Prof. Dr. Christoph Scholl
Dr. Tim Welschehold
Alexander Konrad
Niklas Wetzel

Freiburg, 16.12.2022

Betriebssysteme
Musterlösung zu Übungsblatt 9

Aufgabe 1 (3 + 2 Punkte)

Wahlfreier Zugriff bei I-Nodes und bei FAT

Betrachten Sie ein Dateisystem mit Blockgröße b in Byte und Zeigergröße z in Byte.

a) Angenommen, es handelt sich um ein UNIX-Dateisystem basierend auf I-Nodes. Gehen Sie von der Implementierung in `System V` aus, d.h. jeder I-Node enthält 10 direkte Zeiger sowie jeweils einen einfach-, zweifach- und dreifach indirekten Zeiger.

   Wie läuft ein wahlfreier Zugriff auf das Byte Nummer $n = 50000$ einer Datei konkret ab? Die Blockgröße sei $b = 4$ KB und die Zeigergröße sei $z = 4$ Byte, die Anzahl der direkten Zeiger ist 10. Die Nummerierung der Bytes fängt mit der Nummer 0 an. Geben Sie an, welche Zeiger daran beteiligt sind, an welcher Position in den Blöcken diese zu finden sind und wohin sie zeigen. Geben Sie den Rechenweg mit an. Bezeichnen Sie dabei das vorderste Element in einem Block als „0. Element“.

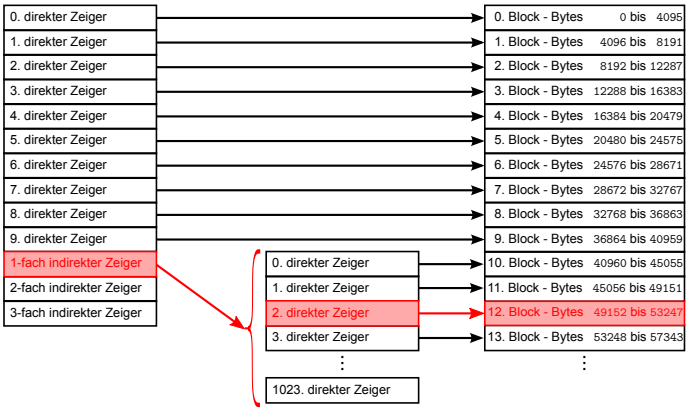
b) Wie läuft der wahlfreie Zugriff auf ein beliebiges Byte Nr. n einer Datei ab, falls statt eines I-Node-Systems nun ein FAT32-Dateisystem verwendet wird? Geben Sie einen allgemeinen Ausdruck für die Anzahl der zu verfolgenden Verweise N innerhalb der FAT in Abhängigkeit von der Blockgröße b an. Wie wird sich die Zugriffszeit in Abhängigkeit von n verhalten?

Lösung:

a) Wahlfreier Zugriff auf Byte Nr. 50000 [1.5]:

- Die zehn direkten Zeiger erreichen die Datenblöcke 0 bis 9 mit den Bytes 0 bis $10 \cdot 4096 - 1 = 40959$, das gesuchte Byte befindet sich also nicht in diesem Bereich.
- Der einfach indirekte Zeiger erreicht Byte $10 \cdot 4096 = 40960$ bis $10 \cdot 4096 + 1024 \cdot 4096 - 1 = 4235263$ (1 indirekter Block mit 1024 Zeigern auf 4KB Blöcke). Das gesuchte Byte lässt sich also über den einfach-indirekten Zeiger finden.

- Der nullte Zeiger im einfach indirekten Block zeigt auf den Datenblock 10, der mit Byte 40960 beginnt. Das Byte 50000 befindet sich $\left\lfloor \frac{50000-40960}{4096} \right\rfloor = 2$ Blöcke weiter, also muss dem zweiten Zeiger auf den 12. Datenblock gefolgt werden.
- Der 12. Datenblock reicht von Byte $12 \cdot 4096 = 49152$ bis $13 \cdot 4096 - 1 = 53247$.
- An Pos. $50000 - 12 \cdot 4096 = 848$ in diesem Block befindet sich also das Byte Nr. 50000.



b) Ablauf bei FAT32:

