Praktikum

zum Modul

Betriebssysteme 2

Prof. Dr.-Ing. Fritz Mehner
Prof. Dr.-Ing. Hans-Georg Eßer

Fachhochschule Südwestfalen

Fachbereich Informatik und Naturwissenschaften

Version 1.9.2
Stand: 3. Oktober 2019 (WS 2019/20)

Prof. Dr.-Ing. Fritz Mehner
Prof. Dr.-Ing. Hans-Georg Eßer (esser.hans-georg@fh-swf.de)
Fachhochschule Südwestfalen
Fachbereich Informatik und Naturwissenschaften
Frauenstuhlweg 31
58644 Iserlohn

Inhaltsverzeichnis

Einführung

Das vorliegende Dokument enthält die Praktikumsaufgaben zum Modul **Betriebssysteme 2** des Bachelor-Studienganges *Informatik*. Die Durchführung der Praktikumsaufgaben gibt Gelegenheit, das in der Vorlesung Gehörte anzuwenden und zu vertiefen. Zusätzlich muss der Stoff aus geeigneten Büchern oder Tutorien zur *Unix*-Linux-Systemverwaltung erarbeitet werden. Dazu sollte auch auf dem eigenen Rechner eine vergleichbar eingerichtete Linux-Installation zur Verfügung stehen.

> Es ist zwingend erforderlich, auch außerhalb der Lehrveranstaltungen das in der Vorlesung erworbene Wissen praktisch nachzuvollziehen, anzuwenden und zu vertiefen.

Der zur Lösung der jeweiligen Aufgaben benötigte Stoff entspricht dem Stand, der in der Vorlesung bis zur Bearbeitungszeit erreicht wurde. Das heißt jedoch nicht, dass es für einzelne Aufgabenteile nicht elegantere Lösungen gibt, die aber Kenntnisse erfordern, die im Augenblick noch nicht zur Verfügung stehen.

Erforderliche Unterlagen

Die folgenden Dokumente sind erforderlich und sollten auf Ihrem dem Rechner vorhanden sein. Zur besseren Handhabung sollten Icons auf dem Desktop oder Links im Browser angelegt werden.

- Terminplan zum Praktikum: Kursseite `http://swf.hgesser.de/b2-ws2019/`
- Bash Reference Manual [FSF10], `http://www.gnu.org/software/bash/manual/`
- Bash Style Guide und Kodierungsrichtlinie [Meh14], `https://lug.fh-swf.de/vim/vim-bash/StyleGuideShell.de.pdf`

Die Literaturliste zum Modul *Betriebssysteme 2* enthält weitere Angaben.

Durchführung des Praktikums

Teilnahme und Testat Die selbständige Bearbeitung der Praktikumsaufgaben ist Pflicht. Die *Bearbeitungspflicht* gilt als erfüllt, wenn für mindestens 80 Prozent der Praktikumsaufgaben die selbständige und erfolgreiche Bearbeitung durch den Betreuer bescheinigt wurde. Daraufhin wird der Erwerb der Vorleistung bescheinigt. Nur diese Vorleistung berechtigt zur Teilnahme an der Klausur. Die Abgabetermine regelt der Terminplan (siehe unten).

Eine Pflicht zur regelmäßigen Anwesenheit im Praktikum gibt es nicht. Die Praktikumstermine dienen dem Testieren der Abgaben und sollen zur persönlichen Beratung genutzt werden.

Terminplan Für die Testierung der Aufgaben gibt es einen Terminplan, der auf der Webseite zu der jeweiligen Veranstaltung eingesehen werden kann. Dieser Terminplan ist *verbindlich*. Eine Nachfrist für eine Aufgabe wird nur dann gewährt, wenn zum festgesetzten Termin die Lösung im wesentlichen vorlag und deshalb nur Schwächen oder Fehler zu beheben sind.

Der Praktikumsbetreuer kann für einen vorher festgelegten Teil einer Praktikumsgruppe einen um eine Woche späteren Termin festlegen, um die Abgaben zu entzerren und damit mehr Zeit für die Beratung aufwenden zu können.

Die zu testierenden Aufgaben müssen zu *Beginn des jeweiligen Praktikumstermins* vorliegen, um den Betreuern die Durchsicht der Aufgaben aller Teilnehmer zu ermöglichen.

Vorbereitung und Durchführung Die erfolgreiche Durchführung des Praktikums setzt voraus, dass der zugrundeliegende *Stoff im wesentlichen bekannt* ist und dass die Aufgabenstellung *vollständig gelesen und verstanden* wurde.

Die Zeitdauer von zwei Wochenstunden, die formal für das Praktikum angesetzt ist, wird in der Regel nicht zur vollständigen und richtigen Bearbeitung der geforderten Aufgaben ausreichen, so dass wesentliche Teile der Lösung außerhalb des Praktikums und vor dem Abgabetermin erarbeitet werden müssen.

Die Praktikumstermine dienen unter anderem zur Klärung der Aufgabenstellung und bieten Gelegenheit, Einzelfragen zum Stoff der Vorlesung mit dem Betreuer zu besprechen. Weiterhin ist die Diskussion der gewählten Lösungswege und die Festlegung von Verbesserungen und Berichtigungen ein wesentlicher Zweck der Veranstaltung.

Programmdokumentation Für die zu erstellenden Programme gilt ein *Mindeststandard* für die Programmdokumentation, der im Dokument „Bash Style Guide und Kodierungsrichtlinie“ [Meh14] erläutert ist. Nicht oder mangelhaft dokumentierte Programme werden nicht anerkannt.

Programmtests Grundsätzlich ist *jedes vorzulegende Programm* vorher vom Autor zu testen. Das geschieht in der Regel durch die sorgfältige Überprüfung der Lösung. Wenn Testfälle vorgegeben sind, müssen diese zur Vereinfachung der Kontrolle in der angegebenen Form verwendet werden.

Tests sind selbstverständlich auch dann durchzuführen, wenn Sie nicht ausdrücklich in der Aufgabenstellung gefordert werden.

Bewertung der Aufgaben

Praktikum Betriebssysteme 2

Aufgabe	Teil 1	Teil 2	Teil 3	Teil 4	Teil 5	Teil 6	Summe
1	3	3	2	2	-	-	10
2	2	2	2	2	2	-	10
3	2	2	3	3	-	-	10
4	6	4	-	-	-	-	10
5	1	1	1	1	3	3	10
6	6	4	-	-	-	-	10
7	7	3	-	-	-	-	10
							$\sum 70$

Tabelle 1: Auflistung der Teilpunkte

Ziel der Bearbeitung einer Aufgabe ist die vollständige Bearbeitung und Lösung der Aufgabenstellung. Bei der Testierung können jedoch Teilaufgaben anerkannt werden, so dass ein mangelhafter Anteil nicht das gesamte Testat in Frage stellt. Tabelle 1 listet die Punkte auf, die bei den einzelnen Teilaufgaben erzielt werden können. Die Nummerierung der Teilaufgaben stimmt mit den Abschnittsnummern überein (Aufgabe 3, Teil 2 findet sich in Abschnitt 3.2). Einige wenige Abschnitte sind nicht aufgeführt, weil diese Erläuterungen zu den davorstehenden Aufgabenteilen enthalten und somit keine eigene Wertung besitzen.

Gesamtpunktzahl Insgesamt können 70 Punkte erreicht werden. Mit **56 Punkten** (80 Prozent) ist die Bearbeitungspflicht erfüllt. *Es empfiehlt sich aber, auch nach Erreichen der nötigen Mindestpunktzahl die verbleibenden Aufgaben weiter zu bearbeiten, da die Inhalte des Praktikums auch klausurrelevant sind.*

Zur Darstellung

Programmcode, Programmausgaben, Programm- und Dateinamen, *Bash*-Schlüsselwörter und Menüeinträge erscheinen in Schreibmaschinenschrift mit fester Zeichenbreite.
In Codebeispielen und Listings werden für Code und Kommentar zur Verbesserung der Lesbarkeit unterschiedliche Schriftstile einer nichtproportionalen Schriftart verwendet: Schlüsselwörter sind halbfett und blau gesetzt (zum Beispiel `while` im Vergleich zu `varname`). In der gewählten Schriftart sind Null (`0`) und das große O (`O`) sowie die Eins (`1`) und das kleine L (`l`) gut unterscheidbar.

In den abgedruckten *Bash*-Skripten sind die Kopfkommentare aus Platzgründen meist weggelassen. In den zu erstellenden Lösungen müssen sie selbstverständlich vorhanden sein.

Titelseite Albrecht Dürer, „Ritter, Tod und Teufel“. Frühe Darstellung eines Systemadministrators: Der Beschützer des Guten, von bösen Mächten umringt. Der Ausgang der Unternehmung ist ungewiss …

Dieses Dokument wurde in LaTeX 2_ε unter Linux erstellt.

1 Linux-Grundlagen

1.1 Anmelden, Abmelden, KDE-Desktop **3 Punkte**

1.1.1 Anmeldung

Linux ist ein Mehrbenutzersystem. Zur Anmeldung als nichtprivilegierter Benutzer verwenden Sie die folgenden Angaben:

Benutzer	`student`
Passwort	`linux`

Danach erscheint die KDE-Benutzeroberfläche des Benutzers `student`. Sie befinden sich im Home-Verzeichnis `/home/student` dieses Benutzers.

Zur Anmeldung als Systemverwalter (Administrator, Benutzername `root`) verwenden Sie

Benutzer	`root`
Passwort	`linux`

Danach erscheint die KDE-Benutzeroberfläche des Benutzers `root`. Sie befinden sich jetzt im Home-Verzeichnis `/root` dieses Benutzers.

Da der Systemverwalter alle Rechte besitzt, verwenden Sie diesen Zugang nur dann, wenn es ausdrücklich gefordert ist.

1.1.2 Abmeldung, Rechner herunterfahren

In der linken unteren Ecke des Desktops öffnen Sie mit einem Mausklick das sogenannte K-Menü. Der unterste Eintrag erlaubt die Abmeldung. Als Alternative halten Sie auf dem Desktop-Hintergrund die rechte Maustaste gedrückt. Es erscheint ein Menü, das ebenfalls die Abmeldung ermöglicht.

Bei der Abmeldung werden mehrere Möglichkeiten angeboten. Die Abmeldung als Benutzer beendet die augenblickliche Sitzung und schließt alle Anwendungen. Danach kann eine neue Anmeldung erfolgen (zum Beispiel eines anderen Benutzers). Bei Betriebs- oder Praktikumsende wird der Rechner ausgeschaltet. Dafür ist die Auswahl *Rechner ausschalten* zu wählen. Danach werden alle Prozesse aller Benutzer beendet und der gesamte Rechenbetrieb durch eine Reihe einzelner Schritte geordnet eingestellt.

Auf keinen Fall darf zur Beendigung des Betriebs einfach der Strom abgestellt werden!

1.1.3 KDE-Desktop

Machen Sie sich zunächst mit einigen wichtigen Einrichtungen des KDE-Desktops vertraut.

Anwendung starten Viele Anwendungen sind über das bereits erwähnte K-Menü (linke untere Ecke) erreichbar. Die wichtigsten Anwendungen sind über Icons auf dem Desktop erreichbar.

Mehrere Desktops Es sind mehrere Desktops vorhanden, so dass Anwendungsfenster offen bleiben können. Ein anderer Desktop kann auf mehrere Arten erreicht werden:

Mittlere Maustaste Niederhalten der mittleren Maustaste, Auswahl der geöffneten Anwendung.

Arbeitsflächenumschalter in der Kontrollleiste am unteren Bildschirmrand.

Tastenkombination Strg + F1 : zur 1. Arbeitsfläche, Strg + F2 : zur 2. Arbeitsfläche, und so weiter

Dateibrowser *Dolphin* Der Browser *Dolphin* (Icon ) kann zur Navigation im Dateibaum verwendet werden.

Dateimanager *Midnight Commander* (`mc`) Siehe Abschnitt 1.1.4.

1.1.4 Dateimanager *Midnight Commander* (`mc`)

Der Dateimanager *Midnight Commander* ist eine halbgrafische Konsolenanwendung, die ein sehr schnelles Arbeiten mit Verzeichnissen und Dateien erlaubt. Der *Midnight Commander* steht funktionsgleich auch auf den nichtgrafischen Konsolen zur Verfügung.

Der Bildschirm des *Midnight Commander* ist in vier Teile gegliedert. Der meiste Platz des Fensters wird durch zwei Verzeichnisse eingenommen. Standardmäßig ist die zweite Zeile am unteren Rand des Fensters die Shell-Befehlszeile. Die untere Zeile zeigt die Belegung der Funktionstasten. Die oberste Zeile ist eine Menüleiste und wird über die Funktionstaste F9 erreicht. Die Funktionstasten sind in der folgenden Tabelle aufgelistet.

Funktionstaste	Wirkung	
F1	**Hilfe**	
F2	**Menü**	fortgeschrittene Einstellungsmöglichkeiten
F3	**Anzeige**	Inhalt einer Datei anzeigen
F4	**Bearbeiten**	Datei editieren
F5	**Kopieren**	Dateien oder Verzeichnisse in das jeweils andere Fenster/Verzeichnis kopieren
F6	**Umbenennen**	Dateien oder Verzeichnisse umbenennen
F7	**Mkdir**	ein neues Verzeichnis anlegen
F8	**Löschen**	Dateien oder Verzeichnisse löschen
F9	**Menüs**	die Menüs in der Kopfzeile anspringen
F10	**Beenden**	

Tabelle 1.1: Die Funktionstasten des *Midnight Commanders*

Eine hohe Bediengeschwindigkeit ist nur durch Menübenutzung nicht möglich. Ein wesentliches Bedienkonzept ist deshalb bei vielen Anwendungen die Verwendung von Tastenkombinationen. Die wichtigsten Tastenkombinationen sind in der folgenden Tabelle wiedergegeben.

Tastenkombination	**Wirkung**
⇆	zum anderen Verzeichnis wechseln
Einfg	Datei oder Verzeichnis markieren
Alt + t	Darstellung wechseln: kurz, lang, benutzerdefiniert
Strg + \	Verzeichnis-hotlist zeigen und verwenden
+	Dateigruppe auswählen
-	Dateigruppe abwählen
Strg + o	Verzeichnisanzeige ein-/ausschalten
Alt + h	„History“ der Kommandozeile des aktuellen Fensters
Alt + H	Verzeichnis-Chronik aufrufen (zur Navigation)

Tabelle 1.2: Einige Tastenkombinationen des *Midnight Commanders*

Der Browser *Dolphin* (Icon) kann ebenfalls mit zwei Fenstern betrieben werden. Dazu ist der Menüpunkt *Einstellungen / Ansichtsprofil laden / Midnight Commander* auszuwählen.

> Machen Sie sich mit dem Desktop, dem *Dolphin* und insbesondere mit dem *Midnight Commander* so weit vertraut, dass Sie sicher mit Dateien und Verzeichnissen umgehen können.

1.1.5 Grundlegende Bedienmöglichkeiten der *Bash*-Shell

In den folgenden Abschnitten wird das Arbeiten mit der *Bash*-Shell erlernt. Tabelle 1.3 zeigt eine Auswahl der wichtigsten Bedienmöglichkeiten.

Bedienmöglichkeit (Auswahl)	**Befehl / Tastenkombination**
Blättern	⇑ + Bild ↑ , ⇑ + Bild ↓
Kommando- und Dateinamenergänzung	⇆
alle Kommando- und Dateinamenergänzungen	⇆ ⇆
Shell beenden	Strg + d
Programm abbrechen	Strg + c
Blättern in der history-Liste (Kommandowiederholung)	↑ , ↓
Suchen in der history-Liste (inkrementell)	Strg + r
Eingabeumleitung aus einer Datei	< *Dateiname*
Ausgabeumleitung in eine Datei	> *Dateiname*
Pipe (Befehlsverkettung)	*Befehl* | *Befehl*

Tabelle 1.3: Wichtige Bedienmöglichkeiten der *Bash*-Shell

1.2 *Unix*-Handbuchseiten und `help` **3 Punkte**

1.2.1 Hilfe zu Shell-Befehlen – manual pages zu eigenständigen Programmen

Online-Hilfe steht für Kommandos und viele Anwendungen durch die Handbuchseiten zur Verfügung. Wenn keine grafische Oberfläche zur Verfügung steht (zum Beispiel Server), können die Handbuchseiten im Textmodus mit Hilfe der Konsolenanwendung `man` eingesehen werden. Tabelle 1.4 zeigt die wichtigsten Hotkeys für den von `man` gestarteten Dateibetrachter. Rufen Sie die Handbuchseite für das Kommando `ls` (ls - list directory contents) auf:

```
man ls
```

Stellen Sie für das Kommando `ls` die Bedeutung des Schalters `-l` fest und geben Sie eine entsprechende Verzeichnisliste in einem Shell-Fenster aus.

Bedienmöglichkeit	**Befehl / Tastenkombination**
Suchen	*/Suchbegriff*
nächster Vorkommen des Suchbegriffes	`n` (**n**ext)
vorheriges Vorkommen des Suchbegriffes	`N` (**N**ext)
Verlassen	`q` (**q**uit)

Tabelle 1.4: Die wichtigsten Bedienmöglichkeiten von `man`

Für das Programm `man` (Programm zum Einsehen der Online-Handbücher) existiert auch ein Handbuch. Es kann mit dem Befehl

```
man man
```

eingesehen werden. Die Handbuchseiten können unter einer grafischen Oberfläche auch mit dem Browser *Konqueror* eingesehen werden. Geben Sie in der Adresszeile den Befehl `man:`*Befehl* ein (zum Beispiel `man:man`). (In Dolphin funktioniert das nicht.)

Die Handbuchseiten sind in mehrere Kataloge gegliedert. Ermitteln Sie mit Hilfe von `man man` die Themen der Kataloge mit den folgenden Nummern:

man 1 : _____

man 2 : _____

man 3 : _____

man 5 : _____

man 8 : _____

Die Handbuchseiten bieten Hilfe für Befehle, die als eigenständige, ausführbare Programme oder Skripte vorhanden sind (zum Beispiel `ls` oder `mc`).

1.2.2 Hilfe zu Shell-Builtins

Eine Reihe von Befehlen ist direkt in der Shell eingebaut. Für diese Befehle steht der Shell-Befehl `help` zur Verfügung. Der Befehl `cd` (change directory) erlaubt den Wechsel des Verzeichnisses. Der Versuch, mit `man cd` eine Handbuchseite aufzurufen, schlägt fehl, während

```
help cd
```

Auskunft gibt.

In einigen Fällen gibt es zu Shell-Befehlen namensgleiche Builtins. `printf` ist so ein Fall. Lesen Sie die Hilfe zum Builtin `type` und stellen Sie damit fest, welche Versionen von `printf` auf Ihrem Rechner vorhanden sind und von welcher Art diese sind.

1.3 Lokalisierung der Shell **2 Punkte**

1. Ermitteln Sie den Zweck des Befehls `printf` aus dem Handbuch 1 und den Zweck der gleichnamigen Funktion `printf` aus Handbuch 3 :

   `printf` (man 1) _____

   `printf` (man 3) _____

   Geben Sie in einem Shell-Fenster als nichtprivilegierter Benutzer mit Hilfe des Shell-Befehls `printf` nacheinander aus:

   - Eine Zeichenkette (zum Beispiel `Hallo, Welt!`),
   - die vierstellige ganze Zahl 4711 mit einer Breite von 8 Zeichen,
   - die reelle Zahl 47,11 mit einer Breite von 8 Zeichen und 2 Nachkommastellen
   - die reelle Zahl 47.11 mit einer Breite von 8 Zeichen und 2 Nachkommastellen.

2. - Führen Sie nun die letzten beiden Versuche als Benutzer `root` durch. Melden Sie sich dazu mit dem Befehl `su` im Shell-Fenster des nichtprivilegierten Benutzers mittels login shell an (erforderlichen Schalter im Handbuch nachsehen). Was stellen Sie fest ?
   - Vergleichen Sie danach in beiden Fällen die Ausgaben, die der Shell-Befehl `locate` anzeigt; dazu müssen Sie zunächst mit

     ```
     sudo zypper in mlocate
     ```

     das Programm `locate` nachinstallieren.

1.4 Dateibaum erzeugen und löschen **2 Punkte**

Öffnen Sie ein Shell-Fenster und erzeugen Sie mit Hilfe der Befehle `mkdir` und `cd` den Dateibaum in Abbildung 1.1. Die Dateien `test*.dat` werden als leere Dateien durch den Befehl `touch` erzeugt, zum Beispiel

```
touch test0101.dat test0102.dat
```

Vorgehensweise:

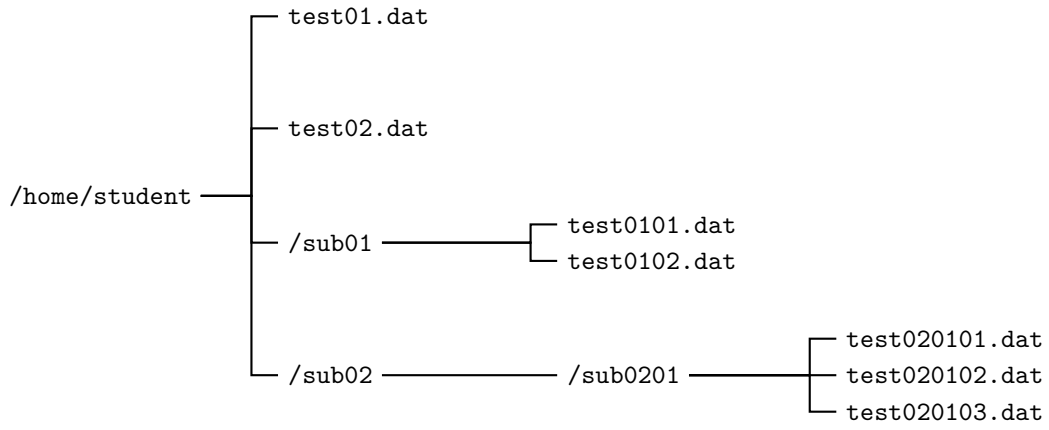

Abbildung 1.1: Zu erzeugender Dateibaum

- Kontrollieren Sie nach jedem Schritt mittels `ls`.
- Sehen Sie sich danach das Endergebnis mit einem Dateimanager an (`mc`, `konqueror`).
- Beseitigen Sie danach die Dateien einzeln mittels `rm`.
- Beseitigen Sie dann die Verzeichnisse einzeln mittels `rmdir`.

Führen Sie die Schritte dieses Abschnittes einzeln aus und notieren Sie sich die korrekten Befehle und deren Schalter als **Ergebnisprotokoll zur Vorlage bei der Testierung dieses Aufgabenblattes**.

2 *Unix*-Kommandos

Erstellen Sie für jede der folgenden Teilaufgaben ein eigenes Skript (`loesung-2-1.sh`, `loesung-2-2.sh`, ...), auch wenn die Lösung nur aus einer Zeile bestehen sollte. Diese Vorgehensweise erleichtert die Abgabe, dokumentiert die Lösung und erspart Tipparbeit.

2.1 Dateiinhalte filtern **2 Punkte**

Listing 2.1: Datei `phone.book`

```
Hans!768760!Willy-Brand-Platz 12
Hans-Werner!12780!Ernst-August-Allee 37
Hans-Dieter!88773312!Georg-Friedrich-Händel-Weg 233
Georg!7772221!Herzog-Wolfgang-Steig 87
Hase!76428765!Wolfsburger Str. 55
Bert!7665654!Haselünner Str. 65
Wolf!8595485!Brunnenweg 77
```

Die Datei `phone.book` (Listing 2.1) enthält ein winziges Telefonverzeichnis mit den Spalten *Name*, *Telefonnummer* und *Straße*. Die Spalten sind durch das Zeichen `!` getrennt. Die Datei wurde automatisch erzeugt. Das Format kann als gegeben betrachtet werden, das heißt die Darstellung muss nicht weiter überprüft werden.

Machen Sie sich zum Lösen der folgenden Aufgabenteile insbesondere mit den Werkzeugen `sort`, `cat`, `cut`, `tr`, `head`, `grep` und `column` vertraut. Lesen Sie die dazugehörigen Handbuchseiten und verschaffen Sie sich einen Überblick über die vorhandenen Schalter. Lösen Sie folgende Aufgaben:

1. Sortieren der Datei nach den Namen (1. Spalte).
2. Sortieren der Datei nach fallenden Telefonnummern (2. Spalte).
3. Nur die Straßen ausgeben (3. Spalte).
4. Alle Ausrufezeichen in Doppelpunkte umwandeln.
5. Nur die erste Zeile der Datei ausgeben.
6. Nur die Zeile mit den Namen `Hans` und `Wolf` ausgeben. Stellen Sie sicher, dass genau diese beiden Namen in der ersten Spalte gesucht werden.

Hinweis. Bei den ersten drei Aufgaben fällt die Kontrolle schwer, weil keine Formatierung vorhanden ist. Ergänzen Sie deshalb die Sortieranweisungen in der folgenden Art, um eine tabellenartige Ausgabe zu erreichen:

```
sort ...  | column -t -s!
```

Was ist die Bedeutung der Schalter `-t` und `-s`?

2.2 Auswertung einer Datendatei **2 Punkte**

Listing 2.2: Beginn der Datei `results.csv`

```
C7300,ELLIOT,GARY,D,80.1,CA18
C8801,DEARAUJO,KRINESH,C,73,CA18
C9001,JEFFERIES,DIANNE,D,83.4,CA18
C9201,HUAT,KENNETH,D,83.62,CA18
```

Listing 2.2 zeigt den Beginn der Datei `results.csv`. Die Einträge innerhalb einer Zeile sind durch Kommas getrennt (csv, comma separated values) und haben folgende Bedeutung:

Matrikelnummer, Nachname, Vorname, Note, Punkte, Abschluss

1. Bestimmen Sie die Anzahl der Studierenden in dieser Datei. Achtung: die Datei kann Leerzeilen enthalten!
2. Erzeugen Sie eine aufsteigend sortierte Liste aller vorkommenden Matrikelnummern.
3. Erzeugen Sie eine nach Namen sortierte Liste aller Studierenden mit der Note `HD`. Beachten Sie dabei, dass die Zeichenfolge `HD` Bestandteil eines Namens oder einer Abkürzung sein könnte. Es reicht also keinesfalls, nur nach `HD` zu suchen. Verwenden Sie zur Kontrolle der Ausgabe `column` (siehe Aufgabe 2.1).

2.3 Alle Benutzer des Rechners ermitteln **2 Punkte**

Informieren Sie sich mittels `man 5 passwd` über das Format der zentralen Passwortdatei `/etc/passwd`. Erstellen Sie aus dieser Datei mit Hilfe von `cut` eine Datei `benutzer`, die die Liste aller aufsteigend sortierten Benutzernamen enthält.

Befehl: _____

2.4 Liste von Verzeichnissen **2 Punkte**

Erstellen Sie mit Hilfe von `du` und `sort` eine nach fallender Größe (in Bytes) geordnete Liste aller Verzeichnisse unterhalb von `/usr`. Die jeweils angezeigte Verzeichnisgröße soll nicht die Größe der Unterverzeichnisse enthalten. Zeigen Sie die Liste mit dem Pager `less` an.

Befehl: _____

2.5 Sortierte Liste von Verzeichnissen **2 Punkte**

Erstellen Sie mit Hilfe von `ls` eine nach fallender Größe geordnete Liste aller Dateien im umfangreichsten Verzeichnis unterhalb von `/usr` (wurde in Abschnitt 2.4 ermittelt). Zeigen Sie die Liste mit dem pager `less` an.

Befehl: _____

Testen Sie Ihren Befehl vorher an einem kleinen, selbsterstellten Verzeichnisbaum und überprüfen Sie sorgfältig das Ergebnis!

3 Prozesse, Dateien, Shell

3.1 Grundlegende Informationen zu Benutzern **2 Punkte**

Stellen Sie mit Hilfe des Kommandos `groups` fest, zu welchen Gruppen `root` gehört:

Gruppen: _____

Melden Sie sich in einem neuen Shell-Fenster als Benutzer an (`su student`). Stellen Sie fest, zu welchen Gruppen der Benutzer `student` gehört:

Gruppen: _____

Stellen Sie mit Hilfe des Kommandos `id` für `root` die numerische Benutzernummer (uid), die numerische Gruppennummer (gid) fest und die Gruppennummern der Mitgliedsgruppen fest:

uid: _____ **gid:** _____

Gruppen: _____

Nummern: _____

3.2 Grundlegende Informationen über Prozesse **2 Punkte**

- Melden Sie sich gegebenenfalls unter der grafischen Oberfläche ab. Wechseln Sie zur ersten alphanumerischen Konsole (Strg + Alt + F1) und melden Sie sich als Benutzer `root` an.
- Wechseln Sie zur zweiten alphanumerischen Konsole (Alt + F2) und melden Sie sich als Benutzer `student` an; rufen Sie den *Midnight Commander* `mc` auf.
- Wechseln Sie zur dritten alphanumerischen Konsole (Alt + F3) und melden Sie sich ebenfalls als Benutzer `student` an; rufen Sie die Handbuchseite von `ps` auf.
- Wechseln Sie nun zurück zur ersten Konsole (Alt + F1).

Erläutern Sie die Wirkung folgender Kommandos:

Kommando	Wirkung	Benutzer
`ps`	_____	`root`
`ps`	_____	`student`
`ps a`	_____	`root`
`ps au`	_____	`root`
`ps x`	_____	`root`
`ps aux`	_____	`root`

Stellen Sie mit Hilfe des Befehls `wc` fest, wieviele Prozesse augenblicklich laufen. Beachten Sie, dass `ps` ohne weitere Maßnahme eine Kopfzeile ausgibt, die nicht mitgezählt werden darf.

Anzahl laufender Prozesse: _____

Starten Sie die Anwendung `top` um eine laufende Anzeige der Prozessliste zu erhalten. `top` wird durch Eingabe des Zeichens `q` beendet.

Starten Sie auf einer anderen Konsole folgenden Prozess

```
yes  >  /dev/null &
```

Beobachten Sie die Ausgabe von `top` und stellen Sie die Prozessnummer von `yes` fest:

PID: _____

Brechen Sie den Prozess mit dem Kommando `kill` *`PID`* ab.

Wechseln Sie zurück zur grafischen Oberfläche (Strg + Alt + F7), melden Sie sich als `root` an und starten Sie den `yes`-Prozess erneut in einem Shell-Fenster.

Rufen Sie die Anwendung `Systemmonitor` (K-Menü→System→Überwachung→Systemmonitor) auf und beenden Sie den `yes`-Prozess.

Ermitteln Sie den Zweck des Befehls `yes` und erläutern Sie die oben verwendete Befehlszeile.

3.3 Shell

3 Punkte

3.3.1 Prozesse im Vordergrund und Hintergrund

Führen Sie folgende Befehle in einem Shell-Fenster aus:

```
sleep 10
sleep 10 &
```

Wie verhalten sich der Prompt und die Shell ?

