

Musterlösungen zu den Hausaufgaben auf
Blatt 11
der Übungen zur Vorlesung
“Grundlagen Betriebssysteme und Systemsoftware

G.Groh, 21.01.2009

Allgemeine Infos

Die **maximale Länge für Dateinamen** beträgt **255 Zeichen**. In Unix Dateisystemen werden Dateien als lineare Bytesequenz realisiert. In NTFS hingegen bestehen **Dateien aus mehreren Dateiattributen**, die jeweils als **Bytestrom** repräsentiert werden. Das Prinzip wurde bei Apple Mactintosh abgeschaut und sollte ursprünglich eine Interoperabilität zwischen den beiden System gewährleisten. Einer dieser Ströme ist der eigentliche **Inhalt dieser Datei** und wird im Folgenden als **Hauptstrom** bezeichnet. Beispielsweise entnimmt ein Bildbearbeitungsprogramm die Bilddaten einer Datei aus dem Hauptstrom und zusätzliche Thumbnaildaten aus einem weiteren benannten Strom. Ein Beispiel aus der Textverarbeitung wäre das Abspeichern des finalen Dokuments über den Hauptstrom, während eine Änderungshistorie über einen benannten Strom gesichert werden kann.

Struktur des Dateisystems

Die meisten NTFS Platten benützen **Blöcke** mit einer **Größe** von **4 Kilobyte**. Es sind allerdings **Blockgrößen von 512 Byte bis 64 Kilobyte möglich**. In jeder Partition der Platte gibt es eine **Master File Table (MFT)**, eine lineare Sequenz von 1kB großen Einträgen, die selber als File irgendwo innerhalb der Partition abgelegt wird. Die Adresse des ersten MFT Blocks ist im Bootblock gespeichert. Jeder MFT Eintrag beschreibt eine **Datei** oder ein **Verzeichnis**. Er enthält **Attribute** wie beispielsweise Name und Zeitstempel und eine **Liste von Adressen** der zugehörigen Blöcke. Bei großen Dateien können mehrere MFT Einträge genutzt werden, um die Blöcke der Datei zu verwalten. Sind die Werte von Attributen zu groß, so werden sie selbst in **Blöcken verwaltet** und als **Strom** zur Verfügung gestellt.

Um den Zugriff auf Daten zu optimieren, werden bei sehr kleinen Dateien die Daten **direkt in die MFT** geschrieben. Bei größeren Dateien werden die Daten nach Möglichkeit in aufeinander folgende Blöcke zusammengeschrieben und nur der erste dieser Blocksequenzen im MFT Eintrag referenziert.

Sicherheitsrichtlinien

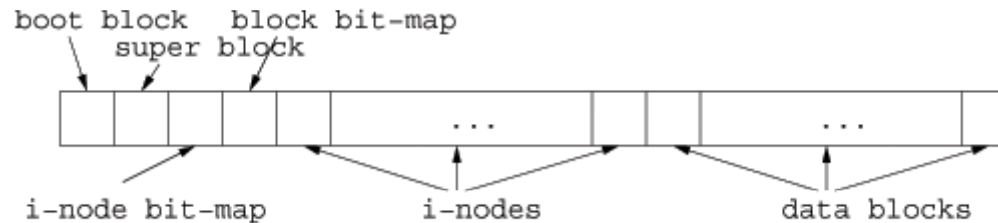
Jeder Benutzer wird als Subjekt mit einer eindeutigen **SID** identifiziert. Zusätzlich werden alle Dateien mit einer **DACL (Discretionary Access Control List)** versehen. In dieser DACL lassen sich **Allow** oder **Deny** Einträge zur Spezifikation der Zugriffsrechte von Gruppen oder einzelnen Subjekten für dieses Objekt angeben. Folgendes Beispiel veranschaulicht die Rechtevergabe:

Für eine Datei des Users Max kann spezifiziert werden, dass jeder User Lesezugriff hat, er selbst und Peter Schreibzugriff, die Gruppe der Gäste und Ingo gar keinen Zugriff.