- Die einfach verkette Liste der Datenblöcke der Datei wird sequentiell gelesen [0.5]
- Für die Anzahl N der verfolgten Verweise innerhalb der FAT gilt $N = \lceil \frac{n}{b} \rceil - 1$. Danach wird dem gefundenen Verweis auf den Plattenblock gefolgt.[0.5]
- Die Komplexität ist $\mathcal{O}(n)$, also ist die Zugriffszeit in Abhängigkeit von n asymptotisch linear.[0.5]

Aufgabe 2 (2 + 2 + 2 + 2 Punkte)

Hardlinks und symbolische Links

Für diese Aufgabe wird angenommen, dass Sie auf einem Dateisystem arbeiten, das auf dem in der Vorlesung vorgestellten I-Node-Konzept basiert. Die Blockgröße des Dateisystems betrage 1024 Byte. Sie haben eine Datei `/home/max/brief.doc`, die 2,5 KB an Daten enthält. Weiterhin haben Sie 2 symbolische Links („`symlink1`“ und „`symlink2`“) und 3 Hardlinks („`hardlink1`“, „`hardlink2`“ und „`hardlink3`“), die alle auf `brief.doc` verweisen.

a) Erläutern Sie anhand einer Skizze, wie diese Situation im Dateisystem realisiert ist. Ergänzen Sie dazu Abbildung 1 um die fehlenden Verzeichniseinträge, I-Nodes und Datenblöcke. Stellen Sie Verweise durch Pfeile zwischen den Elementen dar.

b) Wieviel Plattenplatz (in Anzahl Blöcken und in KB) benötigen die Datenblöcke der Datei `brief.doc` und die Datenblöcke der Links auf `brief.doc`? Wie viele I-Nodes und wie viele Verzeichniseinträge werden benötigt?

c) Die Zugriffsrechte der Datei `brief.doc` werden mit Hilfe des Befehls `chmod` geändert. Wie ändern sich dadurch die Zugriffsrechte der Hardlinks und der symbolischen Links? Probieren Sie dies in der Praxis aus und erklären Sie das beobachtete Verhalten.

d) Wenn Sie einen symbolischen Link anlegen, werden die Dateirechte als `lrwxrwxrwx` angezeigt. Versuchen Sie mit `chmod` die Rechte eines symbolischen Links zu ändern. Was passiert mit den Rechten des symbolischen Links und warum könnte dieses Verhalten sinnvoll sein?

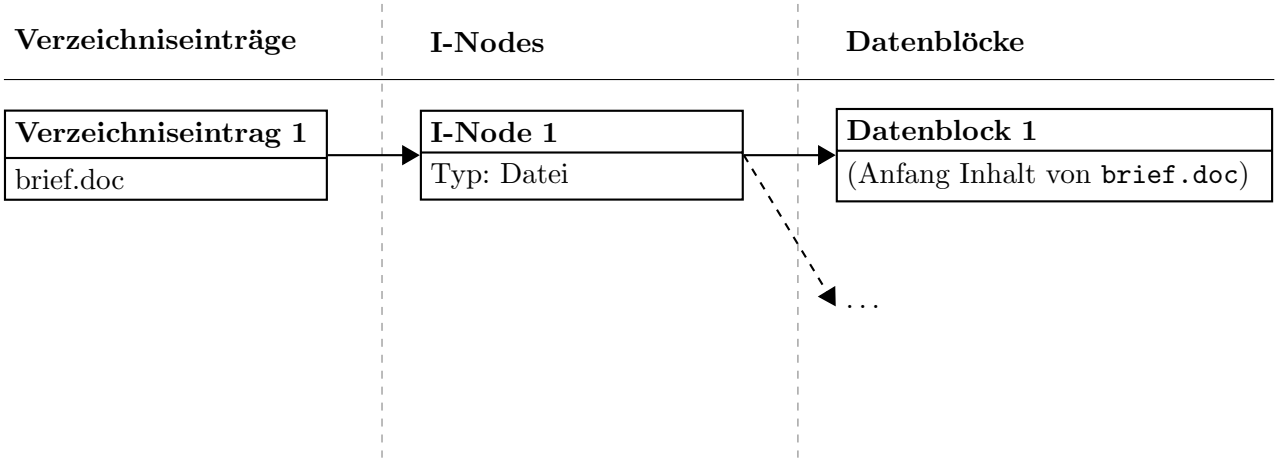


Abbildung 1: Realisierung der Dateien im Dateisystem

Lösung:

