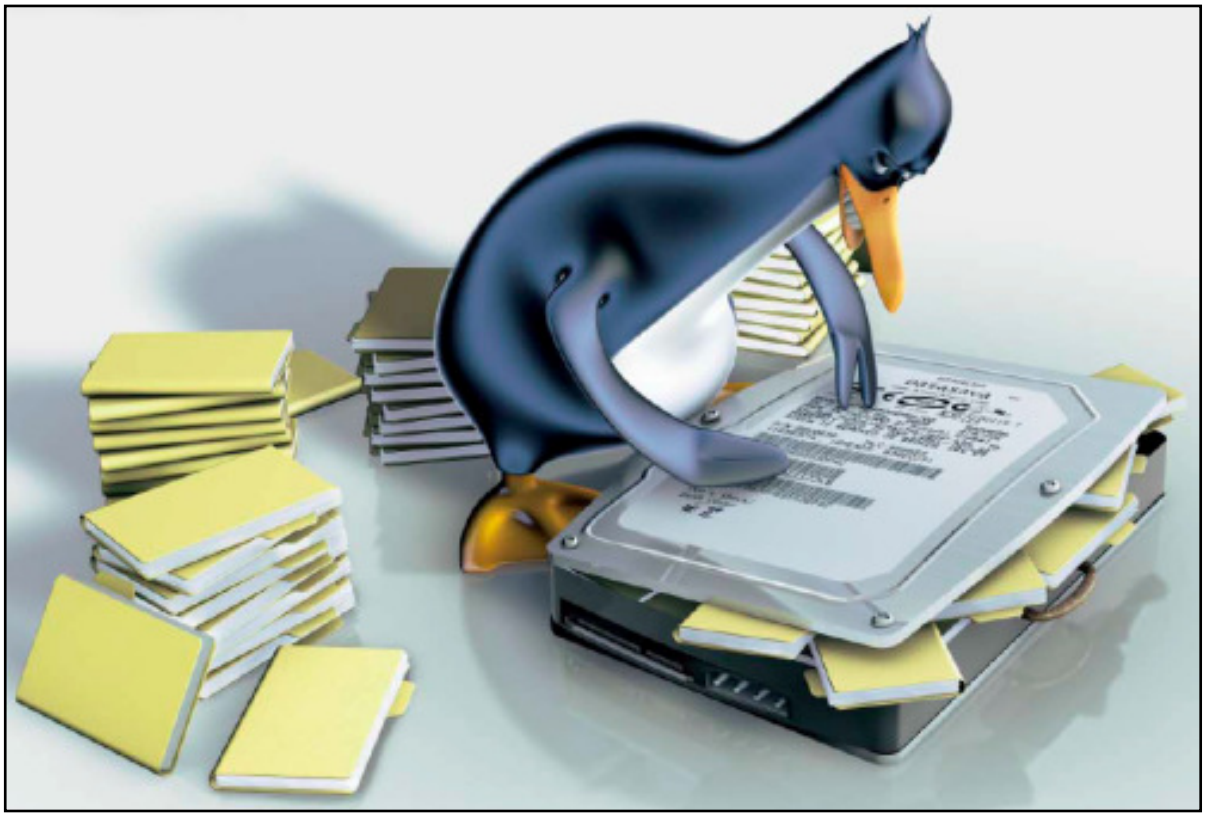
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Grundlagen der Dateisysteme



Daniel Lieck
Wintersemester 2010/2011

Ausarbeitung Grundlagen der Dateisysteme
Proseminar Speicher- und Dateisysteme

Inhalt

1 Einführung

Diese Ausarbeitung befasst sich im Rahmen des Proseminars "Speicher- und Dateisysteme" mit den Grundlagen von Dateisystemen. Nach der Einführung in das Thema und einem kurzen geschichtlichen Überblick, wird näher auf Arten, Struktur sowie Zugriff auf Dateisysteme eingegangen.

1.1 Dateisysteme - wofür eigentlich?

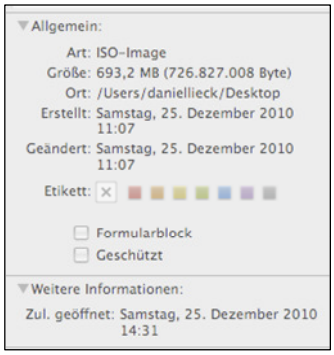
Wir alle öffnen, ändern, erstellen oder löschen täglich Dateien mit dem Rechner. Diese Dateien müssen strukturiert auf einem Datenträger abgelegt und eindeutig identifiziert werden können. Hier kommt das Dateisystem zum Einsatz. Es sorgt dafür, dass jede Datei leicht abgespeichert und wiedergefunden werden kann.

Das Dateisystem bildet also eine Art Ordnungs- und Zugriffssystem. Die Eindeutigkeit einer Datei wird durch Dateiname und Dateipfad sichergestellt (deshalb ist es auch z.B. nicht möglich eine Datei in einem Verzeichnis mit einem Namen zu erstellen welchen es dort bereits gibt). Es gibt noch weitere Informationen (Attribute) über jede einzelne Datei, die das Dateisystem verwaltet, sogenannte Meta-Daten (siehe 1.2).

Für den Menschen ist es wichtig, dass Dateiname und die computerinterne Dateiadresse (physische Adresse, Blocknummer, Spur, Sektor usw.) in Einklang gebracht werden, das Dateisystem muss also dafür sorgen, dass beispielsweise bei einem Klick auf eine bestimmte Datei die richtige physische Adresse auf der Festplatte angesprochen wird und so die richtigen Daten zurückgegeben werden können. Mit Hilfe von Verzeichnissen organisiert das Dateisystem die einzelnen Daten und macht sie so leicht zugänglich.