Die Rechte werden seit Windows 2000 in einer **eigenen Datei** gespeichert. Dadurch ist es möglich, dass viele Dateien effizient die gleichen Sicherheitsrichtlinien nutzen.

Aufgabe 2.1 Lösungsvorschlag

Der Hintergrundspeicher (z.B. eine Festplatte) wird in **Blöcke** unterteilt, denen Unix folgende Struktur aufträgt:



Die einzelnen Bereiche haben dabei folgende **Funktionen**:

- **boot block**: Wird zum Starten des Systems benötigt
- **super block**: Beschreibt die Struktur des Dateisystems (Größe des gesamten Dateisystems, Größe und Lage der bit-map Blöcke, Wurzel des Dateisystems, etc.)
- **i-node bit-map bzw. block bit-map**: Ein Bit je (i-node oder Daten-)Block zeigt jeweils an, ob der zugeordnete Block frei ist
- **i-nodes**: Jeder i-node beschreibt genau eine Datei (vgl. Teilaufgabe c)
- **data block**: Enthält die in den Dateien gespeicherten Daten

Aufgabe 2.1 Lösungsvorschlag

Die Adressen der ersten n Blöcke einer Datei sind dabei **selbst im i-node enthalten ("direct")**. Werden mehr Blöcke benötigt, so wird ein zusätzlicher Block reserviert, der die Adressen weiterer Datenblöcke aufnimmt. Der Eintrag **"single indirect" im i-node** verweist dabei auf diesen zusätzlichen Block. Sollte auch dies noch nicht ausreichen, so wird eine weitere Indirektion eingeführt: Der Eintrag **"double indirect"** enthält die Adresse eines Blocks mit "single indirect"-Verweisen. Analog zeigt **"triple indirect"** auf einen Block mit "double indirect"-Verweisen.

Verzeichnisdateien werden in Unix wie **normale Dateien** mit Hilfe eines **i-nodes** abgelegt. Der i-node verweist dabei auf Datenblöcke, die in Tabellenform Datei- bzw. andere Verzeichnisnamen und die zugehörige i-node Nummer enthalten.

Die **Anzahl der möglichen Blöcke** einer Datei ergibt sich aus der Summe, der direkt, single indirect, double indirect und triple indirect adressierbaren Blöcke. Eine Adresse (Verweis auf einen Block) benötigt 4 Bytes, damit lassen sich bei einer Blockgröße von 1024 Bytes bei single indirect 256 Datenblöcke verwalten. Analog für double und triple indirect.

Aufgabe 2.1 Lösungsvorschlag

Es ergibt sich bei $n=10$ folgende maximale Anzahl von Blöcken für eine Datei:

i-node: 10 Blöcke

single indirect: 256 Blöcke

double indirect: $256 * 256 = 65536$ Blöcke

triple indirect: $256 * 256 * 256 = 16777216$ Blöcke

D.h. es sind insgesamt $10 + 256 + 65536 + 16777216 = 16843018$ Blöcke für eine Datei möglich.

Bei einer Blockgröße von 1 KB ergibt sich somit als maximale Größe einer Unix-Datei: $16843018 \text{ KB} \approx 16448 \text{ MB} \approx 16 \text{ GB}$

Aufgabe 2.2 Lösungsvorschlag

Der Zugriff auf alle Dateien und Verzeichnisse im Unix-System werden über **Zuriffsrechte** geregelt. Es gibt Rechte für das **Lesen** (**read**), **Schreiben** (**write**) und **Ausführen** (**execute**) von Dateien, die in einem **9-Bit-Vektor** abgelegt sind. Die Rechte gelten jeweils für den **Besitzer** (**owner**), **Mitglieder der Gruppe des Besitzers**, sowie alle **anderen**. Bei Verzeichnissen entspricht das Ausführen-Recht den Wechseln in das Verzeichnis.