a) Beispiel für Skizze (Pfad in Datenblöcken 3/4 kann relativ oder absolut angegeben werden) findet sich in Abb. 2. [2]. Sofern es vom Studierenden begründet wird (z.B. mit Verweis auf das ext-Dateisystem), dürfen die Datenblöcke 3 und 4 auch fehlen und der Pfad stattdessen in den I-Nodes 2 bzw. 3 angegeben sein. In diesem Fall sollte auch für b) eine entsprechend konsistente Lösung mit nur zwei Datenblöcken angegeben sein.

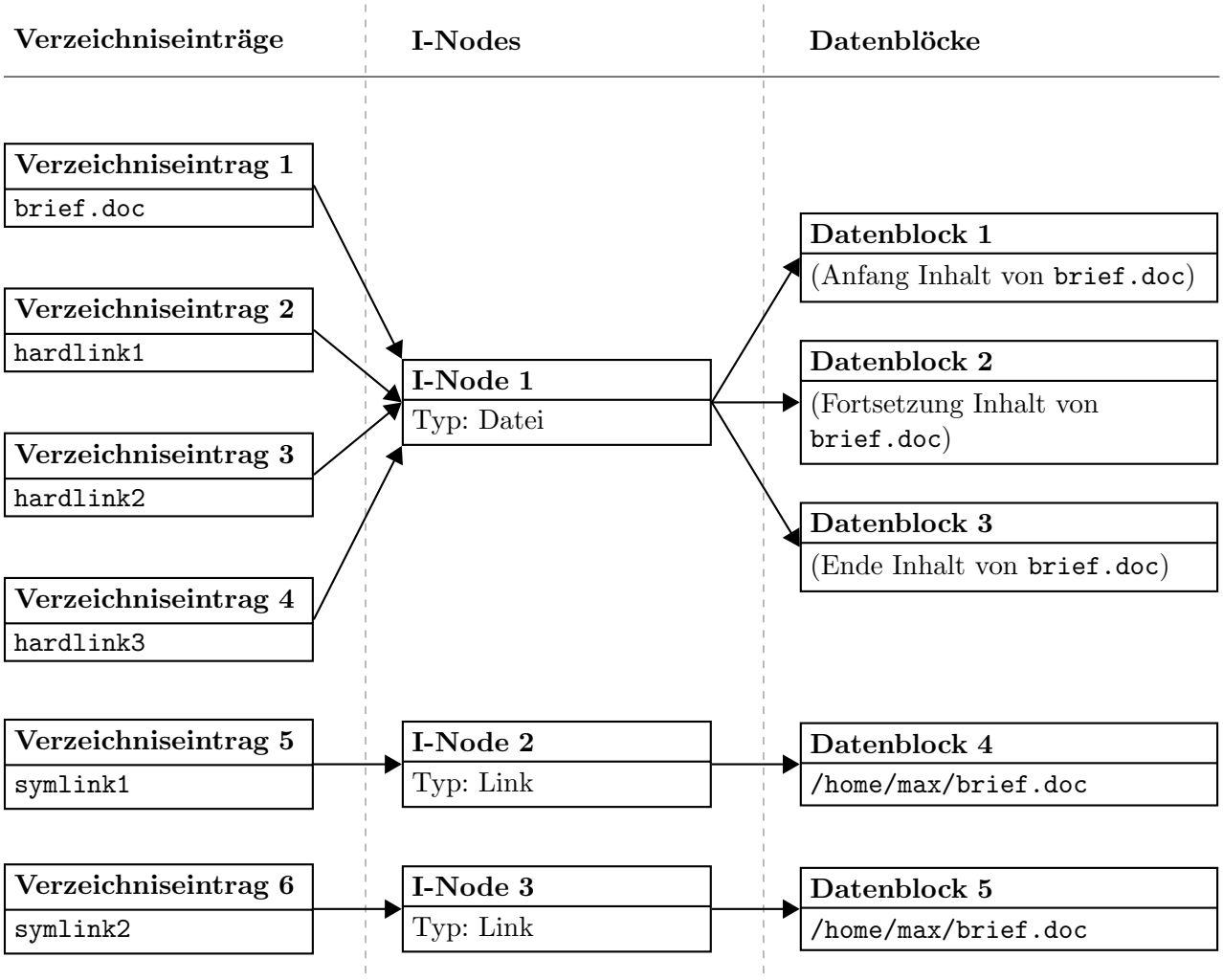


Abbildung 2: Realisierung der Dateien im Dateisystem

b)
- Datei selbst: 1 I-Node, 3 Datenblöcke, 1 Verzeichniseintrag ($\Rightarrow 3\,\text{KB}$)
- Symlinks: jeweils 1 I-Node, 1 Datenblock für die Linkdatei und 1 Verzeichniseintrag ($\Rightarrow 2 \cdot 1\,\text{KB}$)
- Hardlinks: jeweils 1 Verzeichniseintrag, 0 I-Nodes und 0 Datenblöcke

Insgesamt 5 Datenblöcke = 5 KB Speicherplatz, 3 I-Nodes, 6 Verzeichniseinträge [1]. Alternativ können die zusätzlichen Datenblöcke der symbolischen Links wegfallen, wenn die Pfadnamen in den I-Nodes gespeichert werden. Dies ist bei dem Dateisystem ext möglich und sollte bei Verwendung von den Studenten angegeben werden.

Die Verzeichniseinträge benötigen zusätzlich Speicherplatz in den Verzeichnistabellen der Elternverzeichnisse, in denen die Dateien liegen. Dieser Speicherplatz und der Speicherplatz für die I-Nodes selbst wird hier vernachlässigt.

c) Die Zugriffsrechte aller Hardlinks ändern sich mit, da die Rechte im I-Node gespeichert werden und die Datei und alle Hardlinks auf das selbe I-Node zeigen [0.5]. Die Zugriffsrechte aller symbolischen Links bleiben bei `lrwxrwxrwx`, da sie eigene I-Nodes haben [0.5].

d) Die Zugriffsrechte des symbolischen Links können nicht geändert werden [0.5]. Stattdessen wird der `chmod`-Befehl auf das Linkziel angewandt, insofern die entsprechenden Rechte bestehen.