Listen Sie rekursiv alle Dateien unterhalb von `/usr` und leiten Sie die Ausgabe in eine Datei um:

```
ls -lR  /usr >  usr.list   &
ps
fg
```

Holen Sie den Prozess (solange er noch läuft!) mit dem letzten Kommando in den Vordergrund.

3.3.2 Einfache Shell-Muster zur Beschreibung von Dateinamen (globbing)

Ermitteln Sie zunächst den Hauptzweck des Werkzeuges `touch`, indem Sie die zugehörige Handbuchseite lesen. Wenden Sie `touch` auf eine Testdatei an und überprüfen Sie die Wirkung.

Erzeugen Sie in einem Unterverzeichnis mittels `touch` die folgenden 14 Dateien:

```
feb86    jan12.89
jan19.89 jan26.89
jan5.89  jan85    jan86 jan87
jan88    mar88    memo1 memo10
memo2    memo2.sv
```

Führen Sie die folgenden Befehle aus. Erläutern Sie die Befehle und deren Ergebnisse.

```
echo *
echo *[^0-9]
echo m[a-df-z]*
echo jan*
echo *.*
echo ?????
echo *89
echo jan?? feb?? mar??
echo [fjm][ae][bnr]*
```

3.3.3 Dateien mit besonderen Namen

Erzeugen Sie Dateien mit den folgenden Namen:

```
stars* stars1 stars2
-top
hello my friend
"goodbye"
```

Kommandos zum Anlegen der Dateien:

Löschen Sie nun diese Dateien wieder. Befehle zum Löschen der Dateien:

3.3.4 Sortierreihenfolge und Lokalisierung

Die Bash-Shell für Benutzer-Logins ist auf neueren Linux-Systemen lokalisiert, das heißt Fehler- und Meldungstexte der Shell und vieler Werkzeuge werden in der Landessprache ausgegeben. Die

zur Lokalisierung gehörenden Einstellungen werden durch den Befehl `locale` ermittelt. Die erste Zeile der Ausgabe zeigt die Spracheinstellung (`LANG=de_DE.UTF-8`: Deutsch für Deutschland, 8-Bit-Unicode-Codierung), die fünfte Zeile die zugrundeliegende Sortiereinstellung (`LC_COLLATE=...`):

```
LANG=de_DE.UTF-8
LC_CTYPE="de_DE.UTF-8"
LC_NUMERIC="de_DE.UTF-8"
LC_TIME="de_DE.UTF-8"
LC_COLLATE="de_DE.UTF-8"
 ...
```

Ermitteln Sie zunächst die **Spracheinstellung** für die Benutzer `root` und `student`:

`root` _____

`student` _____

Legen Sie nun mittels `touch` in einem leeren Unterverzeichnis Dateien mit folgenden Namen an:

`a b c x y z A B C X Y Z ä ö ü Ä Ö Ü`

Führen Sie zunächst als Benutzer `root` die folgenden Befehle aus und notieren Sie die Ergebnisse.

Befehl (als `root`)	**Ergebnis**
`echo *`	
`echo [a-z]*`	
`echo [A-Z]*`	
`find *`	

Führen Sie die gleichen Befehle als Benutzer `student` aus und notieren Sie ebenfalls die Ergebnisse.

Befehl (als `student`)	**Ergebnis**
`echo *`	
`echo [a-z]*`	
`echo [A-Z]*`	
`find *`	

Vergleichen Sie die Ergebnisse und beschreiben Sie die Unterschiede.

3.3.5 Ein-/Ausgabeumleitung

1. Erzeugen Sie eine Liste der Benutzernamen aus dem ersten Feld der Datei `/etc/passwd`. Die Liste soll in alphabetischer Reihenfolge sortiert sein. Verwenden Sie dazu `sort` und `cut`. Das Ergebnis wird durch *Ausgabeumleitung* in die Datei `benutzerliste.txt` geschrieben.
2. Ermitteln Sie die Anzahl der Zeilen in der Datei `/etc/passwd`, die die Zeichenkette `bash` enthalten. Verwenden Sie dazu nur `grep`:
   - Versorgen Sie `grep` mittels *Eingabeumleitung* mit dem Inhalt der Datei `/etc/passwd`.
   - Übergeben Sie den Dateinamen `/etc/passwd` als Kommandozeilenparameter an `grep`.
   - Überprüfen Sie das Ergebnis indem Sie die Datei `/etc/passwd` im Editor (nur Lesen) öffnen und die Anzahl der Benutzer abzählen.

3.4 Shell-Programmierung **3 Punkte**

Erstellen Sie das Skript `create_dirs.sh` (Listing 3.1).

Listing 3.1: Skript `create_dirs.sh`

```
#!/bin/bash -
#===============================================================================
#
#          FILE: create_dirs.sh
#
#         USAGE: ./create_dirs.sh number_of_dirs
#
#   DESCRIPTION: Unterverzeichnisse d0 d1 ... erzeugen
#
#       OPTIONS: ---
#  REQUIREMENTS: ---
#          BUGS: ---
#         NOTES: ---
#        AUTHOR: Vorname Nachname, Nachname.Vorname@fh-swf.de
#  ORGANIZATION: Fachhochschule Südwestfalen, Iserlohn
#       CREATED: 26.08.2013 18:40
#      REVISION:  ---
#===============================================================================

#-------------------------------------------------------------------------------
#  Aufruf / Anzahl der Aufrufparameter überprüfen
#-------------------------------------------------------------------------------
if [ ${#} -lt 1 ]
then
  echo -e "\n\tAufruf: $0 Anzahl_der_Verzeichnisse\n"
  exit 1
fi

#-------------------------------------------------------------------------------
#  Verzeichnisse anlegen und zählen
#-------------------------------------------------------------------------------
anzahl=0
erfolg=0
while [ $anzahl -lt $1 ]
do
  mkdir d$anzahl && ((erfolg++))
  ((anzahl++))
done

#-------------------------------------------------------------------------------
#  Kontrollausgabe
#-------------------------------------------------------------------------------
echo -e "\n${erfolg}/${anzahl} Verzeichnisse angelegt\n"
```

Machen Sie die Datei ausführbar und erzeugen sie 20 Verzeichnisse mittels

```
chmod +x  create_dirs.sh
./create_dirs.sh  20
```

Kontrollieren Sie, ob alle Verzeichnisse vorhanden sind.

Erstellen Sie nun das Skript `mehrfachkopieren.sh` dessen **Anfang** in Listing 3.2 angegeben ist.

Listing 3.2: Anfang des Skripts `mehrfachkopieren.sh` (einige zu erstellende Zeilen ausgeblendet)

```
1  #!/bin/bash -
2  #================================================================================
3  #
4  #          FILE: mehrfachkopieren.sh
5  #
6  #         USAGE: ./mehrfachkopieren.sh number_of_dirs
7  #
8  #   DESCRIPTION: Datei (Kommandozeilenparameter) in alle vorhandenen
9  #                Unterverzeichnisse kopieren
10 #
11 #       OPTIONS: ---
12 #  REQUIREMENTS: ---
13 #          BUGS: ---
14 #         NOTES: ---
15 #        AUTHOR: Vorname Nachname, Nachname.Vorname@fh-swf.de
16 #  ORGANIZATION: Fachhochschule Südwestfalen, Iserlohn
17 #       CREATED: 26.08.2013 18:40
18 #      REVISION:  ---
19 #================================================================================
20
21 #-------------------------------------------------------------------------------
22 #  Aufruf / Anzahl der Aufrufparameter überprüfen
23 #-------------------------------------------------------------------------------
24
25 #-------------------------------------------------------------------------------
26 #  Prüfen, ob die zu kopierende Datei existiert
27 #-------------------------------------------------------------------------------
28
29 #-------------------------------------------------------------------------------
30 #  Unterverzeichnisse ermitteln
31 #-------------------------------------------------------------------------------
32 liste=$(find -type d -name "[^.]*")
```

Ermitteln Sie die Bedeutung des Zusatzes `-name "[^.]*"` im `find`-Befehl in Zeile 32. Das kann der Einfachheit halber außerhalb des Skriptes in einem Shell-Fenster geschehen.

mit `-name` _____

ohne `-name` _____

Bedeutung _____

Weitere Vorgaben:

- Die Datei muss, wie jedes Skript, einen Kopfkommentar enthalten (siehe Listing 3.2).
- Die an das Skript übergebenen sind am Anfang des Skripts auf Vollständigkeit und Gültigkeit zu überprüfen.
- Dateien und Verzeichnisse, die als Parameter übergeben werden, sind auf Vorhandensein und Lesbarkeit (Rechte) zu überprüfen.
- Die Anzahl der kopierten Dateien wird gezählt und am Ende des Skriptes ausgegeben.

Erstellen Sie nun ein Skript `mehrfachloeschen.sh`, welches eine Datei (Name = 1. Kommandozeilenparameter) in allen nicht versteckten Unterverzeichnissen löscht. Der Parameter wird wie oben überprüft. Die Anzahl der gelöschten Dateien wird gezählt und am Ende des Skriptes ausgegeben.

Erstellen Sie weiterhin ein Skript `remove_dirs.sh`, welches alle nicht versteckten Unterverzeichnisse und alle darin enthaltenen Dateien und Verzeichnisse löscht. Die Anzahl der gelöschten Unterverzeichnisse wird gezählt und am Ende des Skriptes ausgegeben.

4 Benutzerverwaltung

4.1 Mehrere Benutzer mit einem Shell-Skript einrichten **6 Punkte**

Die Datei `user+num.txt` enthält 50 Namen und Matrikelnummern neuer Studentinnen und Studenten. Mit dem zu entwickelnden Skript `user-add.sh` soll diese Datei eingelesen werden und danach mit Hilfe dieser Informationen automatisch alle Benutzerzugänge und `home`-Verzeichnisse erzeugt werden. Die benötigten Passwörter werden ebenfalls automatisch erzeugt und zur weiteren Verwendung zusammen mit den login-Namen der neuen Benutzter in eine Ausgabedatei geschrieben. Der Vorgang besteht aus mehreren Schritten, die zunächst einzeln entwickelt und erprobt werden müssen.

Diese Aufgabe muss als Benutzer **`root`** ausgeführt werden. Richten Sie zunächst Ihre Arbeitsumgebung ein (Editor, Datei-Browser), falls das nicht nicht geschehen ist.