Das **Beispiel** `rw-r-----` besagt, dass der Besitzer der Datei die Datei lesen und schreiben darf, andere Mitglieder der Gruppe können die Datei nur lesen, während alle anderen Benutzer keine Rechte an der Datei haben. Eine detaillierte Rechtfestlegung, z.B. für einzelne Benutzer unabhängig von der Gruppenzuordnung, ist nicht möglich.

Zugriffsrechte lassen sich in Unix mit Hilfe des Befehls `chmod` ändern.

Die Rechte werden nur beim Öffnen einer Datei geprüft. Hat also ein Benutzer das Schreibrecht an einer Datei, so kann er solange die Datei schreiben, wie er die Datei geöffnet hat. Eine Änderung der Rechte, also insbesondere eine Rechterücknahme, wirkt sich erst beim nächsten Öffnen der Datei aus.

Aufgabe 2.2 Lösungsvorschlag

Wem gehört die Datei? Immer noch dem Benutzer der sie kopiert hat.

Wer kann sich den Namen der Datei mit ls anzeigen lassen? Nur der Besitzer des Verzeichnisses, denn nur er kann das Verzeichnis lesen. Der Name der Datei in diesem Verzeichnis steht nur im Inode des Verzeichnisses.

Wer kann die Datei lesen? Nur der Besitzer der Datei, also in unserem Fall nicht der Besitzer des Verzeichnisses.

Wer kann die Datei verändern? Nur der Besitzer der Datei, er muss dazu nur die Datei schreiben dürfen.

Wer kann die Datei löschen? Alle Benutzer, da jeder Schreibzugriff auf das Verzeichnis hat. Ausser dem Besitzer des Verzeichnisses müssen aber alle vorher den Namen der Datei kennen, da sie sich nicht einfach das Verzeichnis anzeigen lassen können.

Aufgabe 3.1 Lösungsvorschlag

- 1 Zugriff, um den Datenblock des aktuellen Verzeichnis zu laden.
- 6 Zugriffe für die weiteren Verzeichnisse (2 pro Verzeichnis, einer für den I-Node, der andere für die Tabelle).
- 4 Zugriffe, um den letzten Block des Files zu finden: einer für den I-Node, dann wegen Dreifacher Indirektion weitere drei
- 2 Zugriffe, um den Block zu lesen, das Byte im Speicher zu ändern, und den Block zurückzuschreiben.
- 1 Zugriff für das Update der Änderungszeit im I-Node.

Total: 14

Aufgabe 3.2 Lösungsvorschlag

12 direkt, 1 indirekt, 1 doppelt indirekt , 1 dreifach indirekt

$$\mathbb{P}_{\text{direkt}} = \frac{12 \cdot 1\text{KB}}{10^6\text{KB}} = 1.2 \cdot 10^{-5}$$

$$\mathbb{P}_{\text{indirekt}} = \frac{1 \cdot 256\text{KB}}{10^6\text{KB}} = 2.56 \cdot 10^{-4}$$

$$\mathbb{P}_{\text{doppeltindirekt}} = \frac{1 \cdot 256^2\text{KB}}{10^6\text{KB}} = 6.5536 \cdot 10^{-2}$$

$$\mathbb{P}_{\text{dreifachindirekt}} = 1 - \mathbb{P}_{\text{direkt}} - \mathbb{P}_{\text{indirekt}} - \mathbb{P}_{\text{doppeltindirekt}} = 0.934196$$

$$\mathbb{E} = 1 \cdot \mathbb{P}_{\text{direkt}} + 2 \cdot \mathbb{P}_{\text{indirekt}} + 3 \cdot \mathbb{P}_{\text{doppeltindirekt}} + 4 \cdot \mathbb{P}_{\text{dreifachindirekt}} = 3.933$$