Operációs rendszerek alapjai

1. gyakorlat

Tematika

- Történelem, linux alapok
- Terminál tartós használata
- ITK-s szerverek, ssh, scp
- Shell script
- Szövegkezelés: grep, sed, awk
- Terminál - internet interakció
- Automatizálás, ütemezés
- Windows, cmd, PowerShell
- Esettanulmányok
- *

Gyakorlat célja

- Kényelmes működés terminálban (platformtól függetlenül)
- Alapvető műveletek elvégzése és automatizálása script segítségével
- Ismerkedés néhány nagyobb hangvételű feladattal
- Programozás és fordítás terminálban

Követelmények

- Beadandók
  - Három darab: 10, 15, 15 pontért
- ZH
  - Félév végén: 60 pontért
- Szorgalmi?
- Ponthatárok
  - 0-59 elégtelen
  - 60-69 elégséges
  - 70-79 közepes
  - 80-89 jó
  - 90-100+ kiváló

Mai gyakorlat

- Történelem
- Linux alapok
  - fájlrendszer
  - alapvető parancsok
  - átirányítások, pipe-ok

Unix

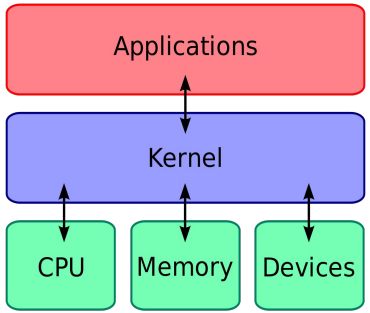
- 1970 körül assembly-ben írták
  - UNICS - Uniplexed Operating and Computing System
  - rendszermag (kernel), héj (shell), szövegszerkesztő és assembler
  - 1973-ban átírták C-re a könnyebb portolás érdekében
  - ma már licenszelhető védjegy, meg kell felelni a specifikációknak
    - pl. Mac OS X
- Moduláris design - a programok interakciója a fontos
- Fontos emberek
  - Ken Thompson
  - Dennis Ritchie
  - Brian Kernighan

GNU - GNU’s not Unix

- 1980-as években Richard Stallman hirdette meg a GNU Project-t
  - Unix-szerű operációs rendszer: nagyjából teljesíti a specifikációkat, de ingyenes
- Programok
  - Emacs - szövegszerkesztő
  - GCC - GNU Compiler Collection (C, C++, Objective C, Fortran stb.)
  - GIMP - GNU Image Manipulation Program
  - GNOME - GNU Network Object Model Environment grafikus felület
  - GRUB - Grand Unified Boot Loader
- Mit jelent, hogy szabad szoftver?
  - szabadon használható, terjeszthető, módosítható
  - (L)GPL - (Lesser) General Public License

Linux

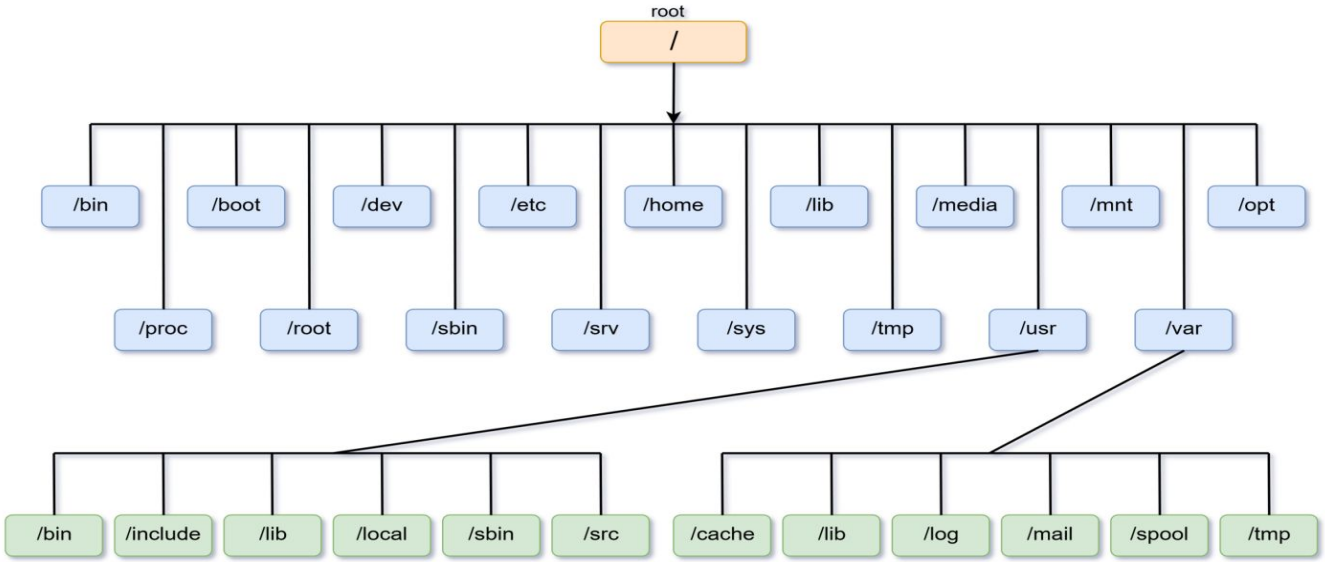
- 1991-ben Linus Torvalds írta főleg C-ben
  - nem voltak elérhető kernelek (GNU kernel is csak akkoriban készült el)
- Open-source GPL-2.0 licenc alatt github link
  - még ma is több ezren szerkesztik Linus menedzselésével
- Személyi operációs rendszernek szánta
  - minden területen domináns, kivéve a személyi felhasználást :(
- Linux kernel felé rengeteg rendszert írnak
  - Android
  - disztribúciók



Linux disztribúciók

- Debian
  - ez van az ITK-s gépeken
- Ubuntu
  - Debian alapú, könnyű Windows után
- Mint
  - Ubuntu alapú, flugi ezt használja (úgy tudom)
- Elementary OS
  - könnyű Mac OS X után
- Arch
  - hardcore
- Manjaro
  - gyakorlatilag egy konfigurált Arch

Linux fájlrendszer - alapvető struktúra



Linux fájlrendszer - fontosabb mappák

/	virtuális könyvtár gyökere
/bin	bináris fájlok
/boot	boot-hoz szükséges fájlok
/dev	eszközmeghajtó fájlok (merevlemezek)
/etc	konfigurációs fájlok
/home	felhasználók könyvtárai
/lib	library-k
/media	hordozható eszközök

Linux fájlrendszer - fontosabb mappák

/opt	third-party szoftverek (MATLAB, Qt stb.)
/root	rendszergazda könyvtára
/run	futási idejű adatok
/sys	hardware információk
/tmp	ideiglenes fájlok
/usr	rendszerszintű programok
/var	gyakran változó fájlok, pl. log

Linux fájlrendszer - inode (index-node)

- Minden fájlhoz tartozik egy inode
  - `stat` paranccsal elérhetjük az inode információit
  - POSIX - Portable Operating System Interface szabvány
    - eszköz ID
    - sorszám
    - linkek száma
    - engedélyek
    - user ID, group ID
    - méret
    - létrehozás, hozzáférés, módosítás időbélyege
    - IO blokkméret
    - elfoglalt blokkok száma

Linux fájlrendszer - linkek

- Hard link
  - adott inode-ra (lemezterületre) mutató link
  - link számláló: ha egy inode-ra nulla hard link mutat, akkor törlésre kerül
  - létrehozás:
    - `link oldfile newfile`
    - `ln oldfile newfile`
- Symbolic link
  - elérési útvonal akár könyvtárra is
  - nem növeli a link számlálót
  - lehet a mutatott objektumot kitörölni és újra létrehozni
  - létrehozás:
    - `ln -s oldfile newfile`

Linux fájlrendszer - fájl típusok

- `ls` parancs
  - aktuális mappa listázása
  - `-l` (long) kapcsolóval több infót is láthatunk (engedélyek, tulajdonos, fájlméret byte-ban stb.)
  - `-h` (human) kapcsolóval ember által értelmezhető fájlméret
- Fájl típusa az első karakter
  - sima fájl: –
  - mappa: `d`
  - symlink: `l`
  - pipe: `p` (később)
  - socket: `s`
  - blokk eszköz: `b` (random access eszközhöz, pl. /dev/sda lemez partícióhoz)
  - karakter eszköz: `c` (soros bemenet vagy kimenet eszközhöz, pl. /dev/null nulleszközhöz)

Linux fájlrendszer - fájl engedélyek

- Következő 3x3 karakter
  - tulajdonos engedélyei
  - csoport engedélyei
  - többi felhasználó engedélyei
- Összesen 9 bit
  - `r`: olvasható
  - `w`: írható
  - `x`: futtatható
- `chmod` - change mode paranccsal módosítható
  - `chmod +x file: -rw-rw-r--` **=>** `-rwxrwxr-x`
  - `chmod 777 file`: bármi **=>** `-rwxrwxrwx`

Navigálás a fájlrendszerben

- `ls` - list
  - néhány kapcsoló:
    - `-a (-A)` - rejtett fájlok kiírása (`.` és `..` kihagyásával)
    - `-R` - rekurzívan listáz
- `pwd` - print working directory
- `cd` - change directory
  - abszolút vagy relatív elérési útvonal
  - speciális mappák
    - aktuális mappa: `.`
    - eggyel fentebbi mappa: `..`
    - felhasználó otthoni mappája: ~

Fájlkezelés

- `touch`
  - időbélyeg módosítása
  - ha nem létezik a fájl, akkor létrehozza
- `cp` - copy
  - `-r` kapcsolóval rekurzívan másol
- `mv` - move
  - átnevezésre is alkalmas
- `mkdir` - make directory
- `rm` - remove
  - `-r` kapcsolóval rekurzívan töröl
- `cat` - concatenate
  - fájlok összefűzése és kiírása

Átirányítások

- `echo` paranccsal kiírathatunk terminálra
- `<, >`
  - felülír fájlokat
  - `echo heloszia > helo.txt`
- `<<, >>`
  - hozzáfűz fájlokhoz
  - `echo heloszia megint >> helo.txt`

Fájl deskriptorok

- 0 - standard input
- 1 - standard output
- 2 - standard error
- Ezekbe és ezekből lehet átirányítani
  - hibák logolása: `command 2>> errors.log`
  - parancs csendes futtatása (nincs kimenet): `command 1> /dev/null`
  - kimenet hibaként megjelenítése: `command 1>&2`

Pipe

- Parancsok/processzek kimenetét beköthetjük más parancsok bemenetére a `|` karakterrel
  - szinkron kommunikáció
  - így elég komplex one-liner programokat is lehet írni
- Egyszerű példa: tegyük fel, hogy az első öt objektum érdekel csak az aktuális mappában
  - `head -n`
    - kiírja egy fájl első n sorát, alapból n=10
    - bemenetet is elfogad
  - `ls -l | head -5`
- Probléma: mi van, ha aszinkron kommunikáció kell?
  - megoldási lehetőség egy ideiglenes fájl használata az üzenet elmentésére és beolvasására
  - merevlemezre írunk, úgyhogy ez lassú és elég fapados

Elnevezett pipe

- Processzek közti aszinkron kommunikációt könnyíti meg
  - segítségével nem kell csinálni egy ideiglenes fájlt, ami tárolja az adatot, hiszen a memóriát használja
- Létrehozás
  - `mkfifo my_pipe`
- Bemenet (több is lehetséges!)
  - `ls -l > my_pipe`
- Kimenet (egyszeri kiolvasás, minden adatot megkapunk)
  - `cat my_pipe`
- Törlés
  - `rm my_pipe`

Hasznos parancsok

- `man command`
  - adott parancs manuál oldalának megnyitása
  - online is elérhető pl. itt
- `apropos text`
  - keresés parancsok összefoglalójában
- `whatis command`
  - parancs összefoglalója

1	parancsok
2	rendszerhívások
3	könyvtári függvényhívások (C nyelv)
4	speciális fájlok (/dev)
5	fájlformátumok és konvenciók
6	játékok
7	egyéb
8	rendszerkarbantartó parancsok
9	kernel rutinok

Következő gyakorlat

- Linux telepítése (akár Windows mellé)
- ITK-s szerverek, ssh
  - otthonról géptermi elérés
- Shell script-k írása
- Terminál alapú szövegszerkesztők:)
- (Környezeti) változók, főleg PATH
- Binárisok

Érdekességek, olvasnivalók

- Real Programmers Don't Use PASCAL
- Linus Torvalds Q&A at Aalto University
- Linus Torvalds Q&A at DebConf14
- Operating Systems: from 0 to 1
  - prerequisites
    - basic concepts of electricity: atoms, electron, proton, neutron, current flow, Ohm’s law
    - C programming
    - Linux basics
      - navigation
      - invoke commands with options
      - piping