# Lösung von Übungsblatt 2

## Aufgabe 1 (Klassifikationen von Betriebssystemen)

1. Zu jedem Zeitpunkt kann nur ein einziges Programm laufen. Nennen Sie den passenden Fachbegriff für diese Betriebsart.

   *Einzelprogrammbetrieb (Singletasking).*

2. Was versteht man unter halben Multi-User-Betriebssystemen?

   *Verschiedene Benutzer können nur nacheinander am System arbeiten, aber die Daten und Prozesse der Benutzer sind voreinander geschützt.*

3. Was ist der Unterschied zwischen 8 Bit-, 16 Bit-, 32 Bit- und 64 Bit-Betriebssystemen?

   *Die Bit-Zahl gibt die Länge der Speicheradressen an, mit denen das Betriebssystem intern arbeitet.*

4. Was sind die wesentlichen Kriterien von Echtzeitbetriebssystemen?

   *Reaktionszeit (geringe Latenzzeit) und das Einhalten von Deadlines.*

5. Es gibt zwei Arten von Echtzeitbetriebssystemen. Geben Sie deren Namen an.

   *Harte Echtzeitsysteme und weiche Echtzeitsysteme.*

6. Nennen Sie vier typische Einsatzgebiete von Echtzeitbetriebssystemen und ordnen Sie jedes Einsatzgebiet einer der Kategorien aus Teilaufgabe 5 zu.

   *Typische Einsatzgebiete harter Echtzeitbetriebssysteme: Schweißroboter, Reaktorsteuerung, ABS, Flugzeugsteuerung, Überwachungssysteme auf der Intensivstation,...*

   *Typische Einsatzgebiete weicher Echtzeitbetriebssysteme: Telefonanlage, Parkschein- oder Fahrkartenautomat, Multimedia-Anwendungen wie Audio/Video on Demand,...*

7. Beschreiben Sie den Aufbau eines Thin-Kernels.

   *Der Betriebssystemkern selbst läuft als Prozess mit niedrigster Priorität. Der Echtzeit-Kernel übernimmt das Scheduling. Echtzeit-Prozesse haben die höchste Priorität $\Longrightarrow$ minimale Latenzzeiten.*

8. Beschreiben Sie den Aufbau eines Nano-Kernels.

*Neben dem Echtzeit-Kernel kann eine beliebige Anzahl anderer Betriebssystem-Kernel laufen.*

9. Beschreiben Sie den Aufbau eines monolithischen Kernels.

   *Monolithische Kerne enthalten...*

   - *Funktionen für Speicherverwaltung*
   - *Funktionen für Prozessverwaltung*
   - *Funktionen für Prozesskommunikation*
   - *Gerätetreiber*
   - *Dateisysteme-Treiber*

   *Außerhalb des Kerns befinden sich die Benutzerprozesse.*

10. Beschreiben Sie den Aufbau eines minimalen Kerns (Mikrokernels).

    *Im Kern befinden sich üblicherweise nur:*

    - *Notwendigste Funktionen zur Speicher- und Prozessverwaltung*
    - *Funktionen zur Synchronisation und Interprozesskommunikation*
    - *Notwendigste Treiber (z.B. zum Systemstart)*

    *Gerätetreiber, Dateisysteme und Dienste (Server), befinden sich außerhalb des Kerns und laufen wie die Anwendungsprogramme im Benutzermodus*

11. Beschreiben Sie den Aufbau eines hybriden Kernels.

    *Hybride Kerne sind ein Kompromiss zwischen monolithischen Kernen und minimalen Kernen. Die enthalten aus Geschwindigkeitsgründen Komponenten, die bei minimalen Kernen außerhalb des Kerns liegen. Es ist nicht festgelegt, welche Komponenten bei Systemen mit hybriden Kernen zusätzlich in den Kernel einkompiliert sind.*

12. GNU HURD verwendet einen...

    ☐ monolithischen Kern
    ☒ minimalen Kern (Mikrokernel)
    ☐ hybriden Kern

13. Linux verwendet einen...

    ☒ monolithischen Kern
    ☐ minimalen Kern (Mikrokernel)
    ☐ hybriden Kern

14. MacOS X verwendet einen...

    ☐ monolithischen Kern
    ☐ minimalen Kern (Mikrokernel)
    ☒ hybriden Kern

15. Windows NT4/Vista/XP/7/8/10 verwendet einen...

    ☐ monolithischen Kern
    ☐ minimalen Kern (Mikrokernel)
    ☒ hybriden Kern

16. Nennen Sie einen Vorteil und einen Nachteil von monolithischen Kernen.

    - *Vorteile:*
      - *Weniger Kontextwechsel als Mikrokernel $\Longrightarrow$ höhere Geschwindigkeit*
      - *Gewachsene Stabilität*
    - *Nachteile:*
      - *Abgestürzte Komponenten können im Kernel nicht separat neu gestartet werden und das gesamte System nach sich ziehen*
      - *Hoher Entwicklungsaufwand für Erweiterungen am Kern, da dieser bei jedem Kompilieren komplett neu übersetzt werden muss*