1.2 Meta-Daten

Alle Daten die zusätzliche Informationen zu einer bestimmten Datei darstellen werden als Meta-Daten bezeichnet. Dateien haben in einem Dateisystem immer mindestens einen Dateinamen sowie Attribute die nähere Informationen zu dieser Datei beinhalten. Die Dateinamen werden wiederum in Verzeichnissen abgelegt, mit denen dann eine Datei vom System gefunden werden kann.



Von Tim Berners-Lee, Direktor des World Wide Web Consortiums (W3C), stammt die Definition: "Metadaten sind maschinenlesbare Informationen über elektronische Ressourcen oder andere Dinge."

Ein typisches Beispiel für Metadaten sind zusätzliche Informationen zu einem Buch wie etwa der Autor oder das Erscheinungsjahr. Durch Metadaten versucht man so "Daten zu den Daten" elektronisch zu speichern was früher teilweise manuell erfasst wurde (z.B. in Bibliotheksystemen die entsprechende Informationen zu Büchern gesammelt haben).

2 Historie

2.1 Die ersten Dateisysteme

Die ersten Dateisysteme stellten als lineare Dateisysteme die Lochkarten Mitte des 18. Jahrhunderts dar. Sie wurden dazu benutzt um wiederkehrende Arbeitsabläufe rationell zu wiederholen. Verwendet wurden sie u.a. in Webstühlen.

Ein erstes richtiges Dateisystem verwaltet durch ein Betriebssystem entstand dann 1964 mit den DECTapes, ein Magnetbandsystem mit dessen Hilfe Daten (organisiert in Datenblöcke) auf Band geschrieben bzw. gelesen werden konnten. In diesen ersten Dateisystemen gab es weder eine Rechteverwaltung, noch eine Strukturierung der Daten in Verzeichnisse wie wir es heute kennen.

Ein erstes Betriebs- und Dateisystem welches diese Funktionen unterstützte wurde 1974 entwickelt (CP/M – Control Program for Microcomputers). Es war über lange Zeit sehr erfolgreich und stellte den Vorreiter für das noch heute bekannte Dateisysteme FAT. CP/M unterstützte sowohl eine Rechtevergabe als auch das Anlegen von Verzeichnissen (allerdings nur in erster Ebene).

2.2 Gründe für die Weiterentwicklung

Dateisysteme gewannen in der Entwicklung immer mehr an Bedeutung. Mit der rasanten Entwicklung neuer, großer und schneller Speichermedien und Zugriffsmöglichkeiten stieg das Bedürfnis die Daten effizient zu organisieren. Es waren intelligente Werkzeuge zum Speichern und Verwalten der Daten erforderlich.

Zusätzlich zwangen Patentrechte, also Schutzrechte für selbst entwickelte Dateisysteme, die Dateisystementwickler immer wieder dazu neue Lösungen zu entwickeln. Firmen beantragten Patente auf ihre neuartigen Dateisysteme und verlangten entsprechende Zahlungen bei dessen Nutzung in technischen Geräten. Seit 2004 verlangt Microsoft 0,25 $ pro hergestelltes Gerät für die Nutzung des FAT-Dateisystems (maximal 250.000 $ von einem Hersteller). War anfangs die Zahl unterschiedlicher Dateisysteme noch überschaubar, so gibt es heute weit über 100.

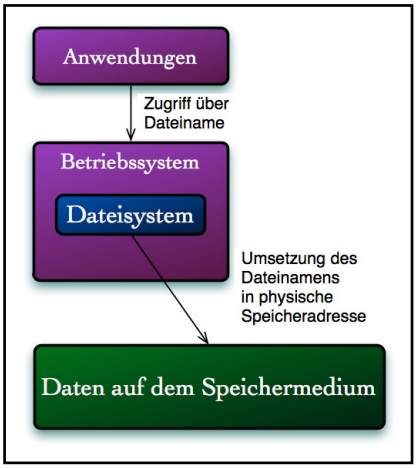
Auch der technische Fortschritt im gesamten IT-Bereich (Globalisierung etc.) sorgt immer wieder für eine Entwicklung der Dateisysteme, so wurden neben den klassischen lokalen Dateisystemen auch Netzwerkdateisysteme und virtuelle Dateisysteme entwickelt (siehe 4. Arten der Dateisysteme).

3 Dateisystem-Struktur und Organisation

Die Struktur des Dateisystems unter Linux unterscheidet sich gegenüber anderen Einbenutzersystemen. Linux kennt keine Laufwerke, die Struktur besteht aus einem Verzeichnisbaum der beim Root-Verzeichnis beginnt (‚/'). Laufwerke wie Festplatten, CD-ROM usw. können jetzt an geeigneter Stelle angebunden werden (mount). Für den Nutzer ist dabei ohne weiteres nicht ersichtlich, ob die zugebundenen Laufwerke lokal oder per Netzwerk eingebunden sind.

Unter Windows hat sich im Laufe der Zeit wenig geändert: Die Systemdateien liegen standardmäßig unter *C:\Windows* und Benutzerverzeichnisse unter *C:\Dokumente und Einstellungen\User\Eigene Dateien*. Laufwerke bekommen jeweils einen festen Laufwerkbuchstaben zugewiesen.
Beim Macintosh gibt es u.a. die Verzeichnisse */Benutzer*, */System*, */Library* in denen Benutzerinformationen, Systemdaten sowie die Bibliotheken zu finden sind.