Die angezeigten Rechte der symbolischen Links haben keine Bedeutung für den eigentlichen Zugriff auf die Zieldatei. Wäre das nicht der Fall, so könnte man einen symbolischen Link auf eine fremde Datei anlegen. Da man selbst der Besitzer des Links ist, könnte man die Rechte des Links nach Belieben ändern und sich somit Zugriffsrechte verschaffen. Aus diesem Grund werden symbolische Links bei Benutzung immer erst dereferenziert und dann werden die Rechte des Linkziels ausgewertet um die Zugriffsrechte zu bestimmen [0.5].

Aufgabe 3 (1 + 1 + 2 + 3 Punkte)

Erstellen Sie zunächst eine Datei `counter.sh` mit folgendem Inhalt:

counter.sh

```
#!/bin/bash
let x=0                          # setze Variable x auf 0
while true                       # Wiederhole endlos
do
    echo $x                      # gib Variable x aus
    let x=x+1                    # Zähle x um 1 hoch
    sleep 1                      # Warte 1 Sekunde
done
```

Das Skript implementiert einen einfachen Zähler. Machen Sie das Skript mit `chmod u+x counter.sh` ausführbar und führen Sie es mit `./counter.sh` aus.

Öffnen Sie ein weiteres Terminal oder eine zweite SSH-Verbindung und bearbeiten Sie die Aufgabe, während `counter.sh` im ersten Terminal weiterläuft.

a) Der Befehl `ps` zeigt Informationen über die derzeit laufenden Prozesse an. Ohne Optionen zeigt `ps` nur die Prozesse des aktuellen Benutzers an, die auf der aktuellen Textkonsole gestartet wurden.

Welche Optionen muss man beim Aufruf von `ps` benutzen, um eine komplette Liste der derzeit laufenden Prozesse zu erhalten?

b) In Linux werden *Signale* verwendet, um Prozesse zu steuern und einfache Nachrichten („Daten stehen bereit“, „Fehler aufgetreten“ usw.) zwischen Prozessen auszutauschen. Dazu gehören unter anderem

- `SIGSTOP`
- `SIGCONT`
- `SIGTERM`
- `SIGKILL`

Wie können Sie den Befehl `kill` verwenden, um diese Signale an den Prozess von `counter.sh` zu senden?

