

Hálózatok építése és üzemeltetése

Linux

Miért Linux?

UNIX

▶ UNIX operációs rendszerek

- ▶ sok évtizedes sikertörténet
- ▶ nagy teljesítmény, megbízhatóság, robusztusság
- ▶ nagyon sok helyen használják
- ▶ különböző környezetekben, különböző feladatokra
 - ▶ vállalati környezet, szolgáltatók, adatközpontok, beágyazott rendszerek, **router**ek, stb.
- ▶ sokoldalú: számos elérhető program (ld. GNU csomagok)
- ▶ sokféle UNIX és UNIX-like rendszer
 - ▶ pl. FreeBSD, Solaris, Mac OS X (FreeBSD alapú), GNU/Linux

▶ **Hálózatkezelés**

- ▶ első TCP/IP protocol stack
- ▶ jól kidolgozott, hatékony hálózatkezelés
- ▶ testreszabható működés
- ▶ sok kísérleti megoldás itt volt implementálva először

UNIX történet

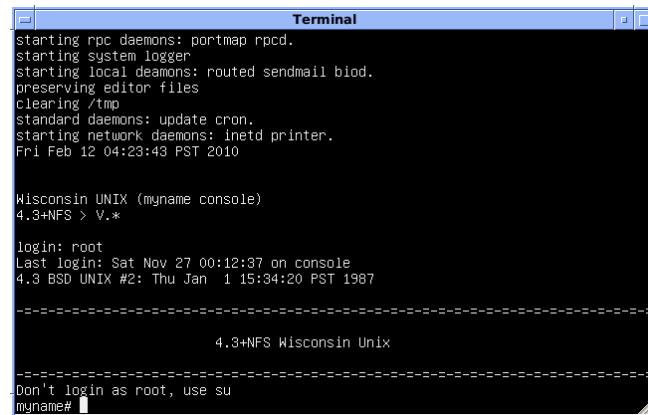
▶ AT&T/Bell Labs

- ▶ 1960-as évek: Multics (Multiplexed Information and Computing System)
 - ▶ MIT, Bell Labs, GE projekt: időosztásos operációs rendszer mainframe-re
 - ▶ túl komplex volt
- ▶ Bell Labs (Ken Thompson, Dennis Ritchie, ...)
 - ▶ kivonultak a Multics projektből
 - ▶ 1969: Unics (Uniplexed Operating and Computing System)
 - K.Thompson 1 hónap alatt implementálta az első Unics-ot PDP-7-re :)
 - kernel, shell, text editor, assembler (1-1 hét alatt)
 - -> Unix (többen tartják saját ötletüknek az elnevezést...)
 - ▶ 1972: C nyelven újraírták (PDP-7 assembly helyett)
 - **hordozhatóság!**
 - több HW architektúrára portolható
 - (a C nyelv sikertörténete szintén fontos volt!)

UNIX történet

- ▶ **Terjedése**
 - ▶ AT&T jogi okok miatt nem léphetett be a computer üzletbe
 - ▶ a forráskódot oda kellett adni, aki kérte
 - ▶ amerikai egyetemek megkapták a forráskódokat, gyorsan terjedt a használata
 - ▶ pl. University of California, Berkeley (BSD)
 - ▶ 1984: Bell Labs leválasztása után fizetős termék lett (System V)
- ▶ **Két fontos UNIX változat a '80-as években (1st phase of the Unix wars)**
 - ▶ **BSD**
 - ▶ Berkeley Software Distribution (University of California, Berkeley)
 - ▶ 1977-től 1995-ig
 - ▶ jelentősebb verziók: 4.2BSD, 4.3BSD, 4.4BSD
 - ▶ **System V**
 - ▶ AT&T belső név: Unix 5.0
 - ▶ System III termék utódja
 - ▶ legfontosabb verzió: System V Release 4 (SVR4)
- ▶ **POSIX (1988-tól)**
 - ▶ Portable Operating System Interface
 - ▶ IEEE "Unix szabvány"

4.2 > V poszterek :)

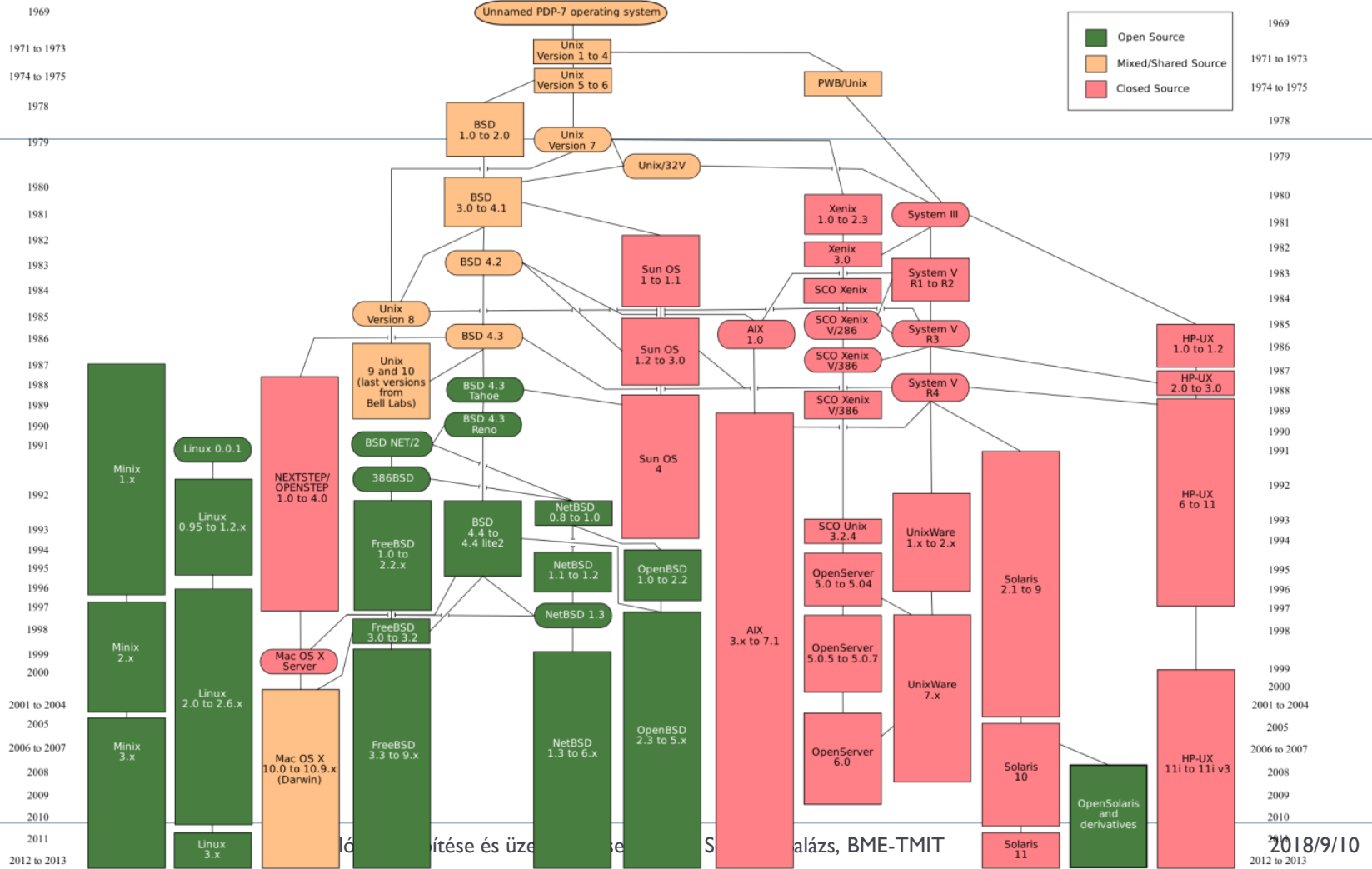


```
Terminal
starting rpc daemons: portmap rpcd.
starting system logger
starting local daemons: routed sendmail biod.
preserving editor files
clearing /tmp
standard daemons: update cron.
starting network daemons: lnxd printer.
Fri Feb 12 04:23:43 PST 2010

Wisconsin UNIX (myname console)
4.3+NFS > V.*

login: root
Last login: Sat Nov 27 00:12:37 on console
4.3 BSD UNIX #2: Thu Jan 1 15:34:20 PST 1987
-----
                                4.3+NFS Wisconsin Unix
-----
Don't login as root, use su
myname#
```

UNIX történet



GNU/Linux

▶ Richard Stallman

▶ GNU Project (GNU's not Unix), 1983

- ▶ cél: teljes Unix-kompatibilis szoftver rendszer megalkotása tisztán szabad szoftver komponensekből
- ▶ számos komponens megvalósult
 - GNU Compiler Collection (GCC), GNU Binary Utilities (binutils), bash shell, GNU C library (glibc), GNU Core Utilities (coreutils), Emacs szövegszerkesztő
- ▶ kivéve a kernel
 - (jelenlegi verzió: GNU Hurd/Mach kernel és mikrokernel)

▶ FSF (Free Software Foundation), 1985

- ▶ egy mozgalom, filozófia

▶ GNU GPL (GNU General Public License), 1989

▶ Linus Torvalds

▶ első Linux kernel: 1991

- ▶ monolitikus kernel (egyetlen nagy program), Intel x86 architektúrára
- ▶ tartalmazott Minix (A. Tanenbaum) és GNU komponenseket

▶ Linux: szigorúan véve csak a kernel

▶ Linux kernel + GNU komponensek: GNU/Linux operációs rendszer

▶ Linux trademark: Linux Foundation

- ▶ üzlet orientált

▶ SZV tárgy: Nyílt forráskódú és szabad szoftverek (vitmav66)

GNU/Linux

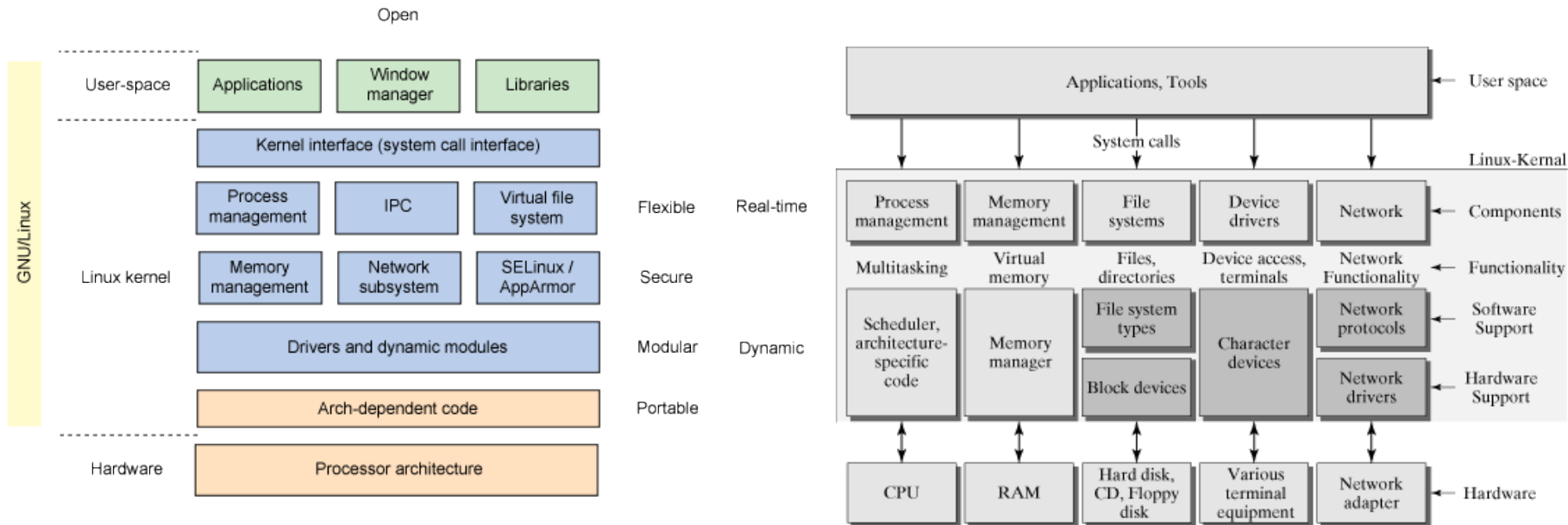
- ▶ Eredetileg: free OS Intel x86 PC-re
- ▶ Ma
 - ▶ a legtöbb HW platformra portolt OS
 - ▶ a legelterjedtebben használt általános célú OS (Android!)
- ▶ Hol használják?
 - ▶ szerverek, szuperszámítógépek, mainframe gépek, adatközpontok
 - ▶ desktop PC, notebook (itt elég kicsi a részesedése...)
 - ▶ beágyazott rendszerek, mobil telefonok, tabletek, TV-k
 - ▶ hálózati eszközök (routerek, switch-ek, stb.)
- ▶ Disztribúciók
 - ▶ “Linux operációs rendszer csomagok”
 - ▶ legfontosabb szabadon elérhető:
 - ▶ Debian GNU/Linux, Ubuntu, Arch Linux, CentOS, Fedora, Gentoo Linux, Linux Mint, openSUSE
 - ▶ egy-két fizetős:
 - ▶ Red Hat Enterprise Linux, SUSE Linux Enterprise Server

GNU/Linux rendszer

A rendszer felépítése

- ▶ A Linux egy
 - ▶ többfelhasználós (multiuser)
 - ▶ többfeladatos (multitasking)
 - ▶ operációs rendszer a Unix-kompatibilis eszközök teljes készletével
- ▶ Három fő rész
 - ▶ **Kernel:** az operációs rendszer központi eleme
 - ▶ szabályozza a rendszer működését, vezérli a perifériákat, a rendszererőforrásokhoz való hozzáférést és olyan alapfeladatokat lát el, mint például a processzek ütemezése vagy a virtuális memória kezelése
 - ▶ mindig privilegizált végrehajtási módban (kernel mode) hajtódik végre, amikor minden fizikai erőforráshoz hozzáfér
 - ▶ kernel módba váltás a rendszerkönyvtárak rutinjaiban levő rendszerhívásokkal történik
 - ▶ **Rendszerkönyvtárak**
 - ▶ standard funkciókat valósítják meg, melyeken keresztül az alkalmazások együtt tudnak működni a kernellel
 - ▶ olyan függvényeket biztosítanak, melyekre több alkalmazásban is szükség van, így azokat nem kell minden programmal együtt külön lefordítani és tárolni, hanem egy közös helyen tárolva bármely program elérheti őket
 - ▶ **Segédprogramok (felhasználói programok)**
 - ▶ a felhasználói igények széles körét lefedő alkalmazások különféle egyedi, illetve speciális feladatokra, mint például: rendszer inicializálás, konfigurálás, állandóan futó programok a hálózati kapcsolatok kezelésére, belépési kezdeményezések kezelésére, naplózásra, stb.

