TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Systemarchitektur, Professur für Betriebssysteme

Betriebssysteme und Sicherheit, WS 2021/22

14. Aufgabenblatt – Dateisysteme und UNIX Rechte

Geplante Bearbeitungszeit: 31.01.2022 – 04.02.2022

Aufgabe 14.1 In einem Unix-Dateisystem beträgt die Blockgröße 4 KiB. Eine Allokations-Bitmap dient der Verwaltung belegter Blöcke. In diesem Dateisystem befindet sich eine Datei `/tmp/A` mit einer Länge von 3000 Byte, welche von einem Programm zum Schreiben geöffnet wird. Das Programm hängt an die bereits vorhandenen Inhalte der Datei weitere 16 KiB Daten an und schließt die Datei danach wieder.

(a) Welche Blöcke muss das Dateisystem lesen, wenn die Datei wie oben beschrieben geöffnet wird?

(b) Welche Änderungen müssen an den Metadaten in den Dateisystemstrukturen vorgenommen werden, um die neu geschriebenen Inhalte im Dateisystem abzulegen?

(c) Was wäre eine sichere Reihenfolge in der alle modifizierten und neuen Blöcke aus der letzten Teilaufgabe geschrieben werden können, wenn man möglichst wenige Inkonsistenzen nach einem Absturz haben möchte. Diskutieren Sie außerdem welche möglichen Inkonsistenzen auftreten können und was diese jeweils für Auswirkungen auf die Ausführung des Systems haben kann.

(d) Viele Festplatten führen die ihnen übermittelten Schreib- und Lesebefehle nicht in der gegebenen Reihenfolge aus. Nutzen Sie die Synchronisationsoperation *Write Barrier* um die Reihenfolge der Schreiboperationen der letzten Teilaufgabe genau zu definieren. Auf welche Situationen ist besonders zu achten

(e) Wie würde die Reihenfolge der Schreiboperationen aus der letzten Teilaufgabe sich verändern, wenn man ein Dateisystem mit Journaling verwendet?

Aufgabe 14.2 Die folgenden Fragen und Aufgaben beziehen sich auf die "klassische" Sicherheitsarchitektur von UNIX.

(a) Welche Informationen werden beim Öffnen einer Datei durch das Betriebssystem überprüft? Was sind gemäß der Vorlesung die beteiligten „Subjekte" und „Objekten"? Wie ist der Ablauf im Betriebssystem, der entscheidet, ob eine Datei geöffnet werden darf oder nicht?

(b) Formulieren Sie eine Rechtezuteilung mittels des Unix-Rechtesystems für folgende Situation:

- Nutzer A darf die Datei D nur lesen.
- Die Mitglieder der Gruppe G dürfen die Datei D lesen, schreiben und ausführen.
- Alle anderen Nutzer im System dürfen die Datei nur lesen und ausführen.

(c) Welche Probleme treten auf, wenn man das Beispiel der letzten Teilaufgabe um folgende Bedingung erweitert?

- Nutzer B darf die Datei D weder lesen, schreiben noch ausführen.

(d) Verdeutlichen Sie ein weiteres Problem, das mit dem Rechtesystem von Unix verbunden ist, anhand des Änderns eines Passworts in einem Unix-Betriebssystem. Die verschlüsselten Passwörter sind in einer Datei `passwd` gespeichert, die von jedem gelesen, aber nur mit Hilfe eines speziellen (gleichnamigen) Programms geschrieben werden kann. Die relevanten Spezifikationen für diese beiden Objekte lauten:

```
rw- r-- r-- root root /etc/passwd

rwx r-x r-x root root /usr/bin/passwd
```

Beschreiben Sie zunächst das Problem, das beim Ausführen des Programms auftritt, und anschließend die Lösung dieses Problems.

Klausuraufgabe I

In einem Unix-artigen Dateisystem werden alle Dateiinhalte und Metadaten in Blöcken auf dem Speichermedium organisiert. Die Größe eines Blocks beträgt 4 KiB.

a) Wie groß muss die Bitmap mit den belegt/frei-Bits für alle Blöcke (inklusive der für die Allokations-Bitmap selbst verwendeten Blöcke) sein, wenn das Dateisystem auf einem 4 GiB großen Speichermedium erzeugt wurde?

b) Unix-Dateisysteme verwalten pro Datei jeweils ein sogenanntes „Inode“. Nennen Sie zwei verschiedene Arten von Metadaten, die in jedem Inode abgelegt sind. Erläutern Sie, welchen Vorteil Unix-Dateisysteme daraus ziehen, dass der Name einer Datei *nicht* im Inode gespeichert wird.

Ein Anwendungsprogramm hat eine neue, 42 KiB große Datei erzeugt. Im Zuge dieser Operation wurden die nachfolgend aufgelisteten Metadatenblöcke im Buffer Cache des Betriebssystems modifiziert bzw. neu allokiert und zunächst gepuffert:

1 Superblock (SB), 1 Allokations-Block für Inodes (IA), 2 Allokationsblöcke für Blöcke (BA), 2 Inode-Blöcke (I_1 und I_2), 1 Verzeichnisblock (V), 1 Indirektionsblock mit Zeigern auf Datenblöcke (Z), 11 Datenblöcke (D)

HINWEIS: I_1 enthält das neue Inode der angelegten Datei, I_2 das geänderte Inode des Elternverzeichnisses.

c) Geben Sie eine nach der Methode des „Synchronen Schreibens“ sichere Reihenfolge an, in der alle genannten Blöcke auf das Speichermedium geschrieben werden können, so dass das Dateisystem nach einem Absturz keine kritischen Inkonsistenzen aufweist. Markieren Sie in Ihrer Lösung die Blockschreiboperation(en) innerhalb der Sequenz, nach denen eine „Write Barrier“ für die Konsistenz nach einem Absturz *notwendig* ist.

Klausuraufgabe II

Ein Unix-artiges Dateisystem organisiert alle Dateiinhalte und Metadaten in Blöcken auf dem Speichermedium. Blöcke enthalten entweder Daten (D), Inodes (I), Verzeichniseinträge (V), eine Allokations-Bitmap für Inodes bzw. Blöcke (IA bzw. BA) oder den Superblock (SB). Die Größe eines Blocks beträgt 4 KiByte. Der initiale Zustand des Dateisystems ist wie folgt:

Block	0	1	2	3	4	5	6	7	8	9	10
Typ	SB	IA	BA	I	V	D	D				
Inhalt	/: 0 frei: 4	0 – 1	0 – 6	0: 4 1: 5,6	A: 1	Nutz-daten	Nutz-daten				

Inode-Zeiger für / und Anzahl freier Blöcke

Inodes 0 und 1 belegt

Block 0 bis 6 belegt

Blockzeiger in Inodes 0 und 1

Abbildung des Dateinamens *A* auf zugehöriges Inode 1

a) Welche Blöcke muss das Betriebssystem lesen, wenn der Nutzer mit dem Kommando `ln /A /B` einen weiteren Hardlink für die Datei *A* im Wurzelverzeichnis anlegt?

b) Nach dem Anlegen des Hardlinks erzeugt der Nutzer mit Hilfe eines Anwendungsprogramms im Wurzelverzeichnis eine neue Datei *C* mit einer Größe von 5144 Byte. Tragen Sie in das unten stehende Schema den Zustand des Dateisystems nach Abschluss dieser Operation ein. Kennzeichnen Sie alle Blöcke, deren Inhalt sich gegenüber dem oben abgebildeten Schema geändert hat.
HINWEIS: In Inode- und Verzeichnisblöcken sei ausreichend Speicherplatz vorhanden.

Block	0	1	2	3	4	5	6	7	8	9	10
Typ											
Inhalt											
geändert											

Zur Leistungssteigerung puffert das Betriebssystem neue oder modifizierte Blöcke zunächst im Hauptspeicher und schreibt diese später im Hintergrund auf das Speichermedium. Dabei werden Blöcke bestimmter, aber nicht aller, Typen zunächst in ein Log (oder Journal) geschrieben. Für das Log sind die Blöcke 11–16 exklusiv reserviert.

c) Tragen Sie in unten stehendes Schema eine mögliche Journal-Transaktion ein, bei der sichergestellt ist, dass Datei *C* nach einem Absturz korrekt und vollständig wiederherstellbar ist. Erläutern Sie, wann die jeweiligen „in-place“-Kopien (Blocknummern 0–10) geschrieben werden und warum nicht alle Blocktypen (welche?) ins Log geschrieben werden müssen.

Block	11	12	13	14	15	16
Typ						

Klausuraufgabe III

In einem System existieren die fünf Benutzer *Alice*, *Bob*, *Carol*, *Dave* und *Oskar*. Alice und Bob möchten auf einfache Weise miteinander kommunizieren und nutzen für diesen Zweck eine Datei `postfach`. Möchte bspw. Alice eine Nachricht für Bob hinterlassen, legt sie diese in der Datei ab. Zu einem späteren Zeitpunkt liest Bob die Nachricht und löscht sie bzw. ersetzt sie durch seine Antwort. Um ihre Kommunikation vertraulich zu halten soll die Datei `postfach` dabei *ausschließlich* für Alice und Bob zugreifbar sein.

a) Geben Sie für die Datei `postfach` eine geeignete Rechtezuweisung mittels Zugriffssteuerlisten (ACL) an.

b) Geben Sie eine mögliche Rechtezuweisung für `postfach` im Rahmen des klassischen Unix-Rechtemodells an. Welche zusätzlichen Voraussetzungen sind dabei erforderlich?

c) Die Datei `postfach` wurde in einem Verzeichnis `public_dir` angelegt, das folgende Rechtezuweisung besitzt

```
rwx  rwx  rwx    root    root    public_dir
```

Wie könnte *Oskar* diese Situation ausnutzen, um die Kommunikation zwischen *Alice* und *Bob* in Zukunft mitzulesen?
HINWEIS: Wir gehen davon aus, dass *Alice* und *Bob* arglos sind und nicht mit einem Angriff rechnen.

Ein zentraler Systemdienst hat aufgrund eines Programmierfehlers eine Datei *system.d* mit den folgenden Unix-Rechten angelegt

```
rws  rws  rwx    root    root    system.d
```

d) Erläutern Sie, warum dies ein Sicherheitsproblem darstellt.

Klausuraufgabe IV

Gegeben sei ein fiktives System des Herstellers *Herkules*. Neben dem Besitzer des Systems, *Bernd*, gibt es einen weiteren Anwender *Hans*. Sowohl *Bernd* als auch *Hans* gehören zur Gruppe *Users*.

a) *Bernd* lädt von der Internetseite des Herstellers die Programmerweiterung *Zeus* herunter. Er möchte sicherstellen, dass die Erweiterung tatsächlich von der Firma *Herkules* stammt und nicht von Dritten manipuliert worden ist.
Nennen Sie ein dafür geeignetes Verfahren oder erläutern Sie kurz warum dieses Problem prinzipiell nicht lösbar ist!

b) Geben Sie eine Rechtezuweisung im Rahmen des klassischen Unix-Rechtesystems an, die allen Benutzern *ausschließlich die Ausführung* des Programms `zeus` erlaubt!
Nutzen Sie dafür die Notation `<Rechte-Bits> <Besitzer> <Gruppe>`.

c) Um das Prinzip der geringst-möglichen Privilegisierung umzusetzen, gibt es einen Benutzer *dialout* welcher keiner Gruppe zugehört. Nur der Benutzer *dialout* darf auf das Modem zugreifen, beispielsweise um es durch die Eingabe der PIN freizuschalten. Für das Übergeben der PIN ans Modem gibt es ein Programm `enterpin`. Nur Mitglieder der Gruppe *Users* sollen damit die PIN eingeben dürfen.
Setzen Sie die nötigen Unix-Zugriffsrechte für `enterpin`, so dass *Bernd* die PIN auf seinem System eingeben kann!

d) Geben Sie eine Umsetzung der in b) beschriebenen Rechtezuweisung mittels Capability-Listen an und nennen Sie einen Nachteil von Capability-Listen gegenüber dem klassischen Unix-Rechtesystem!