Hinweis 1: Die Prozess-ID (PID) können Sie mit `pidof -x counter.sh` bestimmen oder in der Ausgabe von `ps` ablesen.

Hinweis 2: Um die Manpage der *Signale* aufzurufen, verwenden Sie den Befehl `man 7 signal`.

c) Beschreiben Sie, was jedes der vier Signale aus Teilaufgabe b) bewirkt. Erklären Sie auch den Unterschied zwischen `SIGTERM` und `SIGKILL`.

d) Sie möchten `counter.sh` starten, sich vom Rechner abmelden, `counter.sh` aber trotzdem weiter laufen lassen und die Ausgabe nach dem nächsten Einloggen wieder aufrufen. Sobald das Terminal-Fenster geschlossen wird, wird jedoch auch der Prozess beendet. Eine mögliche Lösung besteht darin, das Programm *GNU Screen* zu verwenden. Es wird mit dem Befehl `screen` gestartet. Welche Kommandos und Eingaben werden benötigt, um diese Teilaufgabe mit *GNU Screen* zu lösen?

Hinweis 1: Die englischen Begriffe für das gesuchte Verfahren heißen „detaching“ und „reattaching“.

Hinweis 2: Sollte *GNU Screen* noch nicht auf Ihrem Rechner installiert sein, können Sie das Programm mit dem Befehl `sudo apt-get install screen` installieren. Hierfür benötigen Sie Administratorrechte.

Lösung:

a) Mehrere Möglichkeiten, z.B.

- `ps ax`
- `ps aux`
- `ps -ef`

[0.5 Punkte]

b) Es existieren eine Reihe von möglichen Befehlen, z.B.

- `kill -s <signal value> <pid>`
- `kill -<signal value> <pid>`
- `kill -<option1> <pid>`
- `kill -<option2> <pid>`

Dabei gilt:

	signal value	option1	option2
SIGSTOP	19 (für x86-Architektur)	SIGSTOP	STOP
SIGCONT	18 (für x86-Architektur)	SIGCONT	CONT
SIGTERM	15	SIGTERM	TERM
SIGKILL	9	SIGKILL	KILL

[0.5 Punkte]

Die PID kann entweder manuell ermittelt und eingetragen werden oder aber der Ausdruck `<pid>` wird durch `$(pidof -x counter.sh)` ersetzt.

Ergänzung: Um die Ermittlung der PID zu umgehen, kann statt `kill [...] <pid>` der Befehl `killall [...] counter.sh` mit der selben Syntax verwendet werden.

c) Bedeutung der Signale:

Signal	Bedeutung
`SIGSTOP`	Blockiert den Prozess.
`SIGCONT`	Setzt einen gestoppten Prozess fort [als Hintergrundprozess].
`SIGTERM`	Fordert den Prozess auf sich zu beenden, kann ignoriert werden.
`SIGKILL`	Erzwingt das Beenden des Prozesses.

Unterschied SIGTERM und SIGKILL: SIGTERM kann vom Prozess ignoriert werden oder der Prozess kann noch seine Ressourcen aufräumen und dann terminieren. Das Verhalten wird vom Programmierer vorgegeben. SIGKILL kann vom jeweiligen Prozess nicht ignoriert werden. Der Prozess wird mit SIGKILL beendet, ohne dass er auf das Signal reagieren kann.

[0.25 Punkte pro korrekt beschriebenem Signal, auf 0.5 aufrunden; 0.5 Punkte für Unterschied zwischen SIGTERM und SIGKILL]

d)
1) Screen starten: „`screen`“ oder „`screen -S <sessionname>`“
2) Counter starten: „`./counter.sh`“
3) Detach: Ctrl-a d
4) Ausloggen und wieder einloggen
5) Session suchen: „`screen -ls`“
6) Re-attach: „`screen -r <sessionname>`“

Alternativen:

- screen starten und sofort detachen: „`screen -d -m ./counter.sh`“
- Wenn nur eine Session läuft, kann „`screen -r`“ ohne Session-Name aufgerufen werden. Dieser Befehl re-attached die letzte Session.

[1.5 Punkte]

Abgabe: als PDF im Übungsportal bis 23.12.2022 um 12:00