Dr. Mileff Péter

Unix/linux operációs rendszer üzemeltetése

3. Előadás

Miskolci Egyetem
Általános Informatikai Tanszék

Linkek

- Sok operációs rendszerben:
  - egy az egyes összerendelés van az állományok és az állománynevek között.
  - Minden állománynak egy neve van és minden állománynév egy állományt jelöl.
- A Unix a nagyobb rugalmasság érdekében szakított ezzel a koncepcióval:
  - Az állomány egyedi azonosítója az inode-ja.
  - Ez tartalmaz minden információt az állományról.
    - jogok, méret, hivatkozások száma, stb.
  - A katalógusok csak egy fájlnév – inode összerendelést tartalmaznak:
    - ezért lehetséges, hogy egy katalógusból több néven, vagy különböző katalógusokból hivatkozzunk ugyanarra az inode-ra.
  - Ezt a Unix-ban **link**nek nevezik.

Linkek

- **Hard link: „közvetlen hivatkozás”**
  - Amikor különböző névvel hivatkozunk az inode-ra.
  - Ekkor az állomány addig létezik, ameddig van rá mutató bejegyzés.
  - Hard linket csak egy köteten (partíció, eszköz) belül hozhatunk létre.
- **Szoft link (szimbolikus link): „közvetett hivatkozás”**
  - Ilyenkor különböző nevekkel hivatkozhatunk egy katalógus bejegyzésre.
  - Ilyen módon különböző kötetek között is létrehozhatunk linkeket.

Példák

- Mindkét linket az ***ln*** paranccsal lehet létrehozni.
- **Soft link:** # ln -s ./alma ./körte
  - Ekkor a bármely olyan kérés, amely az „*alma*” jegyzékre vonatkozik, az a „*körte*” jegyzékre fog vonatkozni.
- **Hard link:** # ln ./alma.txt ./körte.txt
- **Hol használják?**
  - **Ezekkel a megoldásokkal fedik el az egyes függvénykönyvtárak verzióváltozásait.**
  - **Példa:**

    /usr/lib/libusb.so → libusb-0.1.so.4.4.4

A /proc fájlrendszer…

A /proc fájlrendszer

- **Nem valódi állományrendszer:**
  - a Linux rendszermag által szimulált jegyzékbejegyzéseket és adatokat tartalmazza.
- Az itt található állomány és jegyzék nem a merevlemezen tárolódnak:
  - csak a memóriában léteznek, és csak akkor, amikor éppen használjuk őket.
  - a jegyzék tartalmát egy ls -l paranccsal kilistázzuk, akkor nulla lesz a mérete.
  - Amikor a proc állományrendszeren belül található állományt olvassuk, a Linux rendszermag által a kéréskor előállított adatokat olvassuk.

A /proc fájlrendszer

- Ha a program ilyen szimulált állományba ír, az adatokat a rendszermagnak küldi el.
  - amely azokat értelmezi és felhasználja.
- A proc állományrendszer ablak a rendszer magja felé, kapcsolattartási felület.

- Néhány fájl:
  - cpuinfo,devices,interrupts,ioports,kcore,locks,
  - meminfo,mouns,partitions,pci,sound,stat,swaps,
  - uptime,version.

A /proc fájlrendszer

- Példa: nézzük meg a ***version*** fájl tartalmát!
- Az egyik fájl (**kcore**) mérete azonban nem nulla:
  - egészen nagy.
  - Ez a rendszermag egyfajta tükre.
  - A fájl nagyjából akkora, amekkora fizikai memóriával rendelkezünk.
  - Ez nem véletlen egybeesés: a kcore a rendszer memóriája, a RAM, minden megtalálható itt, ami az adott pillanatban a fizikai memóriában van,
    - méghozzá folyamatosan frissülve.

Fájlrendszer létrehozása...

Fájlrendszer létrehozása

- Linux/Unix rendszereken több program is használható
  - lemezrészek létrehozása és törlése.
- Az ilyen jellegű programok közül az **fdisk** minden Unix/Linux-ot futtató számítógépen elérhető.
  - Egy interaktív alkalmazás.
- Indítás: egy merevlemezhez tartozó blokkeszköz-meghajtó megadása paraméterként.
  - Pl.: # fdisk /dev/sda
    Command (m for help):

Fájlrendszer létrehozása

- Különféle parancsokkal módosíthatjuk a merevlemez lemezrészeinek szerkezetét.
  - Ha elértük a célunkat, kiírhatjuk az eredményt a merevlemezre.
  - Az fdisk csak a mentéskor változtatja a merevlemez tartalmát.
  - Ha valamit elrontottunk, bármikor kiléphetünk a változtatások elvetésével, megőrizve az eredeti elrendezést.
  - A lemezrészek elrendezésének mentése után nincs szükség a számítógép újraindítására,
    - a módosítások azonnal életbe lépnek, az új lemezrészek használhatók

Mikre képes az fdisk?

- Új partíció létrehozása, létező törlése, partíciók listázása, partíciós tábla ellenőrzése.
- Információ partíciókról:
  # fdisk -l

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	940	1895008+	83	Linux
/dev/hda2		941	1023	167328	5	Extended
/dev/hda5		941	1001	122944+	82	Linux swap
/dev/hda6		1002	1023	44320+	83	Linux

mkfs

- **Fájlrendszer tényleges létrehozása**

mkfs –t <fájlrendszer típus> <eszköz>

- Példa egy ext3 fájlrendszer létrehozására:

mkfs –t ext3 /dev/sda7

- Egy frontend, valódi parancs:
  - mkfs.ext3

A csereterület…

Csereterület

(swap space)

- Szokás látszólagos memóriának (*virtual memory)* is nevezni.
- Nem valódi állományrendszer, állományok tárolására nem alkalmas.
- **Cél:** az operatív memória tehermentesítése, kiterjesztése.
- A Linux/Unix rendszermag képes arra:
  - hogy a memóriának az éppen nem használt területeit a csereterületre mentse,
  - és így memóriát szabadítson fel.
  - természetesen automatikusan visszatölti a mentett memóriatartalmat, amint arra szükség van.

Csereterület

(swap space)

- Automatizált feladat:
  - a mentést és visszatöltést a Linux automatikusan elvégzi,
  - semmiféle feladat nem hárul a rendszeren futó programokra vagy a rendszert használó felhasználókra.
- Hátránya:
  - nagyságrendekkel lassabban működik, mint a számítógép operatív memóriája.
  - Adatok mozgatása a csereterület és a memória között
    - Kevés adat,gyors működés.
    - Sok adat, lassú működés.

A csereterület mérete

- A csereterület mérete dinamikusan állítható.
  - Akár ki is kapcsolható.
- Méretére nincs egyértelmű szabály.
- Nem szerencsés:
  - ha a csereterület nagysága meghaladja az operatív memória nagyságának háromszorosát.
- A legtöbb rendszeren:
  - az operatív memória nagyságának kétszeresénél nem nagyobb csereterületet használunk.
- Létrehozhatjuk külön partíción vagy külön fájlban is.
  - Telepítésekor szokás választani.
  - Később is módosítható.

Csereterület példa

```
$ cat /proc/swaps
Filename    Type        Size    Used    Priority
/dev/hda3   partition   265064  38184   -1
```


Csereterület létrehozása
(külön fájlként)

- Ahhoz, hogy swap területet fájlon használjunk, meg kell határoznunk a méretét
  - max. 2 GB x86 rendszeren.
- Létrehozása:
  - dd parancs nyújt segítséget.
  - amellyel könnyen és gyorsan létrehozhatunk tetszőleges nagyságú üres virtuális lemez image-et.

```
dd bs=1024 count=1M if=/dev/zero of=/path/to/swapfile.n
```

  - 1GB-os fájlt (1 MB * 1024 bytes) hoz létre a "/path/to/swapfile.n"-on

Csereterület létrehozása
(külön fájlként)

- Ezt meg lehet csinálni többször is, ha több swap fájl akarunk használni.
  - maxium 32 darab lehetséges.
- A swap fájlok nem lehetnek elszórtan,
  - az egészet allokálni kell a használat előtt.
- Ha ez sikerült, akkor meg kell formázni (inicializálni),
  - „**swap header**” információval kell ellátni:

```
# mkswap /work/swapfile.1
```

Csereterület létrehozása
(külön partíción)

- Előzetes tervezést igényel
  - még a merevlemez felosztása (partícionálás - fdisk) során.
- A swap terület, amely partíción foglal helyet, be van jegyezve a /etc/fstab fájlba.
  - Például: /dev/hda2 none swap defaults 0 0
- Miután a swap terület allokálva van,
  - meg kell formázni (inicializálni). Ezt csak egyszer kell elvégezni:

mkswap /dev/hda2

Csereterület be/kikapcsolása

- A partíción levő swap terület a boot folyamat init részében automatikusan aktiválódik.
- Az a swap területe, amely fájlrendszeren helyezkedik el, szükséges engedélyezni.
  - Erre minden bootolás során szükség van,
  - célszerű egy init szkriptet írni.
- Swap terület bekapcsolása: a **swapon** parancs.

swapon /work/swapfile.1

- Kikapcsolás:

swapoff /work/swapfile.1

Az *fstab* fájl...

Az fstab fájl

- Szerepe:
  - az itt lefektetett szabályok megkönnyítik az állományrendszerek használatát.
- Megléte rendkívül fontos:
  - az fsck és a mount (tehát a boot során az automata mount is) ebből tájékozódik az elérhető állományrendszerekről.
- Helye: /etc/fstab. Szöveges állomány.
- Tartalma statikus
  - (elvileg programok csak olvashatják) és kézzel kell elkészíteni.

Az fstab felépítése

- Példa:

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    defaults        0       0
/swapfile       none            swap    sw              0       0
/dev/hda1       /               jfs     defaults,errors=remount-ro      0       1
/dev/hda5       /mnt/data       ext3    defaults        0       2
/dev/hdc        /media/cdrom0   iso9660 ro,users,noauto,utf8    0       0
/dev/fd0        /media/floppy0  auto    rw,users,noauto 0       0
/dev/hdb1       /mnt/adat       vfat    defaults,rw,users,uid=0,gid=6,umask=0003,codepage=852,utf8
/dev/hdb2       /mnt/winnt      ntfs    defaults,ro,users,uid=0,gid=6,umask=0003,utf8,noatime   0
/dev/sda1       /mnt/usbcam/    vfat    defaults,noauto,users,noatime   0       0
```


Az fstab felépítése

- **<u>Tulajdonságok:</u>**
  - **Minden mount point-ot (csatlakozási helyet) egy külön sor képvisel**
  - minden opció tetszőleges mennyiségű szóközzel (vagy tabulátorral) van elválasztva.
  - Megjegyzések beszúrása a # jellel jelezve.
- **Első oszlop:**
  - megmutatja, hogy fizikailag hol helyezkedik el a kezelendő állományrendszer.
  - Alapesetben ez egy device (/dev-ben), de lehet még NFS kötet is.
  - Kevésbé statikus megoldás (pl.: pendrive) esetén
    - nagyon hasznos, hogy a mount pontra lehet hivatkozni a volume label (kötetnév) vagy az UUID alapján.

Az fstab felépítése

- **Második oszlop:**
  - megadja, hogy hová kerüljön mountolásra az állományrendszer (ez a mount point).
  - A rootfs (felsőszintű állományrendszer) mindig a / -be mountolódik.
  - Speciális a swap, aminél ez "none" értékkel rendelkezik.
  - A merevlemezeket a /mnt jegyzékbe szokás mountolni, a hordozható lemezeket a /media-ba.

Az fstab felépítése

- **Harmadik oszlop:**
  - megadja az állományrendszer típusát.
  - pl.: ext2, ext3, iso9660, jfs, reiserfs, vfat, xfs
    - csak olyan típusokat írhatunk be, amiket támogat az aktuális futó kernel.
  - Ezt a /proc/filesystems alatt lehet ellenőrizni.
  - Erről is informál a disktype.
  - A floppyt és más hordozható lemezeket (pl.: pendrive) auto-ra szokás állítani a jobb kompatibilitás érdekében.
  - Megjegyzés: vesszővel elválasztva megadhatunk több fajta állományrendszert is, pl.: jfs, ext3.

Az fstab felépítése

- **Negyedik oszlop:**
  - az állományrendszer specifikus mountolási beállítások sorát tárolja.
  - itt van lehetőség egy kis tuningra.
  - A lista elemeit vesszővel kell elválasztani, a lehetőségről a mount manpage tudósít.
  - Néhány példa:
  - **ro/rw:** csak olvasható / olvasható és írható az állományrendszer
  - **exec/noexec:** a bináris állományok futtathatóságát engedi / tiltja
  - **defaults**: az általános működés beállításai: rw suid dev exec auto nouser async

Az fájlrendszer épsége…

Fájlrendszerek épségének ellenőrzése

- A fájlrendszerek viszonylag bonyolult felépítésűek, ezért hajlamosak a meghibásodásra.
- Egy ellenőrzés során a cél mindig azonos:
  - **a fájlrendszer tüzetes átvizsgálásával megtalálni az esetleges problémákat, inkonzisztenciákat**.
- A helyessége és érvényesség UNIX/Linux rendszerekben az ***fsck*** paranccsal ellenőrizhető:
  - Képes megtalálni a problémákat,
  - képes kijavítani,
  - figyelmeztet a nem javítható hibákra.

Fájlrendszerek épségének ellenőrzése

- A fájlrendszert megvalósító kód nagyon sokat tesztelt
  - ezért nagyon ritkán akad probléma. Többnyire:
  - áramkimaradás, hardverhiba vagy kezelési hiba (nem megfelelő rendszerleállítás) miatt keletkezik.
- A legtöbb rendszerben
  - rendszerindításkor automatikusan futtatja az fsck programot.
  - De csak azon fájlrendszereknél, amelyek automatikusan kerülnek felcsatolásra.
  - Az fsck parancsot kézzel kell indítanunk a többi fájlrendszer (például hajlékonylemezek) esetén

Fájlrendszerek épségének ellenőrzése

- Sérült fájlrendszer használata esetén a hibák csak szaporodnak.
- Ha a metaadatokban van esetleg a probléma,
  - a fájlrendszer használata még több adatvesztéshez vezethet!
- Trükkök, hogy elkerüljük az állandó ellenőrzést boot során:
  - ha létezik az /etc/fastboot fájl, nem történik ellenőrzés.
  - az ext2-es fájlrendszereknek van egy speciális jelzése a szuperblokkban
    - jelzi, hogy a fájlrendszer szabályosan lett-e lecsatolva az utolsó felcsatolás óta

fsck futtatása

- **Nagyon fontos:** az fsck programot csak lecsatolt fájlrendszeren szabad alkalmazni!
  - Kivétel ha olvasható módban van felcsatolva.
- **Oka:** a program a nyers lemezblokkokkal dolgozik
  - úgy módosíthatja a fájlrendszert, hogy azt az operációs rendszer nem veszi észre.
- Az fsck parancs csak egy front-end:
  - Minden fájlrendszerhez saját parancs társul:
  - Pl.: # fsck.ext2 , fsck.ext3, stb.

Példák:

- Ext2 fájlrendszer ellenőrzése:
  # fsck.ext2 /dev/hda7 vagy # fsck –t ext2 /dev/hda7
- Ext3 fájlrendszer ellenőrzése:
  # fsck.ext3 /dev/hda7 vagy # fsck –t ext3 /dev/hda7
- Bővebb információ a fájlrendszerről:
  - Dump2fs
- Fájlrendszerek hangolása:
  - tunefs

Köszönöm a figyelmet!