17. Nennen Sie einen Vorteil und einen Nachteil von minimalen Kernen (Mikrokerneln).

    - *Vorteile:*
      - *Einfache Austauschbarkeit der Komponenten*
      - *Beste Stabilität und Sicherheit (in der Theorie!), weil weniger Funktionen im Kernelmodus laufen*
    - *Nachteile:*
      - *Langsamer wegen der größeren Zahl von Kontextwechseln*
      - *Entwicklung eines neuen (Mikro-)kernels ist eine komplexe Aufgabe*

18. Nennen Sie einen Vorteil und einen Nachteil von hybriden Kernen.

    - *Vorteile:*
      - *Höhere Geschwindigkeit als minimale Kerne (da weniger Kontextwechsel)*
      - *Bessere Stabilität (in der Theorie!) als monolithische Kerne*
    - *Nachteile:*
      - *Entwicklung eines neuen (Hybrid-)kernels ist eine komplexe Aufgabe*

19. Ein Kollege empfiehlt Ihnen häufig verwendete Server-Dienste wie z.B. Web-Server, Email-Server, SSH-Server und FTP-Server vom Benutzermodus in den Kernelmodus zu verlagern. Wie stehen Sie zu dieser Idee? Begründen Sie Ihre Antwort. Nennen Sie hierfür einen Vorteil und einen Nachteil.

    *Von Vorteil wäre, das das Betriebssystem und die Server-Dienste insgesamt schneller arbeiten, weil im beschriebenen Szenario weniger Moduswechsel zwischen Benutzermodus und Kernelmodus nötig sind.*

    *Gravierender ist aber der entstehende Nachteil. Es liegt ein Sicherheitsrisiko vor. Komplexe Software wie Server-Dienste sollten nicht im Kernelmodus lau-*

*fen. Softwarefehler in den Server-Diensten könnten zu Systemabstürzen oder zur vollständigen Kontrollübernahme durch Angreifer führen.*

20. Was versteht man unter Single System Image?

    *Die Benutzer und deren Anwendungen wissen nicht, dass die von ihnen in Anspruch genommenen Dienste auf mehreren Rechnern laufen.*

# Aufgabe 2 (Grundlegende Linux/UNIX-Kommandos)

Mit welchem Kommando können Sie...

1. Handbuchseiten („Man Pages“) öffnen

   *man*

2. das aktuelle Verzeichnis in der Shell ausgeben?

   *pwd*

3. ein neues Verzeichnis erzeugen?

   *mkdir*

4. in ein Verzeichnis wechseln?

   *cd*

5. den Inhalt eines Verzeichnisses in der Shell ausgeben?

   *ls*

6. eine leere Datei erzeugen?

   *touch*

7. versuchen den Inhalt einer Datei zu bestimmen?

   *file*

8. den Inhalt verschiedener Dateien verknüpfen oder den Inhalt einer Datei ausgeben?

   *cat*

9. Zeilen vom Ende einer Datei in der Shell ausgeben?

   *tail*

10. Zeilen vom Anfang einer Datei in der Shell ausgeben?

    *head*

11. Dateien oder Verzeichnisse an eine andere Stelle kopieren?

    *cp*

12. Dateien oder Verzeichnisse an eine andere Stelle verschieben?

    *mv*

13. Dateien oder Verzeichnisse löschen?

    *rm*

14. ein leeres Verzeichnis löschen?

    *rmdir*

15. eine Zeichenkette in der Shell ausgeben?

    *echo*

16. die Dateirechte von Dateien oder Verzeichnissen ändern?

    *chmod*

17. Das Password eines Benutzers ändern?

    *passwd*

18. die laufende Sitzung (und damit auch die Shell) zu beenden und den Rückgabewert eines Shell-Skripts festzulegen?

    *exit*

19. das System neu starten?

    *reboot oder alternativ shutdown*

20. das System ausschalten?

    *halt oder alternativ shutdown*

21. einen neuen Benutzer erstellen?

    *adduser*

22. einen Benutzer löschen?

    *deluser*

23. einen Benutzer ändern?

*usermod*

24. die Gruppenzugehörigkeiten des Benutzers ausgeben?

    *groups*

25. eine neue Gruppe erstellen?

    *groupadd*

26. eine Gruppe löschen?

    *groupdel*

27. eine Gruppe ändern?

    *groupmod*

28. den Benutzer ($\Longrightarrow$ Besitzer) ändern, der einer Datei oder einem Verzeichnis zugeordnet ist

    *chown*

29. die Gruppe ändern, die einer Datei oder einem Verzeichnis zugeordnet ist

    *chgrp*

30. einen „Link“ erstellen?

    *ln*

31. eine Datei nach den Zeilen durchsuchen, die ein Suchmuster enthalten?

    *grep*

32. eine Liste der laufenden Prozesse in der Shell ausgeben?

    *ps*

33. einen im Hintergrund der Shell laufenden Prozess in den Vordergrund holen?

    *fg*

34. einen Prozess in den Hintergrund der Shell verschieben?

    *bg*

35. einen Prozess beenden?

    *kill*

36. eine Gruppe von Prozessen beenden?