3.1 Organisation des Dateisystems



Ein Dateisystem kann als Schicht zwischen Betriebssystem (Windows, Mac, Unix) und den Anwendungsprogrammen gesehen werden. Das Betriebssystem und die Anwendungen greifen per Klartextnamen (also dem Dateinamen) auf Dateien zu, erst im Dateisystem wird dieser (abstrakte) Dateiname in eine physische Speicheradresse umgesetzt.

3.2 Datenträgerorganisation

Die Datenträgerorganisation sorgt für die Aufteilung des Sekundär-Speichers (also z.B. der Partitionierung der Festplatte). Eine Festplatte wird dabei in 512 Byte oder 4 Kilo Byte-Blocks aufgeteilt (Blockstrukturierung). Weitere Aufgaben des Dateisystems in dieser Schicht sind die Verwaltung des freien Speicherplatzes, der Umgang mit Fehlern (heutzutage auch oft von den Festplatten selbst übernommen) sowie zusätzliche Funktionen wie z.B. ein RAID-System zur Sicherung der Daten.

3.3 Dateiorganisation

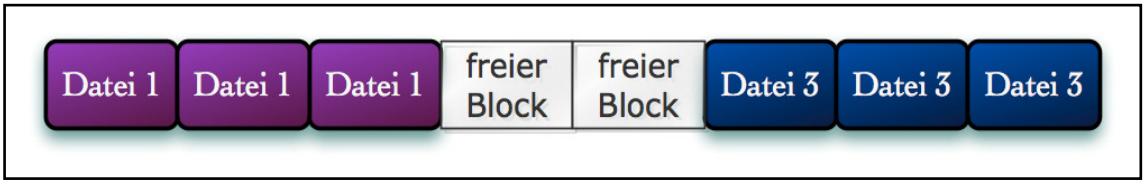
In der Dateiorganisation geht es um die Organisation der logischen Datenblöcke auf der Blockstruktur des sekundären Speichers, also der physischen Verteilung der Daten auf z.B. einer Festplatte und um die Struktur dieser Dateien und Verzeichnisse. Man unterscheidet dabei zwischen der verteilten und der zusammenhängenden Allokation (Verteilung) der Daten:

zusammenhängende Allokation:

Dateien werden in zusammenhängenden Blöcken aufgeteilt, zu jeder Datei müssen also die (Start-)Blocknummer sowie die Anzahl der Blöcke einer Datei bekannt sein.
Durch das ständige Löschen und Hinzufügen von Dateien (und den damit verbundenen Belegungen von Speicherblöcken) kann es also vorkommen, dass bestehende Dateien nicht immer vergrößert werden können, es muss dann ein komplett neuer zusammenhängender Speicherblock mit ausreichend freien Blöcken gewählt werden. Die so "zwischen den Dateien" frei werdenden Speicherblöcke können durch eine externe Defragmentierung "aufgeräumt" werden (z.B. Festplatten-Defragmentierung unter Windows)

Beispielhafte Darstellung einer Speicherstruktur eines Datenträgers mit zusammenhängender Allokation.

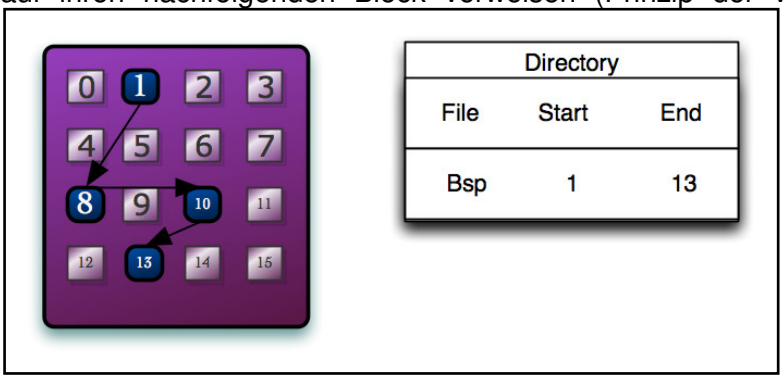
Datei 1 benötigt drei Speicherblöcke, Datei 2 zwei und Datei 3 wiederum drei Speicherblöcke für die jeweiligen Daten. Wenn diese Dateien "nacheinander" angelegt werden entstehen also keine freien (ungenutzten) Speicherblöcke. Es ist jedoch z.B. nicht möglich Datei 1 um eine Anzahl von Speicherblöcken zu vergrößern, es müsste dafür ein komplett neuer Speicherbereich gewählt werden. Die folgende Abbildung zeigt beispielhaft wie die eben beschriebene Struktur nach dem Löschen von Datei 2 aussieht:



Es entstehen jetzt 2 freie Speicherblöcke zwischen Datei 1 und Datei 3. Datei 1 könnte jetzt um diese beiden Speicherblöcke vergrößert werden. Im Rahmen einer Defragmentierung gibt es verschiedene Ansätze diese freien Speicherblöcke "aufzuräumen".

verteilte Allokation:

Bei der verteilten Allokation werden die Daten nicht in zusammenhängende Blöcke aufgeteilt, die Speicherblöcke werden stattdessen "willkürlich vergeben". Eine Möglichkeit Daten zu speichern bildet dabei die Erstellung eines Directorys in dem der Name der Datei, der Start- sowie der End-Block gespeichert werden. Die Datei kann dabei aus mehreren nicht zusammenhängenden Speicherblöcken bestehen, die jeweils auf ihren nachfolgenden Block verweisen (Prinzip der verketteten Liste). Das von Microsoft eingeführte FAT-Dateisystem basiert auf diesem Prinzip. Der Vorteil liegt darin, dass eine externe Defragmentierung jetzt entfällt, da es keine "überflüssigen" Speicherblöcke zwischen Dateien gibt. Auch Dateivergrößerungen sind jetzt problemlos (durch Erweiterung der verketteten Liste und einer Anpassung des End-Blocks im directory) möglich. Ein Nachteil dieses Prinzips ist sicher der Aspekt, dass eine Datei über das gesamte Speichermedium verteilt sein kann und so die Suchzeiten erhöht werden können.