4.1.1 Klartextpasswörter erzeugen

Installieren Sie als Benutzer `root` mit `zypper in pwgen` das Paket `pwgen` zur automatischen Erzeugung von Passwörtern. Das Handbuch von `pwgen` informiert über den Aufruf und die Kommandozeilenparameter. Erzeugen Sie zunächst 10 Passwörter der Länge 8:

```
pwgen 8 10
```

4.1.2 Entwicklung des Skriptes `user-add.sh`

Das zu entwickelnde Skript `user-add.sh` soll die Datei `user+num.txt` einlesen und jeden Eintrag einzeln verarbeiten. Folgende Vorgaben sind zu beachten:

Benutzer `root`	Es wird überprüft, ob das Skript vom Benutzer `root` (UID 0) aufgerufen wird. Wenn das nicht der Fall ist, wird das Skript abgebrochen.
Eingabedatei	Die Lesbarkeit der Eingabedatei ist zu überprüfen.
Klartextpasswort der Länge 8 erzeugen	Das Passwort wird an eine Variable `password` zugewiesen.
login-Name erzeugen	Der Nachname jedes Benutzers wird in Kleinbuchstaben umgewandelt und an die Variable `loginname` als Login-Name zugewiesen.

Benutzer anlegen	Mit Hilfe des Befehls `useradd` wird ein Benutzer und dessen `home`-Verzeichnis angelegt (siehe Handbuch). Der Login-Name steht in der Variablen `loginname`. Die Matrikelnummer wird als Kommentar in `useradd` übernommen. Der Schalter `-p` (beziehungsweise `--password`) von `useradd` darf nicht verwendet werden, weil sonst das (verschlüsselte) Passwort für andere Benutzer in der Prozessliste sichtbar würde. Wenn der Benutzer erfolgreich angelegt wurde, wird das Passwort durch `echo ${loginname}:${password} | chpasswd 2>/dev/null` gesetzt. Ermitteln Sie die Bestandteile dieser Zeile und erklären Sie die Wirkungsweise. (`chpasswd` ist für den Einsatz in Shell-Skripten gedacht.)
Benutzereinrichtung überprüfen	Die fehlerfreie Ausführung von `useradd` wird kontrolliert und gegebenenfalls ein Erfolgszähler erhöht.
Ausgabe	Die Angaben Nachname, Vorname, Matrikelnummer, login-Name und Klartextpasswort jedes Benutzers werden in die Datei `user-added-JJJJMMTT-HHMMSS.txt` geschrieben. Der Namensanteil `JJJJMMTT-HHMMSS` ist der Zeitpunkt, zu dem diese Datei angelegt wurde (Zeitstempel).
Statistik	Am Skriptende wird die Anzahl der versuchten Benutzereinrichtungen (Schleifendurchläufe) und die Anzahl der erfolgreichen Benutzereinrichtungen ausgegeben.

4.2 Mehrere Benutzer mit einem Shell-Skript löschen **4 Punkte**

Entwickeln Sie ein Skript `user-delete.sh`, welches die Datei `user-added-JJJJMMTT-HHMMSS.txt` (siehe oben) einliest und alle darin enthaltenen Benutzer und deren `home`-Verzeichnisse löscht. Hierzu muss das Kommando `userdel` verwendet werden.

Das Skript erzeugt eine Protokolldatei `user-removed-JJJJMMTT-HHMMSS.txt`, in der Nachname, Vorname, Matrikelnummer und login-Name der erfolgreich gelöschten Benutzer eingetragen werden.

Hinweise

- Überprüfen Sie kritische Befehle sorgfältig, indem Sie diese in Ihrem Skript *nicht* sofort in der ausführbaren Form, wie zum Beispiel

  ```
  userdel -r $loginname
  ```

  angeben, sondern diese mit `echo` zunächst nur als Text ausgeben lassen:

  ```
  echo  "userdel -r $loginname"
  ```

- Testen Sie Ihre Skripte zunächst mit einer Eingabedatei, die nur drei Benutzerangaben enthält. Schreiben Sie sich die Namen und die erzeugten drei Passwörter auf und prüfen Sie, ob eine grafische Anmeldung möglich ist.

5 Handhabung verschiedener Dateisysteme

Einführung

a) Besitzer, Gruppe, Zugriffsrechte

Jeder Datei bzw. jeder Ordner hat unter Linux einen Besitzer (owner, user) und eine Besitzergruppe (group) – Dateien in Ihrem Home-Verzeichnis (`/home/student`) gehören z. B. Ihnen (dem Benutzer `student`) und zur Gruppe `users`.

Auf Dateien können Sie lesend (read, r), schreibend (write, w) oder ausführend (execute, x) zugreifen, wenn passende Zugriffsrechte gesetzt sind. Die Rechte sowie die Eigentümerangaben sehen Sie in der Ausgabe von `ls -l` am linken Rand und in den Spalten 3 und 4. Zum Beispiel steht in

```
-rwxr--r--  1 student users 1121 Nov 15 11:14 test.sh
```

der Block `rwx` dafür, dass der Dateibesitzer volle Zugriffsrechte (r+w+x = lesen und schreiben und ausführen) hat, während Mitglieder der Gruppe `users` nur lesen dürfen (`r--`). Der letzte Block (hier auch `r--`) räumt Leserechte zudem für Benutzer ein, die weder der Eigentümer sind noch zur Gruppe `users` gehören.

Mit dem Kommando `chmod` (change mode) können Sie die Zugriffsrechte ändern. Um das Kommando auf eine Datei oder einen Ordner anwenden zu dürfen, müssen Sie entweder der Besitzer oder der Administrator `root` sein. Das Programm erlaubt verschiedene Schreibweisen, um die neu zugewiesenen Rechte anzugeben; für den Anfang reicht es aus zu wissen, dass Sie ein Recht mit `+` vergeben und mit `-` entziehen können – auf welche Benutzer sich das bezieht, geben Sie vor dem Plus/Minus an (`u` = user, Besitzer; `g` = group, Gruppe; `o` = others, sonstige). Und um welches Recht es geht, schreiben Sie hinter das Plus/Minus (`r` = read, `w` = write, `x` = execute).

Die Befehle `chown` (change owner) und `chgrp` (change group) ändern Besitzer und Gruppenzugehörigkeit und können nur von `root` ausgeführt werden. (Außerdem kann der Dateibesitzer – mit Einschränkungen – über `chgrp` auch die Gruppenzugehörigkeit eigener Dateien ändern.) Die Syntax ist jeweils

```
chown username datei/verzeichnis
chgrp gruppenname datei/verzeichnis
```

Darüber hinaus kann `chown` auch beide Eigenschaften in einem Rutsch ändern, indem Sie Username und Gruppenname durch einen Doppelpunkt getrennt angeben:

```
chown username:gruppenname datei/verzeichnis
```

Die Tools `chmod`, `chown` und `chgrp` arbeiten beide auf Wunsch auch rekursiv (für alle Unterordner und enthaltenen Dateien), wenn Sie es mit einer geeigneten Option aufrufen; Details verraten die Manpages.

Das Thema Zugriffsrechte greifen wir in der Vorlesung noch mal ausführlicher auf.

b) suid/sgid

Für Dateien kann ein `suid`- (set user ID) oder `sgid`-Bit (set group ID) gesetzt sein. Das führt, wenn es sich um ausführbare Programmdateien handelt, dazu, dass das Programm immer mit den Rechten des Dateibesitzers bzw. mit den Rechten der Besitzergruppe läuft. Ein Beispiel dafür ist das Programm `passwd`, welches das Passwort eines Benutzers ändert: Es kann von normalen Benutzern aufgerufen werden, um deren eigene Passworte zu ändern, muss dafür aber die Dateie `/etc/shadow` ändern, für die nur `root` Lese- und Schreibrechte hat. Darum ist bei `/usr/bin/passwd` das `suid`-Recht gesetzt, und das Programm gehört `root`. Sie erkennen gesetzte Bits daran, dass in der Besitzer- bzw. Gruppenspalte der Ausgabe von `ls -l` das `x` durch ein `s` ersetzt ist:

```
student@linux:~> ls -l /usr/bin/passwd
-rwsr-xr-x 1 root shadow 51200 Sep 27  2013 /usr/bin/passwd
```

c) Symbolische Links

Ein symbolischer Link erlaubt es, über einen anderen Namen auf eine Datei oder ein Verzeichnis zuzugreifen. Technisch ist das nur eine kleine Spezialdatei, die den Pfad zur Datei / zum Verzeichnis speichert. Sie erkennen symbolische Links in der Ausgabe mit `ls -l` daran, dass in der Rechtespalte ganz am Anfang ein kleines L („`l`“) steht (wo normal ein „`-`“ für Dateien oder ein „`d`“ für Verzeichnisse steht). Sie erzeugen solche symbolischen Links mit `ln -s` (siehe Aufgabe 5.2); wenn Sie die Option `-s` (symbolisch) vergessen, erzeugt das Tool stattdessen einen Hard link – die Unterschiede werden wir in der Vorlesung besprechen.

d) Ausgabeumleitung

Die Umleitung der *Standardausgabe* eines Programms mit `>` haben wir bereits in der Vorlesung gesehen. Daneben lässt sich auf die *Standardfehlerausgabe* mit `2>` umleiten: Darüber landen Fehlermeldungen in einer Datei. Verwenden Sie `>` und `2>` gleichzeitig, erzeugt ein Programm gar keine Ausgabe im Terminal mehr, und die angegebenen zwei Dateien enthalten getrennt die regulären Ausgaben und die Fehlermeldungen.

5.1 Dateirechte **1 Punkte**

Kopieren Sie als `root` das Verzeichnis `/usr/share/man` rekursiv in das Home-Verzeichnis `/home/student` des Standardbenutzers `student`. Für keine der neuen Dateien hat `student` Schreibrechte.

- Setzen Sie für alle Dateien und Verzeichnisse des Unterbaumes Schreibrechte für alle (Benutzerklassen `user`, `group` und `other`) – finden Sie über die Manpage heraus, wie Sie das mit `chmod` mit einem einzigen Kommando erledigen.
- Setzen Sie für alle Dateien und Verzeichnisse des Unterbaumes als Benutzer und Gruppe `student` / `users`.

