3 Laborgyakorlat

Merevlemezek a linux alatt

Linux alatt a merevlemezeket, partíciókat (a több részre osztott winchester részei) nem a szokásos betűjelekkel jelöljük. Más nevet használunk ha az egész lemezre, vagy ha a lemez részeire (partíciókra) akarunk hivatkozni. A „/”alatt van egy „dev” (devices=eszközök) könyvtár. Ebben a könyvtárban vannak felsorolva fájlokként a lemezre (vagy részeire) hivatkozó ún. eszközfájlok. Attól függően, hogy a lemez milyen interfészen csatlakozik a rendszerünkre, más és más neve van.
Régebben csak IDE (PATA) és SCSI eszközök voltak. Ha egy winchester, vagy cdrom IDE eszközként csatlakozik a rendszerhez, akkor a neve /dev/hd{a,b,c,d,e,f,g,h}. Ha SCSI eszközként, akkor a neve /dev/sd{a,b,c,d,e,f,g,h,...}. Időközben születtek új eszközök és új interfészek. Mivel az IDE esetében korlátozott számú eszközfájl állt rendelkezésre, ezért a SATA. és mass storage (pendrive-okat nevezik így) eszközöket a SCSI alrendszer részeivé tették. így ezek neve szintén /dev/sd{a,b,c,d,e,f,g,h,...}.
A SCSI eszközök esetében nagyon nagy különbséget nem teszünk a lemezek között, egyelőre elégedjünk meg azzal az állítással (mely nem igaz), hogy felcsatolás sorrendjében az eszközök neve: első SCSI winchester neve /dev/sda másodiké /dev/sdb, stb. Ha vegyesen használunk SCSI, SATA, USB pendrive-ot akkor is fogadjuk el a fenti állítást, a szakirányos hallgatók majd bővebben tanulnak róla.
IDE eszközök tekintetében teljesen egyértelmű az eszközök elnevezése. Az elsődleges master eszközt a /dev/hda, az elsődleges slave-t /dev/hdb, a másodlagos mastert /dev/hdc, a másodlagos slavet /dev/hdd nek nevezzük, függetlenül attól, hogy CD-ROM vagy winchester az eszköz.
Egy winchestert mindenképp (ez nem teljesen igaz) particionálunk legalább egy részre. Természetesen állhat több részből is. Mindenképp igaz az (történelmi okok miatt), hogy egy winchestert maximum négy elsődleges és/vagy extended partícióra oszthatunk, melyből az extendedet tovább darabolhatjuk logikai partíciókra.
Amennyiben egy extended partíciót felosztunk egy vagy több logikai partícióra, a partíció „használhatatlanná” válik számunkra.

Ezek ismeretében, az elsődleges, extended partíciók neve egy primary master eszközön:
/dev/hda1
/dev/hda2
/dev/hda3
/dev/hda4

A logikai partíciók neve:

/dev/hda5
/dev/hda6
/dev/hda7
…
Ha csak egy extended partíciónk van, amely fel van osztva logikai partíciókra az első használható partíciónk a /dev/hda5, hiszen azzal, hogy felosztottuk az egyetlen partíciónkat, elvesztettük a „jogot”, hogy használhassuk.
Ha 2 elsődleges vagy extended partíciónk van, és ha az első extended, melyet felosztottuk logikai partíciókra, akkor a használható partíciók: /dev/hda2, /dev/hda5, stb.

Készítette: Sinkó Gergely

Arról, hogy rendszerünkön milyen partíciók vannak, meggyőződhetünk a /proc/partitions kiíratásával:

cat /proc/partitions

Partíciók, CD-ROM-ok, pendrive felcsatolása

A fentiek ismeretében úgy tűnhet, hogy az eszközök olvasásra írásra készen állnak a rendszerünk /dev könyvtárában, ami majdnem igaz is. Ahhoz, hogy „normálisan" használhassuk ezeket az eszközöket, fel kell csatolni (mountolni) a rendszerünk valamelyik (lehetőleg) üres könyvtárába. Erre a mount parancs szolgál. Szintaktikája:

mount -o opciók -t fájlrendszer_típus /dev/milyen_eszközt /hova

Ha egy másodlagos master CD-ROM-ot akarunk felcsatolni a /cdrom könyvtárba a következő parancsot kell kiadnunk:

mount /dev/cdrom /cdrom

Ez abban az esetben működik, ha egy CD-ROM eszköz van a gépünkben és a linux automatikusan létrehozza a symlinket a CD-ROM valódi nevére. A /proc/ide/hd{a,b,c,d}/driver fájl tartalmazza, hogy milyen meghajtóprogramot használ a kernel az eszközhöz. Ezeket a fájlokat kiíratva, ha ide-cdrom-ot kapunk válaszul az adott eszköz CD-ROM.

Ha rendszerünk tartalmaz egy SATA winchestert, és egy pendrive-ot szeretnénk felcsatolni, melyen egy elsődleges partíció található (van olyan pendrive, mely gyárilag nincsen particionálva ebben az esetben mintha CD-ROM-ot csatolnánk fel), a következő paranccsal tehetjük meg:

mount /dev/sdb1 /media/pendrive

A mount parancs opciók és csatolók nélküli futtatásával a már felcsatolt partíciókat írja ki.
Egy felcsatolt partíciót az umount paranccsal tudunk leválasztani a rendszerről

Feladat

1. Lépjen be root felhasználóként az első terminálon!
2. Mountolja fel a gépben található CD-ROM-ot!
   (`mount /dev/hda /cdrom`)
3. Csatlakoztasson egy pendrive-ot a rendszer USB portjára, majd csatolja fel!
   (`mount /dev/sdb1 /mnt`)
4. Válassza le a felcsatolt partíciókatt!
   (`umount /mnt, umount /cdrom`)

A pendrive csatlakoztatása után a rendszer üzenetet küld számunkra. hogy új USB eszközt talált. Ettől nem kell megijedni.

5. Csatolja fel a rendszerben található SATA winchester 3. elsődleges partícióját, a /mnt könyvtárba!
   (`mount /dev/sda3 /mnt`)
6. Válassza le a felcsatolt partíciót!

Készítette: Sinkó Gergely

(`umount /mnt`)

A fentiek után joggal gondolhatnánk, mennyire rossz a Linux, hiszen meg egy rádugott pendriveot sem ismer/csatol fel automatikusan. Ez nem igaz. Vannak olyan terjesztések, ahol az összes partíció automatikusan felcsatolódik. DE! Nem minden terjesztés ilyen, ezért a fentiek elsajátítása fontos, hogy ne „bénuljunk" meg egy ilyen rendszer használatakor sem.

Szimbolikus és „hard" linkek

Néha szükségünk lehet arra, hogy egy fájlra több helyen, esetleg több névvel hivatkozzunk. Ilyenkor megoldás lehet az, hogy a fájlt másolgatjuk és átnevezzük. Ez működő megoldás, de sajnos helypazarlás, valamint, ha minden helyen ugyan azt a módosítást szeretnénk végrehajtani, sok munkával járhat. Erre a célra lett kitalálva a symlink (szimbolikus) és hardlink. Mind a két típusról elmondható, hogy egy „mutatót" készít az eredeti fájlra. Azt, hogy ez pontosan hogy is van, a szakirányos hallgatók a későbbiekben hallgatják Hálózati operációs rendszerek című tárgyból. Most elég, annyit megjegyezni, hogy symlink esetén, más partíciókon lévő fájlokra is hivatkozhatunk, míg hardlink esetében csak az azonos partíción lévő fájlra történő hivatkozás működik.
Mind a symlink mind a hardlink létrehozásához az ln parancsot használjuk, symlink esetében kell egy -s kapcsolót is megadni.

Feladat

1. Hozzon létre egy szöveges fájlt, mely a hardlink szót tartalmazza a /tmp könyvtárba!
   (`echo „hardlink" > /tmp/proba.txt`)
2. Készítsen hardlinket a /tmp/proba.txt állományra! Az új hivatkozó fájl neve legyen hardlink.txt!
   (`ln /tmp/proba.txt /tmp/hardlink.txt`)
3. Írassa ki az állományt, valóban hivatkozik-e a hardlink.txt a proba.txt-re!
   (`cat /tmp/hardlink.txt`)
4. Hozzon létre egy szöveges fájlt, mely a symlink szót tartalmazza a /tmp könyvtárba!
   (`echo „symlink" > /tmp/proba2.txt`)
5. Készítsen symlinket a /tmp/proba2.txt állományra! Az új hivatkozó fájl neve legyen symlink.txt!
   (`ln -s /tmp/proba2.txt /tmp/symlink.txt`)
6. Írassa ki az állományt, valóban hivatkozik-e a symlink.txt a proba2.txt-re!
   (`cat /tmp/symlink.txt`)

Processzek kezelése

Processznek nevezzük a futó programokat. Minden ilyen programot, egy ún. process ID-val (PID) azonosítunk.
A rendszerünkön futó processzeket a ps paranccsal tudjuk listáztatni. A processzek azonosítói (PID-jei), a második oszlopban találhatóak! A ps kapcsolói:

- a: összes processz kiíratása;
- e: ugyan csak az összes processz kiírása, változóértékekkel;
- u: user formátum (bővebb információt ad),
- x: terminálhoz (tty-hez) nem kapcsolható (olyan processzek, melyek háttérben indulnak el) processzeket is listázza.

A processzek használt erőforrás igényük alapján történő listázására a top parancsot használhatjuk (a

top parancsból kilépni q billentyűvel lehet).
Azt, hogy egy erőforrást (portot, partíciót), milyen processzek használnak a fuser paranccsal tudjuk megjeleníteni. A fuser néhány lehetséges kapcsolója:

- -v: megjelenítés,
- -m: mountolt partíció, mountolási helye vagy eszközfájl neve,
- -t tcp|udp portszám: az adott portot használó processzek PIDjét listázza ki;
- -k: a talált (-m vagy -t) processzeket állítja le.

A processzek leállítására, szignálok küldésére a kill, killall parancsok használhatóak. A leggyakoribb kapcsolója a -HUP (hangup) -9 (KILL). A kill PID alapján dolgozik, a killall a processz neve alapján. Ha több azonos nevű processz fut, és root joggal adjuk ki, akkor az ÖSSZES processzt leállítja mely azzal a névvel fut.
Egy felhasználó összes processzének (ehhez persze megfelelő jogosultság kell) leállítására a slay parancs használható.

Feladat

1. Jelentkezzünk be az első és második terminálokon root felhasználóként!
2. Indítsunk el egy processzt, a második terminálon, ami biztosan sokáig fog futni!
   (`find / -exec sleep 1 \;`)
3. Az első terminálon nézze meg a find foglal-e nagy erőforrást!
   (`top`)
4. Az első terminálon listázzuk az éppen futó processzeket!
   (`ps aux`)
5. A ps parancs kimenetének, az első oszlopa a felhasználó mely futtatja a parancsot, a második a PID mely azonosítja a futó processzt. Erre a számra van most nekünk szükségünk. Keresse meg a find-hoz tartozó PID-et!
6. Állítsa le kill paranccsal a find programot!
   (`kill -9 PID_amit_az_előbb_megnéztem`)
7. Ha mindent jól csinált a második terminálon leált a find. Lépjen át és győződjön meg róla! Ha leállt indítsa el újra!
8. Nézze meg az első terminálon, hogy fut-e a find!
   (`ps aux`)
9. Ha igen, akkor állítsa le a killall parancs segítségével!
   (`killall -9 find`)
10. A harmadik terminálon lépjen be diak felhasználóként!
11. Indítsa el a második terminálon a find parancsot újra!
12. Váltson át a harmadik terminálra, és próbálja ,meg leállítani a find programot, a fenti módszerek valamelyikével!
13. Azt kell tapasztalnia, hogy nem sikerül leállítani, ennek oka a megfelelő jogosultságok hiánya.
14. Nézze meg, hogy a felcsatolt „/” partíciót milyen processzek használják!
    (`fuser -vm /`)

Készítette: Sinkó Gergely