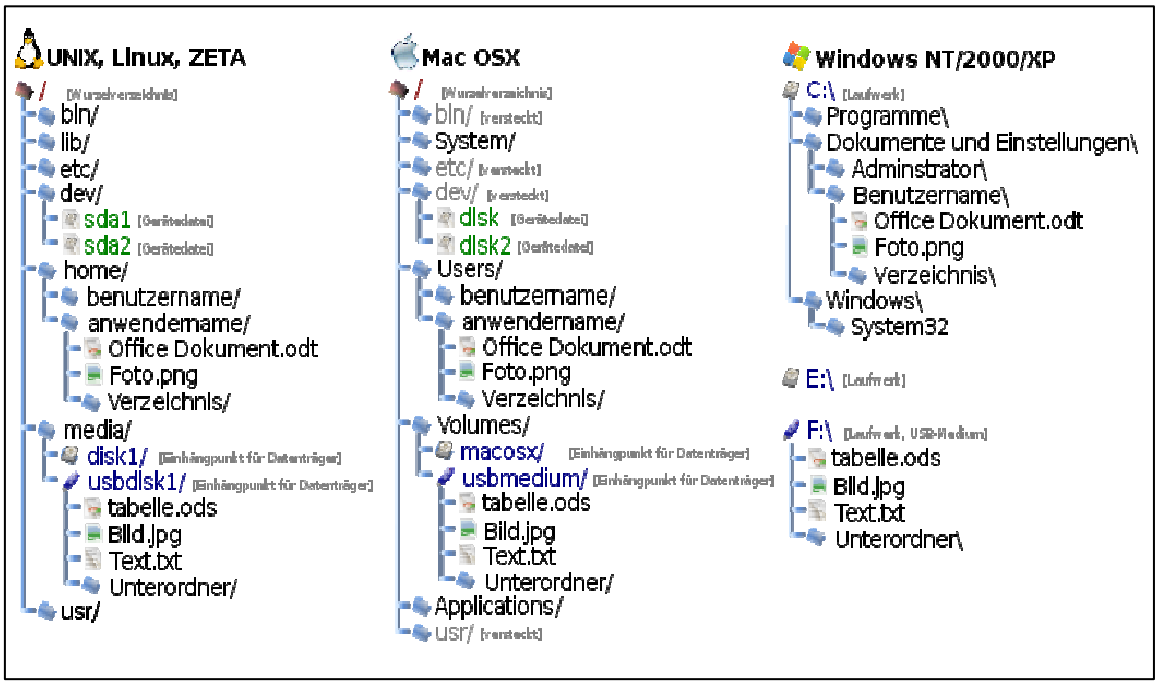
4 Arten der Dateisysteme

Dateisysteme lassen sich in 3 Arten unterteilen, den lokalen Dateisystemen (die auf die jeweiligen Speichermedien wie Festplatten usw. direkt zugreifen, z.B. NTFS unter Windows, HFS unter MAC und Ext3 unter Linux), den Netzwerkdateisystemen und den virtuellen Dateisystemen. Alle 3 Arten können dabei dem hierarchischen Aufbau entsprechen. Im Bereich der Lochkarten und den Magnetspeichersystemen spricht man auch von sogenannten linearen Dateisystemen die jedoch heutzutage kaum noch eine Rolle spielen und deshalb an dieser Stelle nicht weiter ausgeführt werden.

4.1 Hierarchische Aufbau

Der hierarchische Aufbau von Dateisystemen entwickelte sich mit dem Bedürfnis immer mehr Daten möglichst strukturiert abzuspeichern. Früher war es üblich sämtliche Dateien in einem einzigen Verzeichnis zu speichern. Mit zunehmender Speicherkapazität war dies natürlich nicht mehr effizient. Aus diesem Grunde wurden dann sogenannte Unterverzeichnisse (Ordner) eingeführt in denen Daten strukturiert abgelegt werden können. Das ursprüngliche (Haupt-)Verzeichnis wird auch Wurzelverzeichnis genannt. Unter Windows heißt das Wurzelverzeichnis *C:*

In folgender Grafik (Quelle: Wikipedia) sind die Verzeichnisstrukturen der gängigen Betriebssysteme dargestellt:



Bildquelle: http://upload.wikimedia.org/wikipedia/de/1/1f/Filesystem.svg

Die einzelnen Verzeichnisse oder Ordner werden je nach Betriebssystem durch einen Slash oder Backslash voneinander getrennt. Den "Weg" durch verschiedene Ordner zum Zugriff auf eine Datei nennt man Pfad.

4.2 Netzwerkdateisysteme

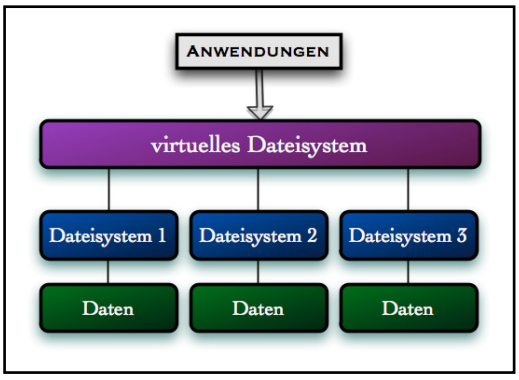
Mit zunehmender Vernetzung der Systeme stieg das Bedürfnis nach einem Zugriff auf die gleiche Datenbasis von verschiedenen Systemen aus. Festplattenkapazität ist teuer, Programme und Daten sollen zentral gespeichert werden und trotzdem für alle relevanten Systeme zugänglich sein. Die Lösung für diese Probleme stellen die Netzwerkdateisysteme dar, also Dateisysteme die über das Netzwerk arbeiten.
In Unix-Systemen ist das gebräuchlichste Protokoll das Network-File-System (NFS), unter Windows kommt oft das Common-Internet-File-System (CIFS) zum Einsatz.

Durch ein Netzwerkdateisystem werden Systemaufrufe (Erstellen, Lesen, Öffnen usw. einer Datei) von einem Client-Rechner aus auf einen Server übertragen, der die Befehle dann entsprechend auf den Massenspeicher ausführt und die Informationen an den Client zurückgibt. Die Befehle bzw. Systemaufrufe sind dieselben wie bei einem lokalen (Nicht-Netzwerk-) Dateisystem wodurch sich der Anwender oft gar nicht bewusst ist, ob er jetzt mit dem lokalen oder einem Netzwerkdateisystem interagiert (transparenter Zugriff).

Um Dateninkonsistenz (z.B. durch gleichzeitiges Überschreiben einer Datei) zu vermeiden werden u. a. sogenannte Metadaten-Server eingesetzt, welche Meta-Daten-Zugriffe von allen Systemen übertragen bekommen (und dadurch aufzeichnen wann welche Datei geschrieben oder überschrieben wird) und dann den Verzeichniszugriff bzw. die Blockierung des Zugriffs mit Hilfe einer verteilten Sperrverwaltung regeln.

4.3 Virtuelle Dateisysteme

Ein virtuelles Dateisystem stellt eine Abstraktionsschicht auf andere konkrete Dateisysteme dar. Es ermöglicht Anwendungen auf unterschiedliche Dateisysteme zuzugreifen und dient als eine Art Schnittstelle zwischen Dateisystemen und dem Betriebssystem. Die eigentlichen Dateisysteme unter dem virtuellen Dateisystem werden so "verdeckt".



Anwendungen können so auf die Daten von unterschiedlichen Dateisystemen zugreifen, so wäre es z.B. möglich, dass in nebenstehender Grafik Dateisystem 1 ein MAC-, Dateisystem 2 ein Unix- und Dateisystem 3 ein Windows-Dateisystem ist. Mit Hilfe des virtuellen Dateisystems als Abstraktionsschicht ist es nun für Anwendungen möglich mit allen 3 Systemen zu kommunizieren. Auch eine Einbindung von Netzwerkdateisystemen ist so möglich.

Die Implementierung eines virtuellen Dateisystems kann im Kernel des Betriebssystems erfolgen (beim PROC-Dateisystem wird beispielsweise eine hierarchische Datenstruktur im Kernel angelegt). Das virtuelle Dateisystem speichert für jede geöffnete Datei einen sogenannten V-Node, in der die Datei beschrieben wird. Außerdem unterhält es eine Mount-Table mit allen eingebundenen Dateisystemen.

5 Verzeichnis-Struktur und Aufbau

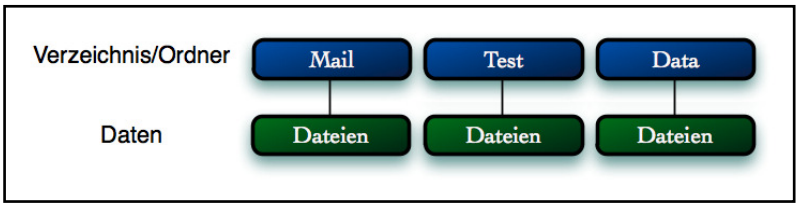
Beim Hochfahren eines Rechners lädt der Kernel als erstes das Root-Dateisystem, in dem alle betriebsnotwendigen Funktionen und Programme enthalten sind. Zu diesen Funktionen zählen Dienstprogramme zum Prüfen und evtl. Reparieren des Dateisystems, zum Sichern und Installieren notwendiger Systemdateien und die Netzwerkprogramme.

Bei den Unmengen an Dateien, die heutige Systeme beinhalten ist eine gute Struktur unabdingbar. Es gibt dabei mehrere Formen wie Dateien und Ordner in einem Dateisystem strukturiert werden können, die im Folgenden kurz vorgestellt werden.

5.1 Ein- und zweistufige Strukturen

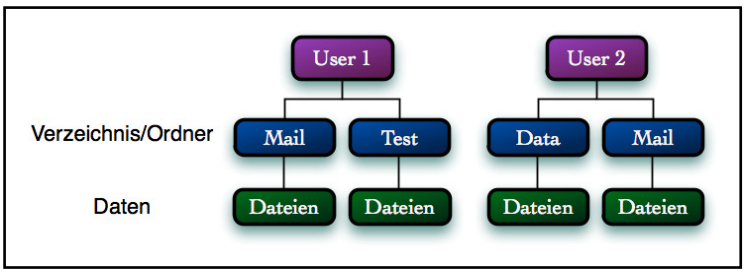
Ein Verzeichnis besteht aus einer Menge von Informationen die benötigt werden um Dateien mit ihrem Namen + Pfad anzusprechen. Neben dem Dateinamen muss ein Eintrag im Dateisystem zu jeder Datei auch Informationen über den Dateityp, dem Zeitpunkt der letzten Änderung, Dateigröße, Eigentümer, Schutzinformationen, Nutzungszähler, Zeitpunkt der Erstellung bzw. letzten Änderung und eine Liste der von der Datei beanspruchten Speicherblöcke auf dem Speichermedium speichern.

Eine einstufige Verzeichnisstruktur (siehe Grafik) bietet nur ein Verzeichnis und damit nur Ordner in erster Ebene in der sich dann alle Dateien befinden. Diese Struktur ist daher besonders für den Einbenutzerbetrieb geeignet.



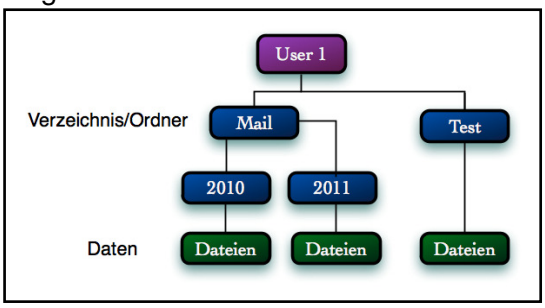
Probleme ergeben sich beim Namensraum und der Gruppierung: Man muss sehr viele Dateien in einem Verzeichnis speichern und für jede Datei einen eindeutigen Namen vergeben. Wenn man in einem Ordner nach einer Datei sucht dessen Name nicht genau bekannt ist, kann das bei einer großen Anzahl an Dateien sehr aufwendig werden.
Heutzutage wird eine einstufige Verzeichnisstruktur in keinem Dateisystem mehr verwendet, weil ein hierarchisches Unterteilungskonzept der Ordner auf großen Datenträgern unerlässlich geworden ist. Das Anfang der 80er Jahre entwickelte Betriebssystem CP/M-80 arbeitete mit einem solchen einstufigen Verzeichnissystem.

In der zweistufigen Verzeichnisstruktur, wird dagegen für jeden Benutzer ein Verzeichnis angelegt. Somit bildet sich der Namensraum aus Benutzername + Dateiname, wodurch diese Struktur bei mehreren Benutzern besser geeignet ist.



5.2 Baumstruktur

Üblich in heutigen modernen Dateisystemen ist die Baumstruktur. Hier ist es möglich beliebig viele Unterverzeichnisse anzulegen, was mehrere Vorteile mit sich bringt. Es ist möglich mehrere unterschiedliche Dateien (in unterschiedlichen Ordnern) mit demselben Namen zu verwenden, da sie über den Pfad eindeutig identifiziert werden können. Außerdem ist jetzt ein effizienteres Suchen in den entsprechenden Ordnern möglich, da die Dateien in der Baumstruktur gruppiert werden können (siehe Grafik, der Ordner Mail ist in den Jahreszahlen 2010 und 2011 unterteilt). Bei einer hohen Anzahl von Unterverzeichnissen kann ein Suchvorgang (durch die Suche in verschiedenen Verzeichnisebenen) aber auch in der Baumstruktur entsprechend lange Wartezeiten verursachen.



5.3 Anforderungen an eine Verzeichnisstruktur

Ein modernes Dateisystem muss heutzutage hohen Ansprüchen genügen. Es muss eine möglichst hohe Geschwindigkeit bieten um Dateien zu suchen, zu erstellen und zu löschen. Gerade die bereits angesprochene Dateiorganisation (siehe 3.3 - Dateiorganisation) spielt hierbei eine große Rolle um den gegebenen Speicherplatz (auf z.B. einer Festplatte) so effizient wie möglich zu nutzen.

Auch die Erstellung von mehreren Dateien mit demselben Namen oder die Benennung einer Datei mit mehreren Namen (Verknüpfungen) spielt eine Rolle. Die Optimierung der Schreib- bzw. Leseoperationen rückt immer mehr in den Vordergrund. Viele Dateisysteme verwenden einen Buffer-Cache in dem die Speicherblöcke für den schnelleren Zugriff zwischengespeichert werden. Auch die Plattenkopfbewegungen in Festplatten werden ständig optimiert.

Ein weiteres Ziel ist es die Notwendigkeit von Defragmentierungen so weit wie möglich zu reduzieren, ohne dabei die Konsistenz der Daten zu gefährden. Die Unterstützung verschiedenster Journaling-Funktionen (siehe 7.2 - Journaling-Dateisystem) kann ebenfalls als Maß für die Leistungsfähigkeit eines Dateisystems herangezogen werden.

6 Zugriff auf Dateisysteme bzw. Daten

Der Zugriff auf ein Dateisystem erfolgt über Befehle die je nach Betriebssystem verschieden sein können. Es gibt eine ganze Reihe teilweiser kostenloser Tools mit denen man Zugriff auf unterschiedliche Dateisysteme erhält. Auch auf mobile Dateisysteme (von z.B. Smartphones) kann man so zugreifen.

6.1 Zugriffsmethoden

Der Zugriff auf Daten in einem Dateisystem kann wahlfrei, sequentiell oder indexsequentiell erfolgen und gibt an, in welcher Reihenfolge man Datensätze einer Datei ansprechen kann.