5.2 Symbolische Links

1 Punkte

Erzeugen Sie eine leere Datei `myfile1` und einen symbolischen Link `myfile2` :

```
1  touch myfile1
2  ln -s myfile1 myfile2
3  ls -l
4  cat > myfile1
5  xxxxxxx
6  yyyyyyy
7  zzzzzzzz
8  ^C
9  ls -l my*
10 cat myfile1
11 cat myfile2
12 cat >> myfile2
13 1111111
14 2222222
15 33333333
16 ^C
17 ls -l my*
18 rm myfile1
19 ls -l my*
```

- Erläutern Sie zunächst die Wirkung der unterschiedlichen `cat`-Befehle.

  _____

  _____

- Wie sind Links in der Verzeichnisliste (`ls -l`) gekennzeichnet: _____
- Erläutern Sie die Wirkung der beiden letzten Zeilen (18, 19):

  _____

5.3 Gerätedateien

1 Punkte

Die Gerätedateien sind im Verzeichnis `/dev` enthalten. Stellen Sie fest, wieviele Einträge der folgenden Typen vorhanden sind (`find`, `wc`):

block device	regular file	link	socket
character device	pipe	directory	

Stellen Sie fest, mit welchen Rechten die erste Festplatte (`/dev/sda`) ausgestattet ist.

Rechte: _____

Setzen Sie Schreib-/Leserechte für alle Benutzer.

Wechseln Sie in ein Konsolenfenster oder auf eine alphanumerische Konsole (Strg + Alt + F1) und lesen Sie die Mausschnittstelle direkt aus:

```
cat  /dev/input/mouse0
```

Bewegen Sie jetzt die Maus (Abbruch mit Strg + c). Wenn die Konsole nur noch zufällige Zeichen anzeigt, dann tippen Sie blind `reset` ein.

Mit welchen Rechten ist die Mausschnittstelle ausgestattet ?

Rechte: _____

5.4 Das proc-Dateisystem **1 Punkte**

Das proc-Dateisystem ist eine Schnittstelle zum Betriebssystemkern, die als Dateisystem ausgebildet ist. Die Dateien können mit Hilfe eines Browsers (zum Beispiel `mc`) ausgelesen werden. Beschaffen Sie folgende Informationen (die Angaben gelten für die im Praktikum verwendete virtuelle Maschine):

Datei	gesucht	Ergebnis
`cpuinfo`	CPU-Modell Frequenz Cache	
`filesystems`	die ersten 5 unterstützten Dateisysteme, bei denen *nicht* `nodev` in der ersten Spalte steht	
`meminfo`	Größe des Hauptspeichers	
`modules`	die ersten 3 geladenen Kernel-Module	
`partitions`	Blockanzahl der `home`-Partition (die zugehörige Gerätedatei ermitteln Sie mittels `df`)	
`version`	Versionsnummer des BS-Kerns	

5.5 Festplatten-Partitionen als Dateien **3 Punkte**

Alle Geräte werden in *Unix*–artigen Betriebssystemen im Verzeichnisbaum unter `/dev` als Dateien virtualisiert. Tatsächlich kann der Inhalt von Partitionen als Image-Datei von einer Partition auf einem Datenträger (Festplatte, USB-Stick, und so weiter) kopiert, als Image-Datei ins Dateisystem eingehängt, sowie der Inhalt geändert und auf den Datenträger zurück kopiert werden.

5.5.1 Festplatten-Partition erstellen

Starten Sie das Partitionierungswerkzeug von `yast2` (→ System → Partitionierer).

1. Wählen Sie in der Baumdarstellung unter **Festplatten** die Platte `sdb` aus. Wählen Sie auf keinen Fall `sda` aus, sonst löschen Sie gleich Ihr vmware-Image!

2. Falls `sdb` Partitionen (`sdb1`, `sdb2`, und so weiter) enthalten sollte, löschen Sie diese. Wählen Sie die zu löschende Partition mit der Maus aus, klicken Sie auf den Button **Löschen** und bestätigen Sie die Sicherheitsabfrage.
3. Erstellen Sie eine neue Partition. Klicken Sie auf den Button **Hinzufügen**, wählen Sie **Primäre Partition** als neuen Partitionstyp und stellen Sie als neue Partitionsgröße 100 MB (nicht mehr!) unter **Benutzerdefinierte Größe** ein, Stellen Sie in den **Formatierungsoptionen** die Optionen **Partition formatieren** und unter **Dateisystem** den Typ **Ext3** und **Partition nicht einhängen** ein und klicken Sie auf **Beenden**.
4. Klicken Sie auf den Button **Weiter**, um die eingestellten Änderungen anzusehen.
5. Überprüfen Sie die geplanten Änderungen! Das ist die letzte Chance, eine nicht beabsichtigte Änderung von `sda` zu verhindern. Klicken Sie auf **Beenden**, wenn alles in Ordnung ist.
6. Sollten Sie ein Aufforderung zum Einhängen der neuen Partition erhalten, wählen Sie **Nichts unternehmen**.
7. Beenden Sie `yast2`.

Überprüfen Sie, ob die Partition `sdb1` vorhanden ist, indem Sie auf der Konsole den Befehl `cat /proc/partitions` oder `df` ausführen.

5.5.2 Festplatten-Partition verwenden

Im Folgenden werden Sie

- einige Dateien auf die gerade erstellte Festplatten-Partition schreiben,
- den Inhalt der Partition in eine Image-Datei schreiben,
- die Image-Datei wie eine Festplatten-Partition einhängen und den Inhalt ändern,
- den Inhalt der Festplatten-Partition mit dem Inhalt der Image-Datei überschreiben und
- überprüfen, dass der Inhalt der Festplatten-Partition und der der Image-Datei übereinstimmen.

5.5.3 Mountpoints erstellen

Um den Inhalt der Festplatten-Partition und der Image-Datei unterscheiden zu können, legen Sie zunächst für beide jeweils ein Verzeichnis als mount point an. Starten Sie eine Root–Konsole und erzeugen Sie mittels `mkdir` die Verzeichnisse `/mnt/image` und `/mnt/partition`, falls diese nicht vorhanden sind.

5.5.4 Dateien auf die Festplatten-Partition schreiben

Hängen Sie zunächst die neue Festplatten-Partition ein. Führen Sie dazu auf der Konsole aus:

```
mount /dev/sdb1 /mnt/partition
```

Überprüfen Sie, ob `/dev/sdb1` unter `/mnt/partition` eingehängt ist.

Kopieren Sie einige Dateien in das Verzeichnis `/mnt/partition`. Entsprechend der Größe der eingehängten Partitionen passen etwa 100 MB in das Verzeichnis.

Hängen Sie die Festplatten-Partition `/dev/sdb1` aus. Es darf keine Programme geben (zum Beispiel Browser, Konsole), die noch auf Dateien im Verzeichnis `/mnt/partition` zugreifen. Führen Sie zum Aushängen folgenden Befehl aus:

```
umount /mnt/partition
```

Überprüfen Sie, dass die Partition tatsächlich ausgehängt ist.

5.5.5 Inhalt der Partition /dev/sdb1 in eine Image-Datei kopieren

Kopieren Sie jetzt den Inhalt der Festplatten-Partition in eine Image-Datei. Führen Sie dazu folgenden Befehl aus:

```
dd if=/dev/sdb1 of=/tmp/partition_sdb1.img
```

5.5.6 Inhalt des Dateisystems in der Image-Datei ändern

Hängen Sie die nun die neue Image-Datei in das Dateisystem ein:

```
mount -t ext3 /tmp/partition_sdb1.img /mnt/image -o loop
```

Nehmen Sie nun einige Änderungen vor. Löschen Sie dazu eine Datei aus `/mnt/image` und kopieren Sie eine andere Datei in das Verzeichnis `/mnt/image`. (Sie ändern dadurch den Inhalt der Image-Datei `/tmp/partition_sdb1.img`, die unter `/mnt/image` eingehängt ist.)

Hängen Sie die Partition mittels

```
umount /mnt/image
```

anschließend aus.

5.5.7 Inhalt der Partition /dev/sdb1 mit dem Inhalt der Image-Datei überschreiben

Führen Sie genau diese Zeile aus und achten Sie auf `/dev/sdb1`:

```
dd if=/tmp/partition_sdb1.img of=/dev/sdb1
```

5.5.8 Inhalt der Festplatten-Partition /dev/sdb1 und die Image-Datei vergleichen

Hängen sie die Image-Datei und die Festplatten-Partition mittels

```
mount -t ext3 /tmp/partition_sdb1.img /mnt/image -o loop
mount -t ext3 /dev/sdb1 /mnt/partition
```

in den Dateibaum ein.

Vergleichen Sie den Inhalt von `/mnt/image` und `/mnt/partition` mittels `diff`

```
diff -r /mnt/image /mnt/partition
```

Die unter `/mnt/partition` eingehängte Festplatten-Partition muss jetzt den gleichen Inhalt haben wie die unter `/mnt/image` eingehängte Image-Datei, weil Sie die Partition mit dem Inhalt der Image-Datei überschrieben haben.

Wenn kein Unterschied angezeigt wird sind entweder die Dateibäume gleich oder Ihr Vergleichsbefehl ist fehlerhaft. Nehmen Sie deshalb in beiden Verzeichnissen ein bis zwei Änderungen vor und wiederholen Sie den Vergleich.

Hängen Sie die Festplatten-Partition und die Image-Datei anschließend wieder aus.

```
umount /mnt/partition
umount /mnt/image
```

5.6 Dateien/Verzeichnisse mit bestimmten Merkmalen suchen **3 Punkte**

Erarbeiten Sie die Lösungen für alle nachfolgenden Teilaufgaben zunächst in einem Shell-Fenster. Tragen Sie dann die Lösungen für 5.6.1 bis 5.6.5 in ein Skript ein, welches die entsprechenden Ergebnisse nacheinander in eine Berichtdatei `report.txt` schreibt. Die Abschnitte sind durch Zwischenüberschriften zu trennen, so dass eine einfache Navigation in einem Editor möglich ist.

5.6.1 Verzeichnisse der Größe nach auflisten

Erstellen Sie mit Hilfe des Befehls `du` eine nach fallender Größe sortierte Liste aller Verzeichnisse unterhalb des Wurzelverzeichnisses. Die Liste soll auf ein Dateisystem beschränkt sein (keine Suche über CDs, Sticks, `/proc`, …).

5.6.2 Dateien der Größe nach auflisten

Erstellen Sie mit Hilfe des Befehls `ls` eine nach fallender Größe sortierte Liste aller Dateien im Verzeichnis `/usr/bin`.

