Hadarics Kálmán

A Debian GNU/Linux, mint

Hálózati operációs rendszer

2009

Tartalomjegyzék

1. Jogi nyilatkozat, támogatás

Hadarics Kálmán, mint szerző nyilatkozom, hogy ez a mű a saját tapasztalatom és munkám eredménye, a benne foglaltak senkinek a szerzői jogait nem sértik.

A jegyzettel kapcsolatban mindenféle építő jellegű kritikát, észrevételt, szívesen fogadok a kami@mail.duf.hu e-mail címen.

Amennyiben új gondolod, hogy ez az ismeretanyag, amit ezen műben megkaptál értékes, kérlek a lehetőségeidhez mérten támogass. Az így befolyt összeget ezen ismeretanyag bővítésére, korszerűsítésére fogom fordítani.

Amennyiben támogatni szeretnél, akkor látogass el a http://kami.duf.hu/debian_jegyzet weboldalra. Itt lesznek információk arról, hogy a támogatásodat hogyan juttathatod el hozzám. A támogatással kapcsolatban szintén érdeklődhetsz a kami@mail.duf.hu email címen. Az előző webcímen a jegyzet aktualizált kiadásai szintén elérhetőek lesznek.

Előre köszönöm mindenkinek a támogatását, jó kedvet az olvasáshoz, sok sikert az életben, és „kössön mindenki jó barátságot” a Debian GNU/Linux operációs rendszerrel.

Dunaújváros, 2010.10.13.

Hadarics Kálmán
kami@mail.duf.hu

2. Bevezetés

A Debian GNU/Linux operációs rendszer az egyik legrégebbi Linux terjesztés. Nagyon sok helyen előszeretettel használják hálózati szolgáltatások, hálózat menedzselési feladatok ellátására. Ezen mű elsősorban olyan informatikus hallgatóknak és olyan személyeknek szól, akik már rendelkeznek tapasztalatokkal, használtak már „desktop" operációs rendszerként Linuxot. A jegyzetben gyakorlati, rendszeradminisztrátori szempontból igyekszem bemutatni a Debian GNU/Linuxot és néhány elterjedt hálózati szolgáltatását. Feltételezem, hogy az olvasó rendelkezik általános ismeretekkel a számítógép hálózatokról és a TCP/IP alapú kommunikációról.
Mivel a Linux a létét a számítógép hálózatoknak köszönheti, ezért ezt úgy „igyekszik meghálálni", hogy mindenféle hálózati feladathoz készülnek alkalmazások. Ezen alkalmazások száma akkora, hogy bemutatni mindet és ezeknek a lehetőségeit reménytelen vállalkozás lenne.
A jegyzet felépítését tekintve több egymástól részben függetleníthető részt tartalmaz. Az első részben a Debian GNU/Linux verziót, irányelveit mutatom be. Ez után következik telepítésének folyamatának az áttekintése és a csomagkezelő struktúrájának és parancsainak a bemutatása. Egy rendszer megbízható működésének és rugalmasságának alappillére a jól megválasztott diszk kezelési módszer. Fontos az adatok redundáns tárolása, az adatok elérésének sebessége és az egyes partíciók átméretezhetősége. Ebben a fejezetben a szoftveres RAID és a LVM lehetőségeit és parancsait mutatom be.
A Linux operációs rendszer magjának, a kernelnek az ismerete is nagyon fontos a rendszer működése szempontjából. A jegyzetben bemutatom a kernelfordítás folyamatát és főbb konfigurációs opciókhoz magyarázatokat fűzök. A fejezet átolvasása után képesnek kell lenni egy gép esetében működő kernel fordítására. A telepítés és a kernelfordítás után röviden ismertetem a PAM lehetőségeit és a felhasználók adminisztrációját. A továbbiakban pedig a TCP/IP alapú hálózat beállításának bemutatásával folytatom. A jegyzetben néhány hálózati szolgáltatást emeltem ki, amelyeket részletesen bemutatok a konfigurálással együtt. A szolgáltatások választásánál figyelembe vettem az egyes programok népszerűségét és természetesen a személyes tapasztalataimat is.
Igyekszem az alkalmazásokat a gyakorlati oldalról megközelíteni és a biztonsági szempontból fontos aspektusokat külön hangsúlyozni. Az alkalmazások esetében általában nem a telepítés a legizgalmasabb kérdés. Nagyon sok esetben csomagból az egyes alkalmazások telepíthetőek. Sokkal érdekesebb kérdés, hogy az alkalmazások hogyan tudnak együttműködni, milyen lehetőségek vannak arra, hogy együttesen ellássák egy informatikai rendszer feladatait.
A jegyzet tartalma alapján, remélem mindenki képes lesz a hálózat és a hálózati szolgáltatások telepítésére Debian GNU/Linux operációs rendszerben. Képesek leszünk Linux alapú rendszerek telepítésére, menedzselésére, integrációjára. Ezen a területen is érvényes a névszerverek esetében használt elv, hogyha nem tudok valamit, akkor megkérdezem attól, aki tudja vagy csinált már ilyet. Fontos dolog, hogy ha valami nem működik elsőre, próbáljunk meg a hiba okára rájönni (sok esetben a Google segíthet) és megtalálni rá a megfelelő megoldást. A logikus, következetes gondolkodást elsajátítva, problémamegoldó képességünket továbbfejlesztve képesek leszünk a Linux tetszőleges módú használatára.

3. A Debian GNU/Linux

3.1. A disztribúció eredete és jellemzői

A Debian a legkorábbi Linux disztribúciók egyike. 1993 augusztusában Ian Murdock indította útjára a GNU szellemében. A projekt nevét a feleség (Debora) neve után kapta. A „Deb” és a „Ian” összekapcsolásából született meg a Debian név.
A disztribúció jelenleg a legnagyobb nem kereskedelmi terjesztések közé tartozik.
Legfontosabb jellemzői:

- teljesen ingyenes és saját szabad szoftver irányelvekkel rendelkezik
- számtalan önkéntes fejlesztő tartja karban
- fejlett csomagkezelő rendszerrel rendelkezik és sok bináris formában elérhető csomaggal (kb. 22600 az 5.0 esetében)
- a felhasználókkal való viszonyt a „Debian társadalmi szerződés” írja

3.2. A Debian szabad szoftverekre vonatkozó irányelvei (Debian Free Software Guidelines – DFSG)

1. Szabad terjesztés
   A Debian összetevőinek licence senkit sem korlátozhat abban, hogy a szoftvert különböző forrásokból származó programokból felépített szoftver-disztribúciók összetevőjeként eladja vagy továbbadja. Az ilyen jellegű eladásokra vonatkozóan a licenc semmiféle szabadalmi vagy egyéb díj megfizetését nem követelheti meg.
2. Forráskód
   A programnak tartalmaznia kell a forráskódot, és a lefordított változat mellett engedélyeznie kell a forráskód terjesztését is.
3. Leszármazott munkák
   A licencnek lehetővé kell tennie a módosításokat és leszármazott munkák készítését, és engedélyeznie ezek terjesztését az eredeti szoftver licencével megegyező licenc hatálya alatt.
4. A szerző forráskódjának sértetlensége
   A licenc korlátozhatja a forráskód módosított formáinak terjesztését, de csak abban az esetben, ha emellett lehetővé teszi „patch fájlok” együttes terjesztését a forráskóddal, amelynek segítségével a program módosítása elvégezhető a fordítás során. A licencnek kifejezetten engedélyeznie kell a módosított forrásból összeállított szoftver terjesztését. A licenc megkövetelheti, hogy a leszármazott munkák neve vagy verziószáma az eredeti szoftverétől eltérjen. (Ez egy kompromisszum. A Debian csoport arra buzdít minden szerzőt, hogy ne korlátozzák se a forrás-, se a bináris fájlok módosítását.)
5. Mentesség a személyekkel vagy csoportokkal szembeni diszkriminációtól
   A licenc semmilyen személlyel vagy csoporttal szemben nem alkalmazhat megkülönböztetést.
6. Mentesség a felhasználási területekkel szembeni diszkriminációtól
   A licenc senkit nem korlátozhat abban, hogy a programot egy adott felhasználási területen alkalmazza. Például nem korlátozhatja egy adott

program üzleti vagy génkutatásban való felhasználását.

7. Licenc terjesztése
   A programra vonatkozó jogoknak úgy kell vonatkoznia mindenkire, akik hozzájutnak a programhoz, hogy ne legyen szükség további licenc elfogadására.
8. A licenc nem lehet a Debianra jellemző
   A programra vonatkozó jogok nem függhetnek attól, hogy a program a Debian része vagy sem. Ha a programot a Debian rendszertől elkülönítve használják fel vagy terjesztik, de ez a programlicencnek megfelelő módon történik, akkor a programhoz hozzájutók mindegyikének azonos jogokkal kell rendelkeznie azokhoz képest, akik a Debian rendszerrel együtt jutottak a programhoz.
9. A licenc nem érinthet más szoftvereket
   A licenc nem tartalmazhat a licenc szoftverrel együtt szállított szoftverekre vonatkozó korlátozásokat. Például a licenc nem határozhatja meg, hogy a vele azonos adathordozón található programok mindegyikének szabad szoftvernek kell lennie.
10. Példalicencek
    A „GPL”, a „BSD” és az „Artistic” licenceket „szabad” licencnek tekintjük.

3.3. „Társadalmi szerződés” a Szabad szoftver közösséggel

1. A Debian 100%-ig szabad szoftver marad
   Megígérjük, hogy a Debian GNU/Linux disztribúciót megtartjuk teljes egészében szabad szoftvernek. Mivel a szabad szoftver kifejezésnek többféle értelmezése is van, a későbbiekben azt is felvázoljuk, hogy milyen irányelvek segítségével döntjük el a szoftverekről, hogy „szabad” szoftverek-e. Támogatjuk azokat a felhasználókat, akik Debian disztribúción nem szabad szoftvereket fejlesztenek vagy használnak, de a rendszer sohasem fog nem szabad szoftvertől függeni.
2. Mindent visszajuttatunk a Szabad szoftver közösségnek
   Ha új összetevőket írunk a Debian rendszerhez, akkor ezeket szabad licenc hatálya alá helyezzük. Mindig a lehető legjobb rendszer létrehozására törekszünk, ezért szabad szoftverek széles körben elterjedhetnek. A hibajavításokat, továbbfejlesztéseket és felhasználói kéréseket eljuttatjuk a rendszerünkben található szoftverek „illetékes” szerzőinek.
3. Nem rejtjük el a problémákat
   A teljes hibajelentési adatbázisunk folyamatosan elérhető lesz a nyilvánosság számára. A felhasználók által elektronikusan leadott jelentések azonnal láthatóvá válnak másoknak.
4. A prioritást a felhasználóink és a szabad szoftverek jelentik
   A lépéseinket a felhasználóink és a szabad szoftverek közössége fogja irányítani, az ő érdekeiket helyezzük előtérbe. Támogatjuk a felhasználóknak a többféle számítástechnikai környezetben való működéssel kapcsolatos igényeit. Nem akadályozzuk meg, hogy kereskedelmi szoftverek készüljenek a Debian rendszerekre, és azt is engedélyezzük másoknak, hogy a Debianból és további kereskedelmi szoftverekből érték növelt disztribúciókat hozzanak létre anélkül, hogy ezért fizetniük kelljen. Ezen célok elérése érdekében 100%-ig szabad, kiváló minőségű szoftverekből integrált rendszert nyújtunk bármely olyan jogi korlátozás nélkül, amely

megakadályozná a disztribúció ilyen jellegű felhasználását.

5. A szabad szoftverekre vonatkozó szabványainknak nem megfelelő programok

   Tudomásul vesszük, hogy bizonyos felhasználóinknak olyan programokra van szükségük, amelyek nem felelnek meg a Debian szabad szoftverekre vonatkozó irányvonalaknak. Az ilyen szoftverek számára hoztuk létre az FTP archívumok „contrib” és „non-free” területeit. Ezen könyvtárakban található szoftverek nem részei a Debian rendszernek, bár be vannak állítva a Debian alatti használathoz. A CD-gyártóknak ajánljuk az itt található szoftvercsomagok licenceinek áttekintését annak meghatározásához, hogy terjeszthetik-e az adott szoftvert az általuk forgalomba hozott CD-ken. Ennek megfelelően, bár a nem szabad szoftverek nem képezik a Debian részét, támogatjuk használatukat, és a nem szabad szoftvercsomagok számára is biztosítunk infrastrukturális hátteret (például a hibakövetési rendszert és különféle levelezési listákat).

3.4. Debian - Az univerzális operációs rendszer

A Debian projektben résztvevő személyek célja hogy egy szabadon használható operációs rendszert hozzanak létre. A legtöbb alapvető alkalmazás a GNU projektből származik. A Debian jelenleg a Linux kernel köré épül. Éppen ezért, amikor az operációs rendszerre hivatkozunk, helyesebb, hogyha Debian GNU/Linuxnak mondjuk. A Debian fejlesztés azonban nem korlátozódik kizárólag a Linux kernel köré. Vannak például a Hurd mikrokernelre (Debian GNU/Hurd), vagy a NetBSD kernelre (Debian GNU/NetBSD) alapuló fejlesztések is. Amikor Debiant mondunk, akkor általában a Debian GNU/Linuxra gondolunk.

3.5. A Debian GNU/Linux verziói

A Debian verziókat a verziószám mellett egy kódnévvel (fantázianévvel) is jellemezhetjük (az 1.0 kivételével). Ezek a kód nevek a Toy Story című animációs filmből erednek.

A Debian GNU/Linux verziói és kód neveik: 1. sz. táblázat

Verzió	***Megjelenés ideje***	***Kódnév***
1.0	1995	-
1.1	1996 június	Buzz
1.2	1996 december	Rex
1.3	1997 június	Bo
2.0	1998 július	Hamm
2.1	1999 március	Slink
2.2	2000 augusztus	Potato
3.0	2002 július	Woody
3.1	2005 június	Sarge
4.0	2007 április	Etch
5.0	2009 február	Lenny
6.0	?	Squeeze

Verzió	***Megjelenés ideje***	***Kódnév***
-	-	Sid

A Debian a terjesztések esetében a következő elnevezéseket is használja:

- **stable**: a mindenkori stabil változat (jelenleg a „Lenny”), amely használata ajánlott. A kiadást egy hosszú tesztelési fázis előzi meg. Nem minden esetben tartalmazza a legújabb programokat.
- **testing**: a tesztelés alatt lévő változat (jelenleg a „Squeeze”). Ide kerülnek azok az alkalmazások, amelyek a következő stabil disztribúcióban benne lesznek.
- **unstable**: a legfrissebb programokat tartalmazza. („Sid” a kódneve, amely nem változik.) Egyáltalán nem garantált, hogy az alkalmazás működni is fog. Tapasztalt felhasználóknak ajánlott.
- **oldstable**: amikor megjelenik egy új stabil kiadás, akkor a korábbi stabil kiadást jelenti

Ezek az elnevezések a gyakorlatban szimbolikus linkeket jelentenek. A linkek pedig a kódnévvel megegyező nevű könyvtárra mutatnak. A verzió változás (új, stabil megjelenése) csak szimbolikus link állítást követel meg.

Van arra is lehetőség, hogy ún. kevert Debian disztribúciót használjunk. Ez annyit jelenthet például, hogy bizonyos csomagokat a stabil verzióból, míg másokat a testing, vagy az unstable verzióból használunk. Ebből elég sok függőségi probléma adódhat, hiszen általában az egyes kiadások különböző GLIBC verziókra épülnek.

4. A Debian GNU/Linux telepítési folyamata

4.1. Előkészület a telepítésre

Az operációs rendszer telepítés nagy hatással lehet a számítógép további állapotára, használhatóságára. Általános szabályként elmondható, hogy mielőtt nekilátnánk a telepítésnek, mentsük le az összes fontos adatunkat a diszkjeinkről. Abban az esetben, hogyha valaki nem ismeri megfelelő mélységben a telepítő programot, véletlenül is hibázhat. Ekkor pedig akár több éves munkája is elveszhet.

4.1.1. A telepítéshez szükséges információk

A telepítés elkezdése előtt a hardverrel és a hálózati beállításokkal tisztában kell lennünk. Továbbá a telepítendő rendszerrel kapcsolatban néhány dolgot tisztáznunk kell, meg kell terveznünk. Ezek közül a legfontosabbak:

- **A hardver eszközök típusai, lehetőségeik.** Mielőtt a telepítésnek nekilátunk, tudnunk kell a hardver eszközök típusait, illetve ezek paramétereit. Ajánlott megnézni, hogy az adott eszközök Linux kompatibilisek-e. Ezt a lépést a hardver eszközök megvásárlása előtt a legjobb eldönteni.
- **A hálózati paraméterek.** A hálózati rendszergazdától kérhetünk a Linux szerver számára IP címet, illetve tudhatjuk meg tőle a hálózat további paramétereit. A továbbiakban feltételezem, hogy TCP/IP alapú hálózatot használunk, illetve IPv4-et.
- **Az új rendszer felhasználási területe, a rendszer működése szempontjából szükséges szolgáltatások.** Ez befolyásolja például a lemezek particionálását, az alkalmazandó elveket a redundáns adattárolásra vonatkozóan.
- **A fájlrendszerek képességei.** Vannak olyan feladatok, amely esetben nem elhanyagolható szempont a fájlrendszer választás sem. Ez azokra az esetekre igaz elsősorban, ahol a rendszer I/O terhelése kifejezetten nagy lesz, és jelentős teljesítménykülönbség adódhat a fájlrendszer típus függvényében.
- **Biztonsági, üzemeltetési megfontolások.** Már a telepítés előtt evidens, hogy a rendszert a későbbiekben üzemeltetni is kell. Éppen ezért nem árt megtervezni a rendszeres és időszakos mentések módját és módszereit. Kellő tervezéssel egyszerűsíthetjük és hatékonnyá tehetjük a biztonsági mentéseket.
- Az előző szempontok figyelembevételével határoznunk kell a lemezek particionálására vonatkozóan. Egy sémát kell definiálnunk, amelyben rögzítjük az egyes partíciók sorrendjét, méretét, és a használni kívánt fájlrendszert.

Minta a particionálásra: 2. sz. táblázat

Partíció	Típus	Méret	Csatlakoztatási pont
/dev/sda1	ext3	100MB	/boot
/dev/sda2	swap	500MB	-
/dev/sda5	RAID	2500MB	-
/dev/sda6	xfs	fennmaradó	/tmp
/dev/sdb1	RAID	2500MB	-
/dev/sdb2	xfs	fennmaradó	/var
/dev/sda5,/dev/sdb1 **RAID1**	xfs	2500 MB	/

Abban az esetben, hogyha a hardver eszközt több architektúra is támogatja, fontos döntés, hogy melyiket használjuk. Jelenleg ilyenre lehetőségünk van a PC-k esetében. Választhatjuk a hagyományos x86 (i386) 32 bites architektúrát, de az újabb processzorokon működik az x86_64 (amd64) 64 bites is. Vigyázzunk arra, hogy ne keverjük össze az ia64 (64 bites Itanium) és az x86_64 architektúrát. Azok az újabb Intel processzorok (pl.: Core2, Quad Core, ...), amelyek támogatják a 64 bites üzemmódot (EMT64), azok esetében csak az x86_64 architektúrát használhatjuk, az ia64-et nem. A Debian telepítő CD-k letöltésekor pedig az amd64-et kell választanunk. A Debian 5.0 esetében létezik 64 bites operációs rendszer is. A legtöbb 32 bites alkalmazás elérhető ez alatt is. Azokkal a programokkal lehet gond, amelyeknek nem hozzáférhető a forráskódja, és csak bináris formában érhetők el. Ilyen például az Acrobat Reader vagy a Adobe Flash, amelyeknek csak 32 bites verziói léteznek. Jelenleg nem érhető el stabil 64 bites Java plugin a Mozilla/Firefox/Iceweasel böngészők alá.

4.1.2. A telepítés célja

A telepítés megkezdése előtt fontos dolog, hogy tisztában legyünk az egész folyamat céljával. Ha egyszerűn akarjuk megfogalmazni, akkor mondhatjuk azt, hogy az előzetes tervben meghatározott séma szerint kell egy Linux operációs rendszernek megjelennie a cél számítógépen. Ez annyit jelent, hogy valamilyen módszerrel a cél számítógépen létre kell hozni a szükséges fájlrendszereket, és adott médiumról (pl.: CD, DVD, hálózat, másik HDD, floppy?, USB pendrive, …) kell fájlokat átmásolni, kicsomagolni és a szükséges beállításokat megtenni. Miután ez bekövetkezett, képessé kell tenni az új rendszert, hogy „külső segítség nélkül” képes legyen elindulni és a továbbiakban működni.

4.1.3. A Debian GNU/Linux 5.0 által támogatott architektúrák

Jelenleg többféle architektúrára vonatkozóan elérhető és telepíthető a Debian GNU/Linux CD segítségével.

Architektúrák, amelyekre létezik telepítő CD: 3. sz. táblázat

Debian architektúra	***Számítógép (CPU) típus***
alpha	Compaq (HP) Alpha AXP
arm	ARM (beágyazott rendszerek)
amd64	64 bites AMD, Intel EM64T
i386	Intel x86 32 bit (IA-32)
ia64	Intel 64 bit (Itanium, IA-64)
hppa	HP PA-RISC
mips	SGI (big-endian)
mipsel	Digital DECstations and Cobalt Qube/RaQ (little-endian)
powerpc	Power Macintosh, IBM pSeries
s390	IBM S/390
sparc	SUN Sparc, UltraSparc

A továbbiakban csak az IA-32 (i386, x86) architektúrára való telepítést fogom bemutatni, a többi architektúrához tartozó telepítő lehetőségeitől, beállításaitól eltekintek.

4.2. A telepítés módszerei

A telepítés elvégzésére tehát többféle módszer létezik. A továbbiakban csak azokat a módszereket ismertetem, amelyek nem járnak az adott gép „szétszedésével”. Ezen módszerek közül a legfontosabbak:

- telepítő CD/DVD segítségével
- telepítő indítása külső USB eszközről
- telepítő indítása hálózatról
- telepítő indítása floppy segítségével (manapság már nem jellemző)

Amennyiben az adott számítógépen már telepítve van egy Linux operációs rendszer, akkor megoldható az is, hogy ennek a felhasználásával telepítünk egy másik lemezre vagy partícióra egy rendszert. A „debootstrap” parancs segítségével egy elérhető Debian tükörszerverről telepíthető az operációs rendszer valamilyen csatlakoztatott könyvtár alá.
Általában tehát a telepítő programot hívjuk segítségül az operációs rendszer telepítéséhez. A Debian GNU/Linux esetében ekkor a következő történik:

- az előzőekben ismertetett módok valamelyikén elindul egy kernel és egy „minimális” Linux operációs rendszer.
- a telepítő programot az elindult operációs rendszer futtatja, amely segítségével a telepítési feladat kivitelezhető

A telepítővel kapcsolatban fontos megemlíteni, hogy nem biztos, hogy a kernel összes lehetőségét támogatja. Ez elsősorban a még kísérleti fázisban lévő támogatásokra vonatkozik.
A telepítő segítségével helyi illetve hálózati csomagokat (package) is elérhetünk. Így akár a legújabb elérhető csomagokat telepíthetjük fel, amelyek korábbi biztonsági hibákat már nem tartalmaznak. A Debian 5.0 esetében minden

csomag el van látva digitális aláírással. A magasabb szintű csomagkezelő alkalmazásokkal (pl.: apt-get, dselect, aptitude, …) csak olyan csomagok telepíthetők, amelyek a fejlesztők hivatalos aláírásával rendelkeznek.

4.2.1. A telepítő CD/DVD-k fajtái

A Debian telepítő CD-k, DVD-k hivatalos helye a http://cdimage.debian.org. Ajánlott azonban valamelyik tükörszervert használni a letöltéshez. A magyar tükörszerver az ftp://ftp.hu.debian.org. Innét különböző formátumokban tölthetjük le a telepítőt:

- ISO képfájlok (iso9660 image) formájában.
  A teljes disztribúció az 5.0 i386 esetében 5 db ISO DVD vagy 31 db ISO CD formájában érhető el. Ezek ftp vagy http segítségével letölthetők.
- Bittorrent fájlok is elérhetők, vagyis a fájlcserélő protokoll segítségével is megoldható az ISO fájlok letöltése.
- Jigdo segítségével is beszerezhetők a telepítő CD-k. A jigdo egy alkalmazás, amely képes több különböző tükörhelyről tölteni a CD egyes részeit, majd pedig összeállítani az ISO fájlt.

Fontos dolog, hogy amennyiben a telepítendő gép rendelkezik hálózati kapcsolattal, akkor nem kell feltétlenül az összes CD-t vagy DVD-t letölteni. Az elsőn rajta vannak azok a csomagok, amelyek minden esetben szükségesek. A továbbiakat pedig telepíthetjük hálózat felhasználásával.
Miután az ISO fájlokat letöltöttük, ajánlott az épségük ellenőrzése. Ezt úgy tehetjük meg, hogy elsőként letöltjük a hozzájuk kapcsolódó MD5 vagy SHA1 ellenőrző összeget tartalmazó fájlokat. Ezek után pedig az `md5sum` vagy a `sha1sum` parancs segítségével (-c kapcsoló) ellenőrizzük, hogy ezek igazak-e a letöltött fájlokra.
A következő ISO fájlok érhetők el (pl: i386 5.0 release 3 esetében):

- debian-503-i386-CD-1.iso: telepítő Gnome grafikus környezettel (full CD)
- debian-503-i386-kde-CD-1.iso: telepítő KDE grafikus környezettel
- debian-503-i386-xfce+lxde-CD-1.iso: telepítő XFCE és LXDE grafikus környezettel
- debian-503-i386-netinst.iso (kb. 150MB): csak egy alaprendszert tartalmaz, és a hálózati telepítéshez szükséges csomagokat
- debian-503-i386-businesscard.iso (kb. 36MB): az alaprendszert is hálózatról tölti le

4.3. A telepítő program indítása

A Debian GNU/Linux 5.0 telepítője kapott egy új induló képernyőt (splash screen) és boot menüt. Itt egyszerűen, menüből módon választhatunk a telepítési módok között. Telepíthetünk karakteres és grafikus felület használatával, és az 'Advanced options' menüpont alatt egyéb módokon is indíthatjuk a telepítést. A 'Help' menüpont kiválasztása után megkapjuk a korábbi verziókból jól ismert telepítési felületet, és leírást. Itt az F1-F10 gombok segítségével megtekinthetünk egy rövid súgót a telepítő használatáról. A legfontosabb információkat a harmadik oldal (F3) tartalmazza. Itt kerül ugyanis leírásra, hogy a telepítő milyen paraméterek segítségével indítható.
A Debian 5.0 telepítőjének indítási lehetőségei:

- **install**: A „alapértelmezett” (egyszerű) telepítési mód karakteres felületen. A telepítő program sok esetben alapértelmezett beállításokkal dolgozik.

Például a hálózat konfigurálása esetében feltételezi, hogy van DHCP szerver a hálózatban, és ennek a segítségével próbálja a hálózati kártyát konfigurálni.

- **installgui**: Az előzővel megegyező telepítési mód, csak nem karakteres, hanem grafikus felületen.
- **expert**: Szakértőknek való telepítési mód. Ebben az esetben a telepítés folyamatába teljes mértékben beavatkozhatunk. A telepítő sokkal bőbeszédűbb, mint az alapértelmezett mód esetében. Itt adódik arra lehetőség, hogy megválasszuk a betöltendő kernel modulokat, és a telepítő opcionális bővítményeit felhasználjuk.
- **expertgui**: Az előzővel megegyező telepítési mód, csak nem karakteres, hanem grafikus felületen.
- **rescue**: Ez a mód nem telepítésre, hanem egy korábban feltelepített rendszer elérésére szolgál. Ennek a felhasználásával elérhetjük az adott partíción lévő fájljainkat, és készíthetünk róluk másolatot. Illetve amennyiben a rendszerünk valamilyen oknál fogva nem képes elindulni, akkor a problémát orvosolhatjuk.
- **rescuegui**: az előzővel megegyező a feladata, csak nem karakteres, hanem grafikus felületen.

Bármelyik menüpont esetében egy kernel indul el. Amennyiben szeretnénk a kernelnek parancssorban paramétereket átadni, akkor ezt egyszerűen az indítási mód után begépelhetjük.
pl.: `install acpi=off`

A telepítőnek szóló speciális boot paraméterek átadására is van lehetőségünk. Ilyen például:
`netcfg/disable_dhcp=true`, amely egyszerű módban letiltja a DHCP használatát
`bootkbd=hu`, magyar billentyűzet kiosztással indítja el a telepítőt az angol helyett.

4.4. A telepítés lépései

- Nyelv, billentyűzet kiosztás, helyi beállítások (locales) definiálása
- CD-ROM érzékelése és csatolása
- Telepítő összetevők betöltése CD-ről
- Hálózati hardver felderítése
- Hálózat konfigurálása
- Lemezek észlelése
- Lemezek particionálása
- Időzóna és óra beállítása
- Felhasználók és jelszavaik felvétele
- Alaprendszer (Base System) telepítése
- A csomagkezelő beállítása
- Szoftverválasztás és telepítés
- A GRUB (LILO) rendszerbetöltő lemezre telepítése

„Expert” módban könnyen megfigyelhetjük a telepítési lépéseket. Az egyszerű mód ugyanis bizonyos lépéseket nem mutat, vagy összevon más lépésekkel. Az

előző listában csak azokat a lépéseket soroltam fel, amelyeket rendszerint használunk. Vannak azonban ezeken túl is lehetőségek, de ezeket általában speciális szituációkban (pl.: ellenőrzés, hibakeresés, opcionális lehetőségek) használjuk.

4.5. A telepítő CD felépítése

A továbbiakban a debian-503-i386-CD-1.iso fájl felépítését fogom bemutatni. A CD gyökerében a következő fontosabb könyvtárakat és fájlokat találhatjuk:

Könyvtárak:

- **.disk** lemezinformációk. pl.: verziószám, CD típus, elérhető telepítő komponensek (udeb fájlok)
- **dists/lenny** a lemezen található Debian verzió külön alkönyvtárban
- **doc** telepítési kézikönyv és FAQ dokumentumok különböző nyelveken, az aktuális kiadás változásai, képességei (release notes)
- **install.386** a telepítő indulásához szükséges kernel és initrd image
- **isolinux** a telepítő főmenüje, a telepítő esetében használt rendszerbetöltő (isolinux) beállításai
- **pics** png formátumban a Debian logo és a telepítő által használt egyéb képek
- **pool** a CD-n elérhető csomagok rendszerezve
- **tools** régebbi, főként DOS-os eszközök a Linux indításához

Fájlok:

- README.{txt,html} rövid leírás a disztribúcióról
- autorun.{inf,bat} „más” operációs rendszerekből a leírás automatikus indítása
- md5sum.txt a CD-n található fájlokra vonatkozóan MD5 ellenőrző összeg

Szimbolikus link:

- debian nevű szimbolikus link, amely a .-ra mutat

4.6. A Debian GNU/Linux csomagkezelése

4.6.1. A csomagok fajtái, a csomagnév felépítése

A Debian saját csomagformátumot használt a disztribúcióhoz tartozó csomagok esetében. Ezt hívjuk DEB csomagnak (.deb), amely a Unix „ar” fájlformátumát követi. Egy csomag nevének felépítése a következő szabályokat követi:

<package>_<versionnumber>-<revisionnumber>_<arch>.deb

Az előző sor esetében az egyes részek jelentése a következő:

- <package> a csomag rövid neve. Amikor egy szoftver része lesz a Debian-nak, akkor kap egy egyedi nevet. Ez általában a későbbiekben sem változik meg.
- <versionnumber> a csomagban lévő szoftver verziószáma, amely alapján a csomag készült

- <revisionnumber> a csomag módosításakor, javításakor az értéke növekszik. Nemcsak számot, hanem karakterláncot is tartalmazhat, amely a Debian verzióra is utal egyben.
- <arch> az architektúra, amely esetében a csomag telepíthető

pl.: bash_3.2-4_i386.deb vagy php5-xsl_5.2.6.dfsg.1-1+lenny4_i386.deb

Az <arch> mező esetében speciális értéket jelöl az „all”. A csomagok között vannak olyanok, amelyek nem különböznek architektúránként. Ezeket hívhatjuk architektúra független csomagoknak. Tehát ezek olyan fájlokat tartalmaznak (tipikusan beállítások, dokumentációk), amelyek esetében teljesen felesleges lenne az architektúránkénti tárolás és megkülönböztetés.

A Debian csomagok kapcsán beszélhetünk a következő fájlokról, formátumokról:
- **.deb** bináris csomag (lehet architektúra független)
- **.dsc** forrás csomag (**d**ebian **s**our**c**e): a csomagkészítés irányítására használatos
- **.orig.tar.gz**: egy program adott verziójának eredeti forráskódja tar.gz formátumban
- **.diff.gz**: a fejlesztők által végrehajtott módosítások gzippel tömörítve diff fájlként

Lehetséges az, hogy egy forráskódból ne csak egy, hanem több bináris csomag jöjjön létre. Ez leggyakrabban olyankor szokott előfordulni, amikor több csomag is ugyanarra a szoftverre épül, annak adott részét felhasználja. Ebben az esetben például a szoftver egy részét belerakják egy shared library-be. Majd a többi csomagot ettől teszik függővé.

Amennyiben egy DEB csomagot kicsomagolunk, a következő részeit különböztethetjük meg:
- csomag fejrész: a csomagra vonatkozó vezérlő és egyéb információk, parancsfájlok (DEBIAN)
- csomag tartalom: a csomagban lévő fájlok abszolút elérési útvonallal (CONTENT)

A csomag tehát annyival több, mint egy archívum, hogy van arra lehetőség, hogy parancsfájlok segítségével akciókat hajtsunk végre. Ennek a segítségével kivitelezhető, hogy amikor például a csomagot telepítjük, akkor rögtön a rendszerhez hozzá is igazítjuk. Így az új program, vagy modul rögtön a telepítést követően használhatóvá válik.
Egy csomag esetében a fejrészben a következő parancsfájlok lehetségesek:
- preinst telepítés (kicsomagolás) előtt
- postinst telepítés (kicsomagolás) után
- prerm törlés előtt
- postrm törlés után

A parancsfájlok mellett minden egyes csomag fejrészében találhatunk egy 'md5sums' nevű fájlt, amely az MD5 ellenőrző összegeket tartalmazza. Emellett talán a legfontosabb fájl, a 'control' nevet viseli (A továbbiakban Debian Control File). Ebben találhatjuk a csomag vezérlő információit.

Példa egy Debian Control File-ra:

```
Package: mc
Version: 2:4.6.2~git20080311-4
Architecture: i386
Maintainer: Debian MC Packaging Group <pkg-mc-
devel@lists.alioth.debian.org>
Installed-Size: 6252
Depends: libc6 (>= 2.7-1), libglib2.0-0 (>= 2.16.0), libgpm2 (>= 1.20.4),
libslang2 (>= 2.0.7-1)
Suggests: mime-support, perl, zip, unzip, bzip2, links | w3m | lynx, arj,
file, xpdf, dbview, odt2txt
Section: utils
Priority: optional
Homepage: http://www.ibiblio.org/mc/
Description: midnight commander - a powerful file manager
 GNU Midnight Commander is a text-mode full-screen file manager. It
 uses a two panel interface and a subshell for command execution. It
 includes an internal editor with syntax highlighting and an internal
 viewer with support for binary files. Also included is Virtual
 Filesystem (VFS), that allows files on remote systems (e.g. FTP, SSH,
 SMB servers) and files inside archives to be manipulated like real files.
```

Az egyes mezők jelentése a következő:
Package: a csomag rövid neve
Version: a csomagban lévő program verziószáma és revíziószáma
Architecture: a csomag architektúrája
Maintainer: a karbantartó neve és e-mail címe
Installed-Size: telepítést követően a csomag tartalmának mérete kBájtban
Depends: a csomag függőségei (előfeltételei)
Suggests: a csomag „javaslatai” más csomagok telepítésére
Conflicts: ütközés más (korábbi) csomagokkal
Replaces: a csomag helyettesít más csomagokat
Section: a csomag besorolása (szekciója)
Priority: a csomag fontossága
Homepage: a program weboldala
Description: a csomag rövid, célratörő leírása

4.6.2. A csomagok egymáshoz való viszonya, állapotaik

Az előző alfejezetben bemutattam a DEB csomagok legfontosabb jellemzőit, csomagok elnevezéséhez kapcsolódó szabályokat, konvenciókat. Természetesen a csomagokra, mint egy nagy rendszer apró elemeire kell tekintenünk. Ezek az elemek azonban nem függetleníthetők egymástól, hanem szoros kapcsolatok lehetnek köztük.
Tegyük fel, hogy *A* és *B* egy-egy Debian csomag. Ebben az esetben a következő lehet a kettő csomag viszonya:

- *A* and *B* **independent**: *A* és *B* független egymástól.
- *A* **depends** *B*: *A* csomag függ *B*-től. Az '*A*' csomag működéséhez feltétlenül szükséges, hogy *B* telepítve legyen.
- *A* **recommends** *B*: *A* csomag javasolja *B*-t. Például az mc segítségével csak akkor tudunk belenézni a bzip2-vel tömörített fájlokban, hogyha a bzip2 telepítve van.

- *A* **suggests** *B*: Mivel a *B* csomag hasonló feladatot lát el, mint az *A* csomag, ezért lehet, hogy adott feladatra a *B* csomagot könnyebben lehet felhasználni.
- *A* **conflicts** *B*: *A* és *B* ütközik egymással. Ez annyit jelent, hogy egyidejűleg a két csomag nem telepíthető. Ennek az oka lehet például az, hogy ugyanaz a fájl található meg mindkettőben.
- *A* **replaces** *B*: *A* helyettesíti *B*-t. Egy korábban létező *B* csomag helyett egy új csomag (*A*) került a disztribúcióba.
- *A* **provides** *B*: *A* csomag tartalmazza *B*-t. Az *A* csomagban benne van mindaz, ami *B*-ben megtalálható.
- *A* **pre-depends** *B*: Előzetes függőség jelzése különböző speciális program-könyvtárakra vonatkozóan.

4.6.3. A csomagkezelés szempontjából fontos könyvtárak, fájlok

Mivel nagyon sok csomag érhető el Debian GNU/Linux operációs rendszerben, ezért fontos, hogy megfelelően legyenek ezek a csomagok rendszerezve. A rendszerezés esetében az egyik fontos szempont a csomagok felhasználására vonatkozó lehetőségek:

- **main**: A DFSG-ben szereplő szabad licenceknek megfelelő szoftverből készített csomagok.
- **contrib**: Azon csomagok, amelyekben lévő szoftverek megfelelnek a Debian szabad licencnek, csak függnek olyan csomagtól (szoftvertől), amely nem felel meg ennek.
- **non-free**: Azok a csomagok, amelyekben lévő szoftverekre vonatkozóan a fejlesztők olyan kikötéseket tettek, amely nem kompatibilis a Debian szabad licenccel.

Egy másik rendszerezési szempont az architektúra. Ennek megfelelően a következő könyvtárakat definiálhatjuk:

- **binary-all**: az összes architektúrára érvényes csomag
- **binary-<arch>**: az adott architektúrához tartozó csomagok, ahol <arch> egy a Debian által támogatott architektúra pl.: i386 vagy amd64
- **source**: a csomag forráskódjait, és a csomagkészítéshez használt fájlokat tartalmazza

Mivel nagyszámú a csomag, ezért egy harmadik rendszerezési szempont a csomag besorolása felhasználási szempontból. Ez a besorolás jelenik meg a Debian Control File Section mezőjében.

A csomagok esetében használt szekció elnevezések: 4. sz. táblázat

Név	***Tartalom***
admin	adminisztráció
comm	(alacsony szintű) kommunikáció
devel	fejlesztői („include”) fájlok
doc	dokumentáció
editors	szövegszerkesztők
electronics	elektronikai alkalmazások
embedded	beágyazott rendszerekhez köthető csomagok
games	játékok, oktatási célú csomagok
gnome	GNOME grafikus felülethez kapcsolódó csomagok
graphics	képszerkesztő, megjelenítő és grafikai alk.
hamradio	amatőr rádiózáshoz kapcsolódó csomagok
interpreters	interpreterek (értelmezők)
kde	KDE grafikus felülethez kapcsolódó csomagok
libdevel	dinamikus program-könyvtárak fejlesztői fájljai
libs	dinamikus program-könyvtárak
mail	elektronikus levelezés
math	matematikai programok
misc	egyéb, a többi szekcióba nem sorolható csomagok
net	hálózathoz kapcsolódó csomagok
news	hírcsoportok, hírolvasás (network news)
oldlibs	régebbi (libc5-höz) tartozó program-könyvtárak
otherosfs	más operációs rendszerek fájljaihoz, fájlrendszereihez való hozzáférés
perl	PERL modulok
python	PYTHON modulok
science	tudományos kutatás
shells	parancsértelmezők
sound	hang keltés, zene lejátszás
tex	a tex (főként matematikusok által kedvelt) szövegszerkesztő bővítményei
text	szövegfeldolgozás
utils	hasznos segédprogramok
web	WWW szolgáltatáshoz kapcsolódó csomagok
x11	X11 grafikus környezethez kapcsolódó csomagok

A Debian az előző három szempont figyelembevételével a következő könyvtárakat és fájlokat használja a csomagok és a csomagokra vonatkozó jellemzők tárolására:

Csomagok jellemzőinek tárolása:

- dists — a csomagok disztribúcióhoz való rendelése
  - etch
  - lenny
    - contrib
    - main
      - binary-<arch>
      - ...
      - binary-i386
        - Release
        - Packages, Packages.bz2, Packages.gz
      - source
        - Release
        - Sources, Sources.bz2, Sources.gz
    - non-free
    - Release, Release.gpg
  - squeeze
  - stable, testing, unstable szimbolikus linkek az adott verzióra

Az előző felsorolás a használt könyvtárstruktúrát mutatja be. Aláhúzással jelöltem az alkönyvtárakat. A dists könyvtárban tehát a disztribúció elnevezései találhatók és/vagy a rájuk utaló hivatkozások is. A legfontosabb tehát, hogy adott csomag melyik disztribúcióhoz tartozik. Egy szinttel beljebb találjuk a licenc szerinti csoportosítást. Ezen belül pedig az architektúránként történő szétválasztást. Ily módon tehát egy fa struktúrát lehet felrajzolni, amelynek a dists/etch/main/binary-i386 és a dists/etch/main/source ága lett teljes mértékben bemutatva a felsorolásban.

A sémában a könyvtárak mellett az alábbi fájlok használatosak:

- Release: az aktuális kiadásra vonatkozó információk, illetve a nyilvántartásban használt fájlokra vonatkozóan ellenőrző összegek
- Release.gpg: a Release fájlhoz tartozó aláírás
- Packages: az adott részfába tartozó összes bináris csomag vezérlő információi egyetlen fájlban (illetve ez tömörítve is megtalálható) + csomagonként a csomag helyére vonatkozó hivatkozás
- Sources: az adott részfába tartozó összes forrás csomag vezérlő információi egyetlen fájlban (illetve ez tömörítve is megtalálható) + csomagonként a csomag helyére vonatkozó hivatkozás

A Packages fájlban tehát benne van a csomagokban lévő vezérlő információ, amely ki van egészítve egyéb mezőkkel. Ez a korábbiakban példaként bemutatott „mc” csomag esetében a következő:

Filename: pool/main/m/mc/mc_4.6.2~git20080311-4_i386.deb
Size: 2140246
MD5sum: 088161530e6e78915cf097cf9af361ab

A csomag elérési útvonalát, méretét, és MD5 ellenőrző összegét is tartalmazza. A Packages fájl frissíthető a csomagokban lévő vezérlő információk, illetve a csomagok elérhetőségének együttes megadásával. A Sources fájl jellegét tekintve hasonlít a Packages fájlra. Annyi a különbség, hogy nem a bináris csomagok jellemzőit, hanem a forráscsomagok jellemezőit tartalmazza.

Csomagok tárolása:

- pool — a csomagok helye
  - contrib
  - main
    - 2
    - ...
    - lib*
    - ...
    - m
      - m-tx
      - ...
      - mc
        - mc_4.6.1-6_i386.deb
      - ...
      - mytop
    - ...
    - z
  - non-free

A csomagok tárolása is egy hierarchikus fa struktúrában történik. A tárolás módja függetleníthető a disztribúciótól, így csak a licenc szerint vannak a csomagok szétválogatva. Ezen belül pedig a csomag első karaktere (betűje) szerint további alkönyvtárakat találunk. A legutolsó szint a hierarchiában a csomag neve. Ezen belül találhatjuk meg a csomagot a korábban definiált név konvenciónak megfelelően. Az előző felsorolás a pool/main/m/mc részfát mutatja be, az elágazásoknál utalva az egyéb lehetőségekre.
A csomagok efféle tárolásának egyik fontos előnye, hogy egy helyen lehet tárolni a különböző terjesztésekhez (stable, testing, unstable) tartozó csomagokat. Az előző példánál maradva az összes mc nevű csomag a pool/main/m/mc könyvtáron belül lesz megtalálható. A másik fő előny, hogy könnyen lehet „szeparálni” az egyes csomagokat és azokat külön adathordozóra elhelyezni. Ez a folyamat egyszerű másolással és a Packages fájlok újra gyártásával megvalósítható.

4.6.4. A csomagkezelés parancsai

A jól megtervezett csomaghierarchiához kell, hogy tartozzanak olyan alkalmazások, amelyek segítségével a csomagok kezelése könnyen kivitelezhető.

Egy csomagkezelő alkalmazás legfontosabb feladatai:

- a csomagok épségének és digitális aláírásának ellenőrzése
- a csomagok jól áttekinthető listázása
- az egyszerű telepítés, frissítés, eltávolítás megvalósítása
- a csomagok közti függőségek, ütközések figyelése
- az elérhető csomagok listájának frissítése

A Debian GNU/Linux alap csomagkezelője a dpkg. Ennek a segítségével parancssorból megvalósíthatók a csomagokkal kapcsolatos műveletek, sőt akár a csomag létrehozása is.

A parancs fontosabb lehetőségei:

- `dpkg -i | --install <package_file>` # csomag telepítése
- `dpkg --unpack <package_file>` # csomag kicsomagolása
- `dpkg --configure <package>` # csomag beállítása
- `dpkg -r | --remove <package>` # csomag törlése
- `dpkg -P | --purge <package>` # csomag törlése konfigurációval együtt
- `dpkg -p | --print-avail <package>` # csomag jellemzőinek megjelenítése
- `dpkg -l | --list <name-pattern>` # csomagok listázása név alapján
- `dpkg -L | --listfiles <package>` # csomag tartalmának listázása
- `dpkg -S | --search <name-pattern>` # keresés csomagok tartalmában fájlnév illeszkedés alapján
- `dpkg –get-selections > <package-list>` # telepített csomagok listázása
- `dpkg --clear-selections` # a csomag kiválasztás törlése
- `dpkg –set-selections < <package-list>` # csomagok kijelölése telepítésre

Az előzőekben felsorolt parancsok esetében a paraméterek jelentése:

- `<package>` # egy csomag rövid neve
- `<package_file>` # egy csomag teljes neve a .deb kiterjesztéssel együtt
- `<name-pattern>` # keresési minta
- `<package-list>` # tetszőleges fájl, amely a csomagok neveit és állapotaikat tartalmazza

A dpkg parