Technische Hochschule Nürnberg
GEORG−SIMON−OHM

Name: .....

Schriftliche Prüfung im Fach

Vorname: .....

"Betriebssystem UNIX/Linux"

Matrikel−Nr.: .....

Prüfungstermin: 14.03.2014

Semester: .....

Prüfungsdauer: 90 Minuten

Note: .....

Punktzahl: .....

Anzahl der Aufgabenblätter: 6
Maximal erreichbare Punktzahl: 100 Punkte + 10 Zusatzpunkte
Zugelassene Hilfsmittel: alles außer Rechner

Aufgabensteller: Stockmayer, Lehrbeauftragter an der Fakultät efi
Prüfer: Stockmayer

Aufgabe 1: Welche Rechte werden bei folgendem Befehl gesetzt:

```
chmod 754 file.txt
chmod 750 dir.d
```

Was bewirken diese Rechte für wen? Für welche Kommandos sind diese Rechte von Bedeutung?
(file.txt ist ein (Text)−File, dir.d ist ein Verzeichnis)

(14 Punkte)

```
chmod 754 file.txt
Besitzer darf alles (lesen, schreiben, ausführen), Gruppen−
mitglieder lesen und ausführen, Rest der Welt nur lesen
lesen: z.B. mit cat file1.txt, less file1.txt
schreiben: z.B. mit echo text >> file1.txt oder mit
        Editor (z.B. vi)
ausführen: durch nennen des Namens, wenn in PATH:
        file.txt (z.B. wenn dort Kommandos drinstehen,
        Skript)

chmod 750 dir.d
Besitzer darf alles (listen, verändern, reingehen), Gruppen−
mitglieder listen und reingehen, Rest der Welt nix
listen: z.B. mit ls dir.d
verändern: z.B. mit mv x1 dir.d/y1 oder kopieren (cp) oder
        löschen (rm)
reingehen: z.B. mit cd dir.txt
```

Kommandos: ls, cd, cat/less/more, vi, cp/mv/ln, rm, ...

Aufgabe 2: Erklären Sie folgende Begriffe (geben Sie, falls möglich, ein Beispiel): (25 Punkte)

Indirektionsblock (was ist das?, wo kommt es vor?):
Im Dateisystem auf der Partition im Datenbereich ein Block mit Adressen auf die Daten oder weiteren Indirektions−blöcken.
Er selber wird im i−Node adressiert oder in einem anderen Indirektionsblock. Ziel: möglichst große Dateien anzusprechen.

Prozess (was ist das?, wie kann er gestartet werden?, Beispiel?):
laufendes Kommando im Hauptspeicher
Befehle in einer Datei laufen lassen (binär oder Skript)
Prozesstabelle im Kern mit PID, PPID
Start durch Nennen des Namens (wenn PATH Verzeichnis enthält)
Hintergrund: &
ps/top/pstree: Listen (Status: SWZTR)
^c, kill: weg (kill −9: absolut weg)
^z, fg, bg, jobs: erweiterte Prozesskontrolle
env: Umgebungsvariable

Trojaner (was ist das?, wie kann er vermieden werden?):
Programm, das vorgibt, ein anderes zu sein und Schaden anrichtet. Z.B. das Passwort der root ausspähen.
Prüfen von wo starten: type
oder Starten mit Pfad: /bin/su
prüfen der PATH−Einstellung: echo $PATH

Pipen (was ist das?, wozu wird das benötigt?, Beispiel):
stdout eines Kommandos weitergeben in die stdin eines 2. Kommandos; Zwischenspeicher (4−64k); Prozeßsynchronisation; es können Zombies entstehen; Zeichen: |
ps −e | grep top | less

Umgebung (Environment) (was ist das?, wozu wird das benötigt?, Beispiel):
Einstellungen des Parentprozesses, die sich auf den Childprozess auswirken:
aktueller Pfad
Environmentvariable: export var=inhalt; env