5.6.3 Dateien ohne Besitzer auf diesem Rechner

Erstellen Sie mit Hilfe des Befehls `find` eine sortierte Liste aller Dateien unterhalb der Wurzel `/` des Dateibaumes, deren Besitzer oder Gruppe auf diesem Rechner nicht existiert. Dabei ist der Abstieg in andere Dateisysteme (zum Beispiel CDs, `proc`, `dev` usw.) zu unterbinden.

5.6.4 Dateien mit Suid- oder Sgid-Bit

Erstellen Sie mit Hilfe des Befehls `find` eine sortierte Liste aller Dateien unterhalb der Wurzel `/` des Dateibaumes, bei denen das suid- oder sgid-Recht gesetzt ist (evtl. Sicherheitslücke). Dabei ist der Abstieg in andere Dateisysteme (zum Beispiel CDs, `proc`, `dev` usw.) zu unterbinden.

5.6.5 Dateien mit allgemeinem Schreibzugriff

Erstellen Sie mit Hilfe des Befehls `find` eine sortierte Liste aller Dateien unterhalb der Wurzel `/` des Dateibaumes, bei denen das Schreibrecht für `world` gesetzt ist (evtl. Sicherheitslücke). Dabei ist der Abstieg in andere Dateisysteme (zum Beispiel CDs, `proc`, `dev` usw.) zu unterbinden.

- Überprüfen Sie die Liste der gefundenen Dateien durch Stichproben.
- Wenn Sie keine Dateien mit den gesuchten Merkmalen finden, dann erzeugen Sie Testdateien mit diesem Merkmalen und prüfen nach, ob Sie diese Dateien finden.

6 Loop-back-Dateisysteme und Secure Shell

6.1 CD-Images via fuse-Dateisystem einhängen **6 Punkte**

Normalerweise sind die Treiber für die unterstützen Dateisysteme (ext3, fat, iso und so weiter) Kerneltreiber, für deren Verwendung Administrator-Rechte benötigt werden. Mit `fuse` (Filesystem in Userspace) steht eine Möglichkeit zur Verfügung, auch nicht-privilegierten Usern die Einbindung verschiedener Dateisysteme in ihrem Home-Verzeichnis zu erlauben. Mögliche Anwendungen sind die Bereitstellung mehrerer CDs/DVDs oder die zeitweise Einbindung verschlüsselter Dateisysteme. Für die von `fuse` unterstützten Dateisysteme stehen Pakete zur Verfügung, die einfach mit `yast2` installiert werden können. Einen Überblick bietet eine Suche im `yast2`-Modul *Software installieren oder löschen* nach dem Begriff `fuse` mit den Optionen *Name – Schlüsselwörter – Zusammenfassung – Beschreibung* unter *Suchen in*.

6.1.1 CD-Image einhängen

Sie werden in dieser Übung bemerken, dass alles unter *Unix*-ähnlichen Systemen eine Datei ist (oder besser: durch den Kernel für den Gebrauch transparent als Datei abgebildet wird) – selbst der Inhalt ganzer CDs, Festplatten-Partitionen und Festplatten. In dieser Übung soll das auf CDs eingesetzte Dateisystem `iso9660` verwendet werden. Angesichts der Kapazität aktueller Festplatten kann man auf diese Weise den Inhalt häufig gebrauchter CDs und DVDs ständig zur Verfügung stellen, ohne „Diskjockey“ spielen zu müssen.

Installieren Sie das Paket `fuseiso` über das Shell-Kommando

```
sudo zypper in fuseiso
```

Erstellen Sie als nicht privilegierter Benutzer `student` in Ihrem Home-Verzeichnis ein Unterverzeichnis `CDs` (`$HOME/CDs`). Kopieren Sie zwei CDs in Dateien mit Endung `.image` in diesem Verzeichnis. Dazu wird eine eingehängte CD[1] wie folgt eingelesen:

```
dd if=/dev/cdrom of=$HOME/CDs/1.image
```

Wenn Sie keine CD verfügbar haben und/oder der Zugriff auf ein Image mit der virtuellen Maschine nicht klappt, können Sie mit dem Befehl

```
sudo mkisofs -J -r -o $HOME/CDs/2.image /etc
```

selbst eine Image-Datei mit einem CD-ISO-Dateisystem erstellen, in dem eine Kopie des Ordners `/etc` (oder analog von anderen Verzeichnissen) landet.

[1] Im VMware Player können Sie über *Hardware-Einstellungen / CD/DVD / Festplatte oder Festplatten-Image wählen* einen Dateiauswahldialog öffnen und da eine ISO-Image-Datei auswählen und dann ein Häkchen vor *CD-/DVD-Laufwerk verbinden* setzen. (Alternativ nutzen Sie eine echte Daten-CD.) Als Beispiel-ISO-Image können Sie die nur 14 MByte große Datei `TinyCore-current.iso` unter `http://distro.ibiblio.org/tinycorelinux/5.x/x86/release/` herunterladen.

Schreiben Sie ein Shell-Skript `cd-mount.sh`, welches automatisch alle CD-Images in einem bestimmten Verzeichnis in den Dateibaum einbindet.

- Das Skript kann einen Verzeichnisnamen als optionalen Kommandozeilenparameter übernehmen. Wird kein Parameter übergeben, dann wird das Verzeichnis `$HOME/CDs` als Ersatzwert verwendet.
- Die Dateiendung `image` der CD-Images und der Name des Basisverzeichnisses werden in Variablen festgehalten. (Das erlaubt eine spätere Anpassung, z. B. an die Dateiendung `iso`.)
- In einer Schleife werden alle Dateinamen mit der Endung `image` eingelesen. Aus dem jeweiligen Namen wird die Endung `.image` entfernt (Kommandosubstitution) und ein gleichnamiges Unterverzeichnis erstellt, falls es nicht bereits vorhanden ist.
- Durch Untersuchung der Information in `/proc/mounts` kann nun festgestellt werden, ob das Verzeichnis bereits als Mountpoint verwendet wird. Ist das nicht der Fall, wird das CD-Image im zugehörigen Verzeichnis eingehängt:

  ```
  fuseiso "$image" "$mountdir"
  ```

- Zum Abschluss wird die Anzahl der erfolgreich eingehängten CD-Image in einer kurzen Meldung ausgegeben.

Die richtig eingehängten Images können mit dem Befehl

```
cat /proc/mounts
```

angezeigt werden. Die Inhalte der eingehängten CD-Images werden nun zum Beispiel von einem Dateimanager wie Unterverzeichnisbäume behandelt.

6.1.2 CD-Image aushängen

Schreiben Sie ein weiteres Skript `cd-umount.sh`, welches alle eingehängten CD-Images aus dem Dateibaum aushängt.

- Das Skript kann einen Verzeichnisnamen als optionalen Kommandozeilenparameter übernehmen. Wird kein Parameter übergeben, dann wird das Verzeichnis `$HOME/CDs` als Ersatzwert verwendet.
- Die Dateiendung `image` der CD-Images und der Name des Basisverzeichnisses werden in Variablen festgehalten.
- In einer Schleife werden alle Dateinamen mit der Endung `image` eingelesen. Aus dem jeweiligen Namen wird die Endung `.image` entfernt (Kommandosubstitution).
- Durch Untersuchung der Information in `/proc/mounts` kann nun festgestellt werden, ob das Verzeichnis gerade als mount point für ein `fuseiso`-Dateisystem verwendet wird. Ist das der Fall, wird das CD-Image ausgehängt:

  ```
  fusermount -u "$mountdir"
  ```

- Anschließend werden alle leeren Unterverzeichnisse unterhalb des Basisverzeichnisses gelöscht.
- Zum Abschluss wird die Anzahl der erfolgreich ausgehängten CD-Images in einer kurzen Meldung ausgegeben.

6.2 Die Verwendung der Secure Shell (ssh) **4 Punkte**

6.2.1 Einrichtung und Anmeldung auf einem Testrechner

Melden Sie sich per SSH auf einem entfernten Linux-Rechner an. Die IP-Adresse wird Ihnen zu Beginn des Praktikums bekannt gegeben. (Sie können sich über den Befehl

```
ssh root@IP-ADRESSE
```

mit dem bekannten `root`-Passwort auf dem Rechner einloggen.)

Wechseln Sie in das Verzeichnis `/home/student/SSH`. Dort erzeugen Sie einen neuen Ordner und benennen ihn nach Ihrer Benutzerkennung.

Erzeugen Sie darin die Datei `testdatei` mit folgendem Befehl:

```
dd if=/dev/zero of=/home/student/SSH/BENUTZERKENNUNG/testdatei bs=1024 count=10
```

(Informieren Sie sich vorher über die Gerätedatei `/dev/zero` und über die Parameter von `dd`.) Überprüfen Sie anschließend die Existenz und Größe dieser Datei.

Legen Sie ein Unterverzeichnis `subdir1` im Ordner `/home/student/SSH/BENUTZERKENNUNG/` an und kopieren Sie die Datei `testdatei` unter drei verschiedenen Namen hinein. Melden sie sich mit `exit` wieder ab.

6.2.2 Secure Copy (scp): verschlüsselter Dateitransfer

Kopieren Sie zunächst die Datei `testdatei` vom entfernten Rechner in Ihr eigenes Arbeitsverzeichnis:

```
scp  root@IP-ADRESSE:/home/student/SSH/BENUTZERKENNUNG/testdatei  testdatei
```

Kopieren Sie dann das Verzeichnis `/home/student/SSH/BENUTZERKENNUNG/subdir1` des entfernten Rechners in Ihr Arbeitsverzeichnis und kontrollieren Sie das Ergebnis:

```
scp  -r  root@IP-ADRESSE:/home/student/SSH/BENUTZERKENNUNG/subdir1  ~
```

6.2.3 Grafische Anmeldung auf dem entfernten Rechner

Melden Sie sich grafisch auf dem entfernten Rechner an, zum Beispiel mit

```
ssh -X  student@IP-ADRESSE
```

Starten Sie zur Überprüfung des Erfolgs eine grafische Anwendung, zum Beispiel den Dateimanager `dolphin`, und melden Sie sich dann wieder ab.

Tragen Sie nun eine Zeile der folgenden Form in die Datei `~/.alias` ein:

```
alias IP-ADRESSE='ssh -X  student@IP-ADRESSE'
```

und laden Sie die `alias`-Datei in die laufende Shell:

```
source ~/.alias
```

Jetzt steht der Shell-Befehl `IP-ADRESSE` zur Verfügung. Überprüfen Sie, ob durch Eingaben dieses Kürzels eine grafische Anmeldung auf dem entfernten Rechner möglich ist.