Beim wahlfreien Zugriff kann ein System direkt auf die (richtige) Speicherstelle zugreifen, ohne das Speichermedium sequentiell durchlaufen zu müssen. Man kann das Verfahren mit dem Aufschlagen eines Buches vergleichen in dem man jede Seite sofort aufschlagen kann ohne alle vorherigen Seiten durchlaufen zu müssen (im Gegensatz zu einer Schriftrolle, die sequentiell durchlaufen werden muss). Ein Bespiel hierfür ist das DVD-Laufwerk eines Rechners.

Sequentieller Zugriff bedeutet, dass ein Zugriff auf die Datensätze nur in fortlaufender Reihenfolge möglich ist, was meist durch das Speichermedium definiert wird. Eine Anwendung muss also am Anfang des Speicherbereiches der Datei mit dem Lesevorgang beginnen und den Bereich so lange durchlaufen bis es die richtige Stelle „gefunden“ hat oder am Ende der Datei "angekommen" ist. Ein Beispiel hierfür ist ein Magnetband, auf das materialbedingt nur fortlaufend zugegriffen werden kann.

Der indexsequentielle Zugriff wird bei besonders großen Datenmengen häufig verwendet. Jeder Datensatz wird bei diesem Verfahren mit einem Schlüssel gekennzeichnet. Dieser Schlüssel wird zusammen mit einer Referenz auf die entsprechende Datei in einer Tabelle gespeichert. Wenn jetzt anhand des Schlüssels nach einer Datei gesucht wird, wird zuerst die Schlüsseltabelle mit den Referenzen auf die entsprechenden Dateien durchsucht und der richtige Eintrag zurückgegeben. Das Verfahren hat den Vorteil, dass es gerade bei großen Datenmengen einen schnellen und damit effizienten Zugriff auf die Daten liefert. Betriebssysteme unterstützen dieses Verfahren heute eher selten, dafür werden meist Datenbanken verwendet.

6.2 Globale Operationen

Globale Operationen sind Operationen, die sich direkt auf das Dateisystem beziehen.

Create: Ist das neue Anlegen einer Datei oder eines Verzeichnisses.
Open: Ist das Öffnen einer bestehenden Datei oder Verzeichnisses.
Close: Ist das permanent machen (oder Schließen) einer Datei.
Delete: Ist das Entfernen einer Datei oder eines Verzeichnisses.

6.3 Lokale Operationen

Lokale Operationen beziehen sich nur auf die entsprechenden Dateien.

Read, Write: Ist das Lesen, Schreiben einer Datei.
Reset, SetPos, GetPos (Seek): Ist das Positionieren einer Datei.

Zusätzlich gibt es noch Zustandsabfragen an einer Datei (z.B. die Länge einer Datei, Zeitpunkt der letzten Änderung usw.)

7 Sicherheitsaspekte

Ein Dateisystem muss dafür sorgen, dass auch beim Mehrbenutzerbetrieb keine Daten ungewollt überschrieben werden. Es muss also eine saubere Trennung der Zugriffe gewährleistet werden. Auch muss ein Vorgang bei evtl. auftretenden Störungen wie z.B. einem Stromausfall entweder ganz oder gar nicht ausgeführt werden, um die Daten immer konsistent zu halten.

In diesem Zusammenhang lässt sich das eigentlich aus dem Datenbank-Bereich stammende ACID-Prinzip auch auf Dateisysteme anwenden.

A: Atomarität, Änderungen werden ganz oder gar nicht durchgeführt.
C: Konsistenz, Stimmigkeit der Daten. Daten sollten immer in einem konsistenten Zustand gehalten werden.
I: Isolation, parallele Zugriffe haben keinen gegenseitigen Einfluss (z.B. kein gleichzeitiges Schreiben von 2 Benutzern auf dieselbe Datei).
D: Dauerhaftigkeit, nach erfolgreichem Zugriff bleiben Daten dauerhaft konsistent.

7.1 Integrität vs. Konsistenz

Daten sind Konsistent, wenn sie stimmig, bzw. plausibel sind. Konsistenz bezieht sich also auf die Zulässigkeit von Daten. Durch Fehlererkennungsverfahren und Paritätsprüfungen können Daten konsistent gehalten werden. Ein Beispiel hierfür ist das Online-Banking, in dem ein vordefiniertes Format für die Kontonummer existiert (z.B. eine feste Anzahl von Zahlen) das bei der Eingabe überprüft werden kann.

Integrität von Daten bedeutet im Gegensatz zur Konsistenz, dass sich alle Dateien in einem "gewollten" Zustand befinden und nicht von unbefugten Dritten oder durch Systemfehler manipuliert wurden. Man will also sicherstellen, dass eine Datei unverändert vom Sender zum Empfänger übertragen wird. Zur Wahrung der Integrität gibt es verschiedenste Prüfverfahren, wie Hash-Funktionen oder digitale Signaturen.

In Dateisystemen wird unbefugtem Zugriff auf Dateien durch das Setzen von entsprechenden Zugriffsrechten entgegengewirkt. Man unterscheidet u.a. zwischen Lese-, Schreib-, und Ausführungsrechten, die bestimmten Benutzern für Dateien oder Verzeichnisse zugeordnet werden können. In einer Systemumgebung lassen sich auch Gruppen mit entsprechenden Rechten erstellen, in die die einzelnen Benutzer dann eingeteilt werden können.

7.2 Journaling-Dateisystem

Seit einiger Zeit gibt es die Journaling-Dateisysteme, welche alle Änderungen bei einem schreibenden Zugriff in einem extra dafür bereitgestellten Speicherbereich (Journal) aufzeichnen. Das hat den Vorteil, dass bei Systemabstürzen, Stromausfällen oder anderen unvorhersehbaren Zwischenfällen abgebrochene Zugriffe auf die Daten anhand des Journals rekonstruiert werden können. So ist eine dauerhafte Konsistenz der Daten gewährleistet. Auch eine Überprüfung der Dateisysteme nach einem Absturz, die bei großen Partitionen sehr lange dauern kann, entfällt so.

Beim Journaling-Vorgang wird jeder schreibende Zugriff auf die Daten zuerst im Journal "notiert". Man unterscheidet dabei zwischen Metadaten – Journaling, bei dem die Konsistenz des Dateisystems sichergestellt wird und Full – Journaling, bei dem auch die Konsistenz der Dateiinhalte berücksichtigt wird. Wenn in der Praxis vom Journaling gesprochen wird, ist meist das Meta – Daten Journaling gemeint.

Beispieleintrag im (Meta - Daten) Journal: Datei X wird von Verzeichnis a nach Verzeichnis b kopiert. Sollte jetzt das System beim Kopiervorgang abstürzen, würde (nach Neustart des Systems) der Vorgang mit Hilfe des Journals rekonstruiert werden können und das Dateisystem wäre wieder in einem konsistenten Zustand.

Ein Nachteil der Journaling-Systeme ist sicher die Tatsache, dass durch das ständige Schreiben in das Journal wesentlich mehr Schreibzugriffe erforderlich sind, die die Gesamtperformance (wenn auch nur leicht durch das Schreiben einer vergleichsweise kleinen LOG-Datei) beeinträchtigen.

Demgegenüber steht aber u. a. der Vorteil, dass beispielsweise beim Herunterfahren eine Sitzung mit Hilfe des Journals gespeichert oder verworfen werden kann. Ein Beispiel ist hier das MAC-Dateisystem HFS+ Journaling, bei dem man die Möglichkeit hat einen Schreibzyklusintervall festzulegen nach dem jeweils Daten auf die Platte geschrieben werden. Es ist dann möglich eine Zyklus von beispielsweise 20 Minuten festzulegen und beim Verursachen eines Systemfehlers einfach die aktuelle Sitzung zu verwerfen, so dass die (fehlerhaften) Daten nicht auf die Platte geschrieben werden.

7.3 Datensicherheit

Durch die immer komplexer werdenden Dateisysteme spielt auch die Datensicherheit bzw. Datenrettung eine immer größere Rolle. Ist ein Dateisystem aufgrund von Abstürzen, Stromausfällen etc. nicht mehr auf normalem Wege zugreifbar, gibt es inzwischen viele Tools die ein Dateisystem wiederherstellen können.

Die meisten arbeiten nach demselben Prinzip: Das Festplattenlaufwerk wird als Ansammlung von Blöcken gesehen. Die Tools arbeiten dann als Abstraktionslevel unterhalb von Dateien oder Verzeichnissen. Traditionelle Backupprogramme unter Unix sind *dump* und *restore*.

8 Fazit und Ausblick

Abschließend lässt sich festhalten, dass Dateisysteme sich im Laufe der Zeit rasant entwickelt haben. Ging es anfangs nur um das einfach Strukturieren von Dateien, spielen inzwischen auch andere Anforderungen wie eine gut organisierte Verzeichnisstruktur, ein effizienter Speicherzugriff und die Wahrung der Datenkonsistenz eine große Rolle und haben dafür gesorgt das sich bis heute eine Vielzahl von Dateisystemen herausgestellt haben.

Angetrieben durch den Siegeszug des Internets wird sich diese Entwicklung sicher noch weiter fortsetzten. Im Mittelpunkt wird dabei weiter der effiziente Zugriff auf mehrere Dateisysteme (gleichzeitig) stehen, sowie die Verwaltung und Strukturierung von immer größeren Datenmengen.

Weitere zukunftsträchtige Funktionen von Dateisystemen lassen sich gut am (stand heute) "Linux-Dateisystem der Zukunft" dem BTRFS festmachen. BTRFS ermöglicht u.a. die Verwaltung von 2 Baumstrukturen (einen für Verzeichnis- und Dateinamen und einen für Datenblöcke), das Erstellen von Snapshots (also Abbilder zu Wiederherstellungszwecken), eingebaute RAID-Funktionen und das Bilden von Checksummen zur Datenüberprüfung. Man darf also gespannt sein, was die Entwicklung der Dateisysteme in den nächsten Jahren mit sich bringt.

9 Quellenverzeichnis

http://www.Heise.de

http://www2.sub.uni-goettingen.de

http://www.uni-protokolle.de

http://www.itwissen.info

http://de.linwiki.org/wiki/Linuxfibel_-_System-Administration_-_Dateisysteme#Das_virtuelle_Dateisystem

Proseminar Linux Internals: Joel, Ngueagni Gnitedem

http://www.trampelwurm.ch/schmidt/wilhelmtux/swissremix/html/Linuxfibel/dirstruct.htm

http://www.lexitron.de

http://de.wikipedia.org/ -> Suchwörter: Dateisystem, Journaling-Dateisystem, Datensicherheit

Betriebssysteme für Medieninformatiker, Dipl.-Inf., Dipl.-Ing. (FH) Michael Wilhelm Hochschule Harz

Betriebssysteme, Harald Kosch

Illustrationen erstellt mit OmniGraffle (MAC)

Abbildung auf dem Deckblatt:
http://www.heise.de/open/imgs/10/2/1/8/0/8/0/e6e229ff4a927dff.png