Aufgabe 3: Erläutern Sie __detailiert__ folgenden Kommandoaufruf (18 Punkte):

```
sort −r ?[a−f]*f |

tee −a /tmp/erg |

cat −n |

grep ’abcd ’ |

lp &
```

sortiere alle Zeilen (lexikografisch) der Dateien aus dem aktuellen Verzeichnis, die mit einem bel. Zeichen beginnen, dann ein a bis f haben und mit f aufhören (dazwischen beliebiges).
Das Resultat sende über eine Pipe (stdout −> stdin) an ein Tee−Stück, das die (sortierten) Zeilen an eine Datei erg im Verzeichnis /tmp speichert (anhängt) und via Pipe weiterreicht an cat,
der Zeilennummern hinzufügt. Die Ausgabe wird gefiltert: nur Zeilen, die den Text abcd gefolgt von einem Blank enthalten werden weitergereicht an den Druckspooler,
der die (nummerierten, sortierten) Zeilen
dann am Defaultdrucker ausgibt.
ALLES läuft im Hintergrund.

Aufgabe 4: Schreiben Sie ein Shell−Skript, das alle 10 Sekunden (nur) die Uhrzeit mit Text ausgibt ("es ist 09:10 Uhr") und bringen Sie es unter dem Namen "uhr.sh" zum Laufen.
Was müssen Sie tun, dass es jeder aus beliebigen Verzeichnissen starten kann?
(15 Punkte)

```
#!/bin/sh                                          # 1 Punkt
# Uhrzeit in Endlosschleife                        # 1
while true                                         # 2
do
     date ’es ist +%T Uhr’   # Format für Uhrzeit    # 2
     sleep 10                                      # 1
done                                               # 1
# exit ist nicht notwendig, da Endlos

chmod a+rx uhr                                     # 3
PATH=$PATH:$HOME     # oder wo auch immer das Skript liegt   # 2
```

Wenn es alle starten können sollen, muss auch das Verzeichnis, in dem das Skript liegt, von allen gelesen und ausführbar sein:

```
chmod a+rx $HOME     # 2
```

Diese müssen ebenfalls die PATH−Variable auf dieses Verzeichnis setzen.

Aufgabe 5: Welche Aufgaben und Möglichkeiten hat eine Shell besonders für die interaktive Kommandozeile (geben Sie Beispiele)?
(17 Punkte)

Kommandohistorie: history, Cursortasten # 2
TAB−Taste: Kommandos, Dateinamen, ... ergänzen # 2
Alias: einfache Namen für Kommandos: alias c=clear # 2
Metazeichen: viele ähnliche Dateinamen gemeinsam ansprechen:
    * ? [] {} ~ # 3
Umlenken: Ergebnisse speichern: ls > erg # 2
Pipen: Ergebnisse weiterreichen an andere Kommandos:
    ls | less # 2
Variable PS1: Im Prompt Infos wie akt. Pfad anzeigen:
    PS1='$PWD −−> ' # 2
Kommandos im Hintergrund laufen lassen: find ... & # 2
Kommandos stoppen und weiter: ^z, bg, fg

auch:
Schützen: echo \$
Startupdateien konfigurieren
unterschiedliche Shells mit unterschiedlichen interaktiven Eigenschaften
Dateien anlegen: > datei
Funktionen (nicht besprochen)

Aufgabe 6: Wie können Sie eine Datei untersuchen nach Typ und genauem Inhalt bzw. was sie mit ihr tun können?
Geben Sie die Kommandos an, die für die Untersuchung in einem Shell−Skript stehen müssten (d.h. kein komplettes Shell−Skript).
(11 Punkte)

1) mit ls −l den Typ anschauen (1. Zeichen: d,−,c,b,...)
   ls −l datei | cut −c1
2) Mit dem 'file'−Kommando kann der Inhalt genau spezifiert werden (text, executable, data, ...)
   file datei
3) mit Hilfe von grep und if kann dies in einem Skript automatisiert werden
   if file datei | grep 'text'
4) durch anschauen der Rechte: x für ausführbar
5) durch testen der Rechte: 'test −x datei'
6) dieses Ergebnis kann dann in einem Shell−Skript mit der if−Verzweigung weiterverwendet werden
   if test −x datei
   then
        ...
7) Auswerten des exit−Codes: $?
8) type (wenn es ein Kommando ist)
9) Ort: find