7 Verschlüsselte Dateisysteme und cron-Jobs

7.1 Einrichtung und Betrieb **7 Punkte**

Ziel: Einrichtung eines verschlüsselten Dateisystems. Das Dateisystem soll durch ein sicheres Verfahren verschlüsselt werden.

7.1.1 Paket für verschlüsselte Dateisysteme installieren

Installieren Sie als Benutzer `root` mit `yast2` oder auf der Kommandozeile mit `zypper install` das Paket `encfs`.

7.1.2 Einrichtung eines verschlüsselten Dateisystems

Machen Sie sich mit der Dokumentation von `encfs` (man-Pages) vertraut; lesen Sie insbesondere auch den Abschnitt `EXAMPLES`. *Hinweis:* Die Einstellungen des Programms werden nicht über Optionen gesteuert, sondern das Programm bietet einen Expertenmodus, in dem interaktiv die Konfiguration erfragt.

Erzeugen Sie ein verschlüsseltes Dateisystem im Verzeichnis `$HOME/crypto`, dessen verschlüsselter Inhalt im Verzeichnis `$HOME/.crypto` gespeichert wird.

Verwenden Sie für die Verschlüsselung den Algorithmus **Blowfish**, die Schlüssellänge **256 Bit** und den Verschlüsselungsalgorithmus **Block**. Für alle anderen Einstellungen übernehmen Sie die Vorgaben, gegen Ende geben Sie zweimal ein Passwort für die Verschlüsselung ein. Das verschlüsselte Dateisystem ist nach dem Anlegen bereits eingehängt.

Hängen Sie das verschlüsselte Verzeichnis wieder aus:

```
fusermount -u $HOME/crypto
```

7.1.3 Verschlüsseltes Dateisystem benutzen

Hängen Sie das Dateisystem ein und wieder aus:

```
encfs  ~/.crypto ~/crypto

fusermount -u ~/crypto
```

Überprüfen Sie die Wirksamkeit der Zugriffsrechte: Hängen Sie das Dateisystem ein. Melden Sie sich als weiterer nicht privilegierter Benutzer an (nicht der Besitzer `student`) und prüfen Sie nach, ob der Wechsel in das verschlüsselte Verzeichnis möglich ist. (Dazu müssen Sie zunächst mit `useradd` einen neuen Account anlegen.) Melden Sie sich abschließend als `root` an und versuchen Sie das Gleiche.

Versuchen Sie, als `root` die Rechte so zu ändern, dass Sie ebenfalls Zugang zum Dateisystem haben.

7.1.4 Vereinfachung der Benutzung

Schreiben Sie zwei Bash-Skripte, die das Ein- und Aushängen des verschlüsselten Dateisystems erleichtern.

Das Skript `mcrypt.sh` prüft in der Datei `/proc/mounts` (mittels `grep`, `cut`), ob das verschlüsselte Verzeichnis nicht bereits als gemountet eingetragen ist.

- Falls das Verzeichnis nicht eingetragen ist, wird es eingehängt.
- Falls das Verzeichnis bereits eingetragen war, wird eine Meldung ausgegeben, dass das Verzeichnis bereits eingehängt wurde.

Diese Vorgehensweise verhindert, dass das Dateisystem mehrfach eingehängt werden kann.

Das Skript `umcrypt.sh` prüft in der Datei `/proc/mounts`, ob das verschlüsselte Verzeichnis als gemountet eingetragen ist. Falls ja, wird es ausgehängt, andernfalls wird eine Meldung ausgegeben, dass das Verzeichnis nicht eingehängt ist.

Wenn beide Skripte einwandfrei arbeiten, machen Sie aus den beiden Skripten jeweils eine Bash-Funktion (`crypt+` zum Einhängen, `crypt-` zum Aushängen) und fügen beide am Ende der Datei `~/.bashrc` an. Diese Datei enthält Einstellungen, die beim Start einer neuen Shell geladen werden.

Funktionen erstellen Sie über die Syntax

```
function name () {
  befehl1
  befehl2
  ...
}
```

und können innerhalb einer Funktion mit `$1`, `$2` usw. auf Funktionsargumente zugreifen.

Über den Befehl

```
source .bashrc
```

kann die Datei jedoch auch in der laufenden Sitzung eingelesen werden. Die beiden Funktionen können dann wie Shell-Befehle aufgerufen werden.

7.1.5 Automatisches Aushängen eines ungenutzten Dateisystems

Das verschlüsselte Dateisystem bleibt solange zugreifbar, wie es gemountet ist. Wird das Aushängen vergessen, bleibt das Dateisystem auch nach der Abmeldung des Benutzers im Dateibaum eingehängt (!) und ist nur durch seine Zugriffsrechte geschützt. Deshalb soll die folgende Gegenmaßnahme eingerichtet werden:

Shell-Logout

Stellen Sie fest, ob die Datei `.bash_logout` im Home-Verzeichnis des Benutzers vorhanden ist – falls nicht, richten Sie eine solche Datei ein. Nun tragen Sie darin den neuen Befehl `crypt-` ein. Mögliche Ausgaben dieses Befehls werden nach `/dev/null` umgeleitet.

Überprüfen Sie die Wirkung:

- Auf der Konsole 1 anmelden (Benutzer `student`) und das Dateisystem einhängen.
- Auf der Konsole 2 anmelden (Benutzer `student`) und mittels `df` die sichtbaren Dateisysteme anzeigen.

- Auf der Konsole 1 abmelden.
- Auf der Konsole 2 mittels `df` die sichtbaren Dateisysteme anzeigen und ebenfalls abmelden.

Achtung: Das Verzeichnis wird bei der nächsten Abmeldung ausgehängt, auch wenn bei Mehrfachanmeldung das Einhängen durch eine andere Anmeldung durchgeführt wurde!

7.2 Automatisierung wiederkehrender Aufgaben: cron-Jobs **3 Punkte**

7.2.1 Testskript für eine stündlich auszuführende Maßnahme

Schreiben Sie als Administrator `root` ein Shell-Skript `test_hourly.sh`, in welchem das aktuelle Datum und die augenblickliche Zeit festgestellt werden (Befehl `date`). Diese Angaben werden bei jedem Aufruf des Skriptes mit einer `echo`-Anweisung an eine eigene Logdatei angehängt (zum Beispiel `/root/test.log`). Testen Sie das Skript zunächst durch einige Aufrufe von der Kommandozeile und löschen Sie die Logdatei anschließend.

Kopieren Sie das Skript nun in das Verzeichnis `/etc/cron.hourly`. Nach spätestens einer Viertelstunde sollte der erste Eintrag in der Logdatei erscheinen, und die Datei `/var/spool/cron/lastrun.cron.hourly` sollte denselben Zeitstempel besitzen. Die Logdatei kann in einem Shell-Fenster automatisch beobachtet werden, wenn sie vorher als leere Datei angelegt wurde:

```
tail -f /root/test.log
```

Stellen Sie dazu die Bedeutung des Schalters `-f` des Befehl `tail` fest.

7.2.2 Überwachung des verschlüsselten Dateisystems durch einen cron-Job

Zur Überwachung des verschlüsselten Dateisystems soll ein Skript regelmäßig feststellen, ob das Dateisystem eine vorgegebene Zeitspanne lang nicht verwendet wurde. Ist das der Fall, dann wird das Dateisystem selbsttätig ausgehängt. Das dient als weitere Schutzmaßnahme, ähnlich einem Bildschirmschoner, der nach einer gewissen Zeit die Benutzeroberfläche sperrt. Erzeugen Sie dazu im `bin`-Unterverzeichnis des Benutzers (`$HOME/bin`) ein Skript `unmount_crypto.sh` mit zwei Kommandozeilenparameter:

1. Absoluter Pfadname des verschlüsselten Verzeichnisses (mount point)
2. Wartezeit (in Sekunden)

Das Skript führt folgende Schritte aus:

- Das Skript wird nur ausgeführt, wenn der erste Parameter eine Verzeichnisangabe ist.
- In der Datei `/proc/mounts` wird nachgesehen, ob das Verzeichnis gemountet ist. Wenn ja, wird über den Befehl `stat` der Zeitstempel des letzten Zugriffs (access time) in Sekunden seit Beginn der *Unix*-Zeitrechnung (epoch) ermittelt. Mit Hilfe des Befehls `date` wird die aktuelle Zeit ebenfalls in Sekunden seit Beginn der *Unix*-Zeitrechnung ermittelt. Wenn der Unterschied zwischen aktueller Zeit und letztem Zugriff größer ist als die Wartezeit (2. Parameter, siehe oben), dann wurde das Dateisystem mindestens für diese zurückliegende Zeitspanne nicht benutzt und wird deshalb ausgehängt. War das Verzeichnis nicht eingehängt oder war die Wartezeit noch nicht abgelaufen, wird das Skript sofort beendet.

Testen Sie das Skript zunächst auf der Kommandozeile. Richten Sie dann als nicht-privilegierter Benutzer einen cron-Job ein, der alle fünf Minuten aufgerufen wird. Das Skript soll mit einer Wartezeit von einer Minute (= 60 Sekunden) gestartet werden. (Das ist zwar für den normalen Gebrauch etwas kurz – aber für die Überprüfung der Lösung sinnvoll.) Überprüfen Sie die richtige Arbeitsweise, indem Sie das geöffnete Verzeichnis eine entsprechende Zeit lang nicht verwenden.

Ein cron-Job (für nicht-privilegierte Anwender) wird mit der Anwendung `crontab` eingerichtet. Über das zu verwendende Format zur Beschreibung eines cron-Jobs klärt `man 5 crontab` auf. Das crontab-Tool startet einen Editor. In der Regel ist das der Editor `vi` oder `vim`. Wer lieber den grafischen Editor `kwrite` verwendet, muss die Zeile

```
export EDITOR=kwrite
```

in die Datei `~/.bashrc` eintragen und diese Datei neu einlesen (`source ~/.bashrc`) oder eine neue Shell öffnen; alternativ können Sie auch (für einmaligen Gebrauch) dem crontab-Aufruf die Zuweisung `EDITOR=kwrite` voranstellen.

Literaturverzeichnis

[FSF10] FSF. Bash Reference Manual. http://www.gnu.org/software/bash/manual/, Free Software Foundation, 12 2010. Bash shell, version 4.2. The official manual.

[Meh14] Fritz Mehner. Bash Style Guide und Kodierungsrichtlinie. https://lug.fh-swf.de/vim/vim-bash/StyleGuideShell.de.pdf, 2014.