A rendszer felépítése



Shell

- ▶ Egy kiemelt felhasználói program: shell (héj)
 - ▶ “operációs rendszer felhasználói felülete”
 - ▶ sikeres belépés után normál esetben elindul a (felhasználó által beállított) shell program
 - ▶ operációs rendszer számára kiadott parancsok beolvasása, értelmezése
 - ▶ pl. billentyűzet segítségével felhasználói és egyéb programok indítása
 - ▶ nem csak parancsértelmezők: script nyelvek is!

- ▶ Elterjedt shell programok

- ▶ sh: Bourne shell (első volt, szinte minden Unix változatban megtalálható)
- ▶ bash: Bourne Again Shell (Linux default)
- ▶ csh: C shell (szintaxisa a C programozási nyelvre épül)
- ▶ ksh: Korn shell (a Bourne shell és a C shell egyfajta ötvözete)
- ▶ ash: Almquist shell (erőforráshiányos környezetekben használják)

```
[usr@localhost]# cd /etc/
[usr@localhost]# ls
azps          gimp          man_db.conf  rc.d
apache        group         modprobe.conf  resolv.conf
asciidoc       group-        modules      resolv.conf.dhclient
asound.state  grub         nplayer       rpc
ut1           gshadow      nstab        scsi_id.config
blkid.tab     gshadow-     ndiswrapper  services
blkid.tab.old gview        nsswitch.conf shadow
contrab       gtk          passwd       shadow-
default       hostname     passwd       shells
dev.d         hosts        playlist      slim.conf
dhclient.conf hotplug      printcap     ash
dhclient-script  init.d      printcap.sample  ssl
dhept.conf     inittab     profiles      sysctl.conf
environment    inputrc     proftpd.conf  syslog.conf
environment-common iproute2    protocols     system
bsd.conf       ld.so.cache qt            timeformat
feron.allow    ld.so.conf  rartiles.lst  udev
feron.conf     limits      rc.d          vware
feron.deny     locale.gen  rc1.d         wpa_supplicant.conf
fdprn         localtime  rc2.d         x11
fglrxprofiles.csv login.access rc3.d         XF86Config
fglrxrc       login.defs  rc4.d         xinetd.conf
fontz         lpd         rc4.d         xinetd.d
fontastic     mail        rc6.d         xsl
fstab         man.conf    rc.boot       xpdfrc
[usr@localhost]#
```

“Grafikus shell”

- ▶ Desktop environment
 - ▶ minden SW, ami kell a desktophoz
 - ▶ window manager (pl. KWin)
 - ▶ file manager (pl. Dolphin)
 - ▶ toolkitok (pl. Qt, GTK+)
 - ▶ témák
- ▶ Például
 - ▶ KDE
 - ▶ GNOME
 - ▶ Xfce
 - ▶ LXDE
- ▶ Grafikus rendszer
 - ▶ X Window System
 - ▶ (Wayland)

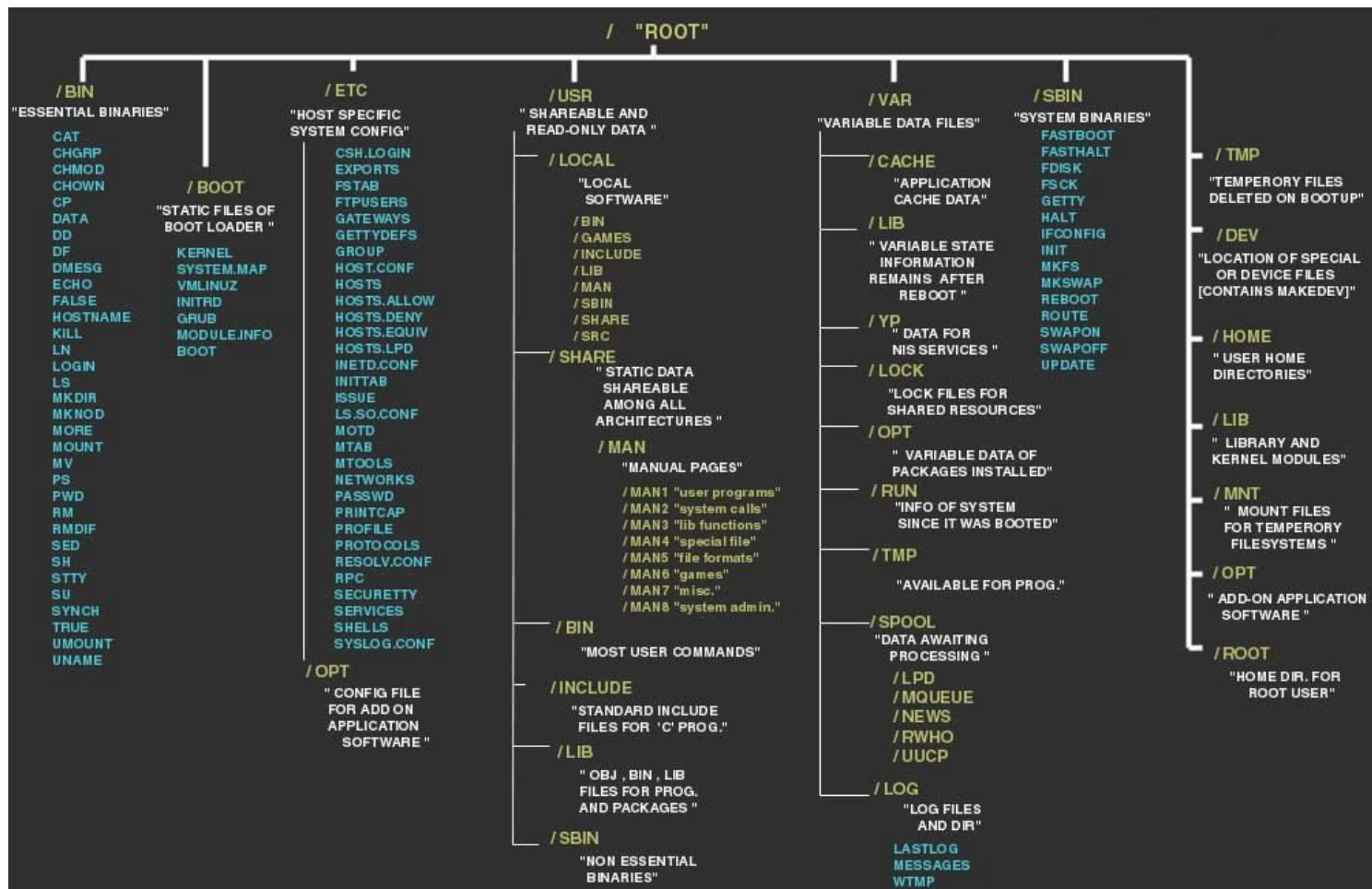


Állományrendszer

- ▶ Unix állományrendszer, fa struktúra
- ▶ Alap koncepciók
 - ▶ minden egy fájl
 - ▶ egységes programozási felület, ugyanazok a parancsok (függvények) használhatók
 - ▶ egyetlen (virtuális) fájlrendszer
 - ▶ minden valós fájlrendszer, hardvereszköz, processz ez alatt
 - ▶ tetszőleges kötetek csatolhatók (mount) a fa struktúra tetszőleges pontjához
 - ▶ típusok: plain, device, directory, socket, symbolic link, ...

Állományrendszer

- ▶ Egy gyökér (root, “/”), minden ez alatt
 - ▶ felépítés: FSH (File System Hierarchy) szabvány szerint
 - ▶ /bin: legfontosabb futtatható programok
 - ▶ /boot: tömörített kernel image-ek (vmlinuz*) és a kernellel kapcsolatos segédfájlok
 - ▶ /dev: hardverekhez tartozó fájlbejegyzések
 - ▶ /etc: konfigurációs fájlok helye (a konfigurációs fájl általában a csomag nevével egyezik meg)
 - /etc/init.d/: A rendszer szolgáltatásait kezelő scriptek. Minden itteni script azonos paraméterekkel hívható meg (start, stop, restart)
 - /etc/fstab: fájlrendszer beállításai
 - ▶ /home: felhasználói könyvtárak (pl. /home/jozsi) (pl. bash esetén “~” karakterrel is lehet hivatkozni)
 - ▶ /lib: A /bin alatti programok által használt megosztott könyvtárak (shared library-k) gyűjtőhelye.
 - ▶ /mnt: az itt levő alkönyvtárakba szokás becsatolni (“mountolni”) a különböző fájlrendszereket.
 - ▶ /proc: virtuális fájlrendszer, interfész a kernel felé; egyes fájlokból a rendszer aktuális állapota olvasható ki (pl. /proc/cpuinfo: cpu információ lekérdezése), másokba írva a kernel beállításait változtathatjuk meg (pl. /proc/sys/net/ipv4/ip_forward: IP forwarding engedélyezése vagy tiltása)
 - ▶ /root: A rendszergazda home könyvtára
 - ▶ /sbin: rendszergazda által használt binárisok; pl. mke2fs, cfdisk, init, ...
 - ▶ /tmp: ideiglenes (temp) fájlok helye; mindenki számára írható, tartalma időnként (beállítástól függően) törlődik.
 - ▶ /usr: Unix System Resources, a programok saját könyvtárai, fájljai vannak itt. Alatta majdnem a teljes FSH le van duplikálva
 - ▶ /var: gyakran változó tartalmú adatfájlok vannak itt; pl. naplófájlok (logok), mailspool, print spool, ...



Védelmi rendszer

- ▶ Többfelhasználós rendszer
- ▶ védeni kell a
 - ▶ fájlokat a háttértárolón
 - ▶ processzeket a memóriában
- ▶ felhasználók, csoportok (users, groups)
- ▶ minden felhasználónak
 - ▶ egyedi azonosítója van
 - ▶ több csoporthoz is tartozhat
- ▶ fájl
 - ▶ egy tulajdonosa van (user) és van egy felhasználói csoportja
 - ▶ módosítás: `chown user:group file`
 - ▶ és védelmi kódja, ami szabályozza, hogy ki olvashatja, írhatja, futtathatja
 - ▶ lekérdezés: `ls -l`
 - ▶ `-rwxr-xr-- 1 user group ...`

Védelmi rendszer

▶ Jogosultságok állítása

- ▶ `chmod [u|g|o] [+|-|=] [r|w|x|...] file`
- ▶ egyes felhasználói körökhöz (u/g/o, tulajdonos/csoport/mások)
- ▶ jogosultságok (r/w/x, olvasás, írás, végrehajtás)
- ▶ adhatók (+), illetve elvehetőek (-)

- ▶ Példa: tulajdonosnak és a csoportjának írási és végrehajtási jogot is adunk (az eddigi jogok mellé):

- ▶ `chmod ug+wx file`

▶ Katalógus (könyvtár)

- ▶ végrehajtás helyett keresési (listázási) jog
 - ▶ fájl védelme nem függ az őt tartalmazó katalógus védelmétől!

Alapvető parancsok

HF: részletesen átnézni a gyakorlatra

Parancsok

- ▶ Parancsok a shell-ben adhatók ki
- ▶ általános felépítésük:
 - ▶ `parancs kapcsolók argumentumok ...`
 - ▶ kapcsolók legtöbbször “-” jellel kezdődnek
 - ▶ ha fájlnev helyén áll “-”, akkor a standard inputot vagy standard outputot jelenti
 - ▶ pl: `ls -l *.txt` (részletes lista a .txt végződésű fájlokról)

Hasznos parancsok

▶ **man**

- ▶ leghasznosabb parancs, minden UNIX-alapú rendszer részét képezi
- ▶ felhasználói kézikönyv, mely az összes parancs, függvény, API hívás leírását tartalmazza, valamint a főbb konfigurációs fájlokat
- ▶ a man oldalakról a q billentyű lenyomásával lehet kilépni
- ▶ pl: `man ls`

▶ **mc**

- ▶ Midnight Commander: könnyen használható fájlkezelő program sok hasznos segédfunkcióval

▶ **shutdown, halt, reboot, poweroff**

- ▶ rendszer leállítása vagy újraindítása

▶ **su, su username; sudo, sudo -u username**

- ▶ superuser jogosultság megszerzése (su), illetve adott felhasználói jogosultság megszerzése
- ▶ sudo használatával egy parancs hajtható végre az adott jogosultsággal

Fájlrendszerrel kapcsolatos műveletek

▶ **cd, pwd, mkdir, rmdir, ls, find, tar**

- ▶ Könyvtárműveletek: aktuális könyvtár megváltoztatása, aktuális könyvtár kiírása, könyvtár létrehozása, üres könyvtár törlése, könyvtár tartalmának kiírása, fájlok keresése a könyvtár hierarchiában nevük vagy tulajdonságaik alapján, könyvtárak archiválása vagy visszaállítása
- ▶ nem üres könyvtárak törlésére a fájl törlési parancs rekurzív változatát kell használni (rm -r)
- ▶ pl: az aktuális könyvtártól (".") rekurzívan keresi a .html fájlokat és a talált fájlokról részletes információt ad (mindegyik találatra végrehajtja az ls -l parancsot):
 - ▶ `find . -name '*.html' -exec ls -l '{}' \;`

▶ **touch, rm, cp, mv, ln**

- ▶ Fájl műveletek: fájl létrehozása vagy "megérintése" (dátumok aktuálisra állítása rajta), törlése, másolása, mozgatása, linkelése (szimbolikus link létrehozásához használjuk a -s opciót)
- ▶ cp, mv, ln paraméterei mindig forrás – cél sorrendben követik egymást
- ▶ -r (vagy -R) kapcsolóval lehet rekurzívan (alkönyvtárakkal együtt) végeztetni fájl műveleteket
- ▶ pl: szimbolikus link létrehozása a szülő könyvtárban elhelyezkedő prog fájlra proglink néven:
 - ▶ `ln -s ../prog proglink`

▶ **df, du, fsck, mount, umount**

- ▶ Adminisztratív műveletek: fájlrendszerek diszkhasználata, adott könyvtárak diszkhasználata (alkönyvtárakkal együtt), fájlrendszer ellenőrzése, fájlrendszer csatolása, illetve leválasztása.

Processzkezelés

▶ **ps, kill, top**

- ▶ processzlista kiíratása, egyes processzek megölése (signal küldése)
- ▶ leggyakoribb kombináció a ps aux
- ▶ a kill parancs paramétere a ps által mutatott process ID
- ▶ ha a kill hatására nem hal meg a processz, úgy a kill -9 kombináció még segíthet (ez a kötelező érvényű felfüggesztés jelzése)
- ▶ a top paranccsal monitorozhatjuk az éppen aktuális processzeket, kilépés q billentyűvel.

▶ **watch, sleep**

- ▶ watch a paraméterében megadott parancsot adott időnként (alapértelmezésben ez 2 sec) lefuttatja, és a kimenetét megmutatja
- ▶ sleep adott ideig alszik (az időt másodpercekben kell megadni).

Szűrők

HF: részletesen átnézni a gyakorlatra

Szűrők

- ▶ Nagyon hasznos Unix eszközök
- ▶ egyszerű programok
 - ▶ standard bemenetüket a megfelelő művelet elvégzése után a standard kimenetükre másolják
 - ▶ sok önmagában nagyon egyszerű műveletet megvalósító szűrő van
 - ▶ általában a segédprogramok képesek szűrőként is működni
 - ▶ szűrők egymás után kapcsolhatók a pipe (csővezeték) segítségével
- ▶ shell által végrehajtott programok alapból 3 megnyitott állománnyal indulnak
 - ▶ standard input (0)
 - ▶ standard output (1)
 - ▶ standard error (2)
 - ▶ ezek átirányíthatók

Szűrők

▶ Bemenet/kimenet átirányítás

- ▶ `prog < file`: standard input átirányítása (vagy hosszabban: `0<`)
- ▶ `prog > file`: standard output átirányítása (vagy hosszabban: `1>`)
- ▶ `prog 2> file`: standard error átirányítása
- ▶ `prog 2>&1`: standard error átirányítása standard outputba
- ▶ `prog 1>&2`: standard output átirányítása standard errorba
- ▶ `prog1 | prog2`: pipe, `prog1` kimenetének `prog2` bemenetére irányítása
- ▶ `prog >> file`: standard output hozzáírása (append) a megadott fájlhoz

Egyszerű példák

▶ **echo, cat, tee**

- ▶ Paraméterként átadott szöveg kiírása (echo), illetve fájlok kiírása és összefűzése (cat). Gyakran használjuk a standard output átirányításával, vagy pipe-okkal együtt. A tee parancs a standard inputról másol a standard outputra, valamint a paraméterként megadott fájlokba is (adatfolyam elágasztása).
- ▶ pl: cat program kimenetét átirányítjuk az f1 fájlba, így a standard bemeneten bevitt sorok az adott fájlba íródnak egészen a fájlvége jel (ctrl-d) beviteléig:
 - ▶ `cat >f1`

▶ **more, less**

- ▶ Fájlok kiírása úgy, hogy egyszerre egy képernyőnyi tartalom jelenik meg, illetve navigálási lehetőség biztosítása. A less a kifinomultabb változat.

▶ **head, tail**

- ▶ Fájlok első (head), illetve utolsó (tail) n sorának kiírása.

▶ **tr**

- ▶ Alapértelmezésben karakterfordítást végez (translate): az első paraméterként megadott karaktereket cseréli a második paraméterben megadottakra. Tartomány is megadható, pl. [0-9] a számokat jelenti, [a-z] a kisbetűket. Ha az első paramétere -d, akkor törli a második paraméterben megadott karaktereket. Pipe részeként vagy átirányítással használjuk.

▶ **wc**

- ▶ Kiírja a sorok, szavak és karakterek számát ("word count").

Egyszerű példák

▶ **cmp, diff, comm**

- ▶ Fájlok összehasonlítása: bájról bájtra (cmp) vagy szöveges fájlokat sorról sorra (diff). A comm parancs két fájl közös sorainak kiíratására használható.

▶ **sort, uniq**

- ▶ A két parancsot általában együtt (egymásba pipe-olva) használjuk és ilyen sorrendben: a sort rendezi a bemenetet, míg a uniq a rendezett bemenet ismétlődő soraiból csak egyet hagy meg.
- ▶ A uniq paraméterezésével többféle működés is elérhető, pl. a sorok különféle számolása (pl. uniq -c), csak a többször szereplő (uniq -d), vagy az egyedi sorok kiíratása (uniq -u). A sort paraméterezésével számok és stringek rendezése is megoldható.
- ▶ Például az alábbi egymás után kapcsolt szűrők a jelszófájlt rendezik a 3. oszlop szerint (-k3) numerikusan (-n) csökkenő sorrendben (-r) és az utolsó két sor lesz az eredmény (tail -n 2). A jelszófájlból a mezők közti szeparátor a :, ami a rendezésnél a -t kapcsolóval adható meg:
 - ▶ `cat /etc/passwd | sort -t: -n -k3 -r | tail -n 2`

grep

▶ **grep, egrep, fgrep**

- ▶ Reguláris kifejezés-illesztő. A paraméterben (idézőjelben!) megadott reguláris kifejezésre (regexp) illeszti a bemenetet.
- ▶ Fontosabb paraméterei:
 - ▶ -A, -B: az illesztett sor környezetét (előző / következő, adott számú sorokat) is megmutatja
 - ▶ -v: fordított működést eredményez (nem illesztett sorokat mutatja)
 - ▶ -q: nincs output, csak a visszatérési értéket állítja be (if feltételeként szoktuk használni)
- ▶ grep és sed parancsok reguláris kifejezéseiben az operátorokat "escape-elni" kell
 - ▶ (különbön karakternek tekinti őket a program)
 - ▶ pl: "\|" a vagy operátor, míg "|" a pipe karakter
- ▶ példák:
 - ▶ `$ echo "bcd" | grep "a.*"`
 - ▶ `$ echo "bcacb" | grep "a.*b"`
 - ▶ `bcacb`

sed

▶ sed

- ▶ Teljes funkcionalitását tekintve sorszerkesztő, mi reguláris fordítóként fogjuk használni.
- ▶ Erre az s parancsa szolgál:
 - ▶ `s/kif1/kif2/`
 - ▶ `kif1` reguláris kifejezést fordítja `kif2` kifejezésre mindazon sorokon, amelyekre `kif1` illeszkedik.
 - ▶ a(z escape-elt) zárójelbe tett kifejezésrészekre vissza lehet hivatkozni `kif2` -ben a `\1`, `\2`, ... referenciákkal.
 - ▶ ha az s parancs záró / -je után még egy `g` paramétert írunk, akkor soronként többször is végez illesztést
- ▶ Példák:
 - ▶ `$ echo 'xxxaaaaxxx' | sed 's/aaa/bbb/'`
 - ▶ `xxxbbbxxx`
 - ▶ `$ echo "a0001b" | sed 's/a\([0-9]*\)b/x\1y/'`
 - ▶ `x0001y`

Reguláris kifejezések

- ▶ `c` Maga a `c` karakter, ha az nem speciális karakter.
- ▶ `\c` Kikapcsolja a `c` karakter speciális jelentését. Pl: `\[`: zárójel kezdődik
- ▶ `^` Sor eleje.
- ▶ `$` Sor vége.
- ▶ `.` Egy darab bármilyen karakter. (Az újsor kivételével minden karakter illeszkedik rá.)
- ▶ `[abc]` Bármelyik karakter a halmazból.
- ▶ `[^abc]` Bármelyik karakter, amelyik nincs a halmazban.
- ▶ `[a-z]` Bármelyik karakter a megadott tartományból.
- ▶ `r*` `r` reguláris kifejezés tetszőlegesen sokszor (akár 0-szor).
- ▶ `r+` `r` reguláris kifejezés 1-szer vagy sokszor. (extended regexp)
- ▶ `r?` `r` reguláris kifejezés 0-szor vagy 1-szer. (extended regexp)
- ▶ `r1r2` `r1` és `r2` egymás után úgy, hogy `r1` a lehető leghosszabban illeszkedjen.
- ▶ `r1|r2` `r1` vagy `r2`. (extended regexp)
- ▶ `(...)` egymásba ágyazott kifejezések. (extended regexp)
- ▶ `r{n}` `r` reguláris kifejezés `n`-szer megismétlődik. (extended regexp)
- ▶ `r{n,}` `r` legalább `n`-szer megismétlődik. (extended regexp)
- ▶ `r{n,m}` `r` legalább `n`-szer, legfeljebb `m`-szer megismétlődik. (extended regexp)
- ▶ `\(r\)` `r` reguláris kifejezés önmaga, amire később hivatkozni lehet `\n` alakban.
- ▶ `\n` hivatkozás az `n`-edik `\(r\)` reguláris kifejezésre.

Reguláris kifejezések: példák

- ▶ Az `/etc/passwd` fájlból írassuk ki az összes olyan sort, amelyben az 'r' és a 't' karakterek között tetszőleges számú 'o' szerepel:
 - ▶ `$ cat /etc/passwd | grep 'ro*t'`
- ▶ Írassuk ki az aktuális könyvtár összes olyan könyvtárát, amihez mindenkinek írási joga van:
 - ▶ `$ ls -l | grep '^d.....w.'`
- ▶ Írassunk ki minden olyan sort, amiben egymás után szerepel ugyanaz a betű:
 - ▶ `$ cat /etc/passwd | egrep '(.)\1'`
- ▶ Cseréljük le minden `.conf` fájlnev részletet `.CONFIG`-ra a kimeneten:
 - ▶ `$ ls | sed s/.conf/.CONFIG/`

Reguláris kifejezések: példák

- ▶ Első és második karakter felcserélése egy fájlban:

- ▶ `$ cat file1 | sed 's/\(.\)\(.\)/\2\1/'`

- ▶ Jelszófájl első két mezőjének felcserélése (mező szeparátor a kettőspont):

- ▶ `$ cat /etc/passwd | sed 's/^\([^:]*\) : \([^:]*\) : /\2:\1:/'`

- ▶ Számoljuk meg melyik login shell hányszor szerepel az /etc/passwd fájlban (utolsó oszlop), majd ezt rendezzük csökkenő sorrendbe és írjuk ki a két legelsőt:

- ▶ `$ cat /etc/passwd | sed 's/.*:\([^:]*\) /\1/' | sort | uniq -c | sort -n -r | head -n2`