    *killall*

37. die Priorität eines neuen Prozesses festlegen?

    *nice*

38. die Priorität eines existierenden Prozesses ändern?

    *renice*

39. eine Liste der existierenden Prozesse als Baumstruktur in der Shell ausgeben?

    *pstree*

# Aufgabe 3 (Permissions / Access Rights)

The source of this tutorial is:
`http://www.ws.afnog.org/afnog2012/unix-intro/presos/permissions-exercises.pdf`

**Notes**

- Commands preceded with `$` imply that you should execute the command as a general user an not as root.
- Commands preceded with `#` imply that you should working as root with `sudo` imply that you are executing commands on remote equipment, or within another program.

**REFERENCE**

If you look at files in a directory using `ls -al` you will see the permissions for each file and directory. Here is an example:

```
drwxrwxr-x    3 bnc   bnc      4096 Feb 25 09:49 directory
-rwxr--r--   12 bnc   bnc      4096 Feb 16 05:02 file
```

The left column is important. You can view it like this:

```
Type User Group Other Links Owner Group  Size  Date   Hour  Name
=====================================================================
d    rwx  rwx   r-x   3     bnc   bnc    4096  Feb 25 09:49 directory
-    rwx  r     r     12    bnc   bnc    4096  Feb 16 05:02 file
```

The directory has `r` (read), `w` (write), `x` (execute) access for the User (= Owner) and Group. For Other it has `r` (read) and `x` (execute) access.

The file has `r` (read), `w` (write), `x` (execute) access for User and `r` (read) only access for everyone else (Group and Other).

You can change permissions with the `chmod` command. `chmod` uses a base eight (octal) system to configure permissions. Or, you can use an alternate form to specify permissions by column (User/Group/Other) at a time.

Permissions have values like this:

```
Letter   Permission   Value
============================
r        read         4
w        write        2
x        execute      1
-        none         0
```

Thus you can give permissions to a file using the sum of the values for each permission you wish to give for each column. Here is an example:

```
Letter   Permission                  Value
=======================================
---      none                        0
--x      execute                     1
-w-      write only (rarely used)    2
-wx      write and execute (rare)    3
r--      read only                   4
r-x      read and execute            5
rw-      read and write              6
rwx      read, write, and execute    7
```

This is just one column. Since we have three areas of permissions (User, Group, Other), it looks like this, if you want to specify all 3 sets:

```
Permissions  Numeric     Description
             equivalent

-rw-------   600         User has read & write permission.
-rw-r--r--   644         User has read & write permission.
                         Group and Other have read permission.
-rw-rw-rw-   666         Everyone (User, Group, Other) has
                         read & write permission (dangerous?)
-rwx------   700         User has read, write, execute permission.
-rwxr-xr-x   755         User has read, write, execute permission.
                         Rest of the world (Other) has read & execute
                         permission (typical for web pages or 644).
-rwxrwxrwx   777         Everyone has full access (read, write,
                         execute).
```

| | | |
|---|---|---|
| `-rwx--x--x` | `711` | `User has read, write, execute permission. Group and world have execute permission.` |
| `drwx------` | `700` | `User only has access to this directory. Directories require execute permission to access.` |
| `drwxr-xr-x` | `755` | `User has full access to directory. Everyone else can see the directory.` |
| `drwx--x--x` | `711` | `Everyone can list files in the directory, but Group and Other need to know a filename to do this.` |

### 1.) CHANGING FILE PERMISSIONS

If you are logged in as the root user on your machine please do the following to become a normal user.

```
# exit
```

Your prompt should change and now include a `$` sign.

```
$
```

Please check your username with the command `whoami`:

```
$ whoami
```

Please create a file and set permissions of the file in various ways.

```
$ cd
$ echo "test file" > working.txt
$ chmod 444 working.txt
```

What does that look like?

```
$ ls -lah working.txt
```

Because the file has no write permission for the owner, the owner can still change the file's permissions. This way, the owner can change the permissions of the file at any time to have write access again.

```
$ chmod 644 working.txt
```

Or, you can do this by using this form of `chmod`:

```
$ chmod u+w working.txt
```

Note: When you type these commands you should be able to use the tab key for command completion once you've typed the `w` in the file name `working.txt`. This will save you a lot of time. It's highly recommended!

To remove the read permission of a file for the user you would do

```
$ chmod u-r working.txt
```

Or, you can do something like this:

```
$ chmod 344 working.txt
```

You probably noticed that you can use the `-` (minus) sign to remove permissions from a file. Try reading your file:

```
$ cat working.txt
```

Now you cannot read your own file. Please make the file readable again by you with the `chmod` command. Look at your reference at the start of these tutorial to figure out what permissions are required.

## 2.) PROGRAM EXECUTION, PRIVILEGES AND SUDO

As a general user you can see that there is a file called `/etc/shadow`:

```
$ ls /etc/shadow
```

But, you cannot see its contents:

```
$ less /etc/shadow
```

Figure out the permissions of this file.

As a general user, however, you can see the content of the `/etc/shadow` file if you do the following:

```
$ sudo less /etc/shadow
```

What is `sudo`? Read about it:

```
$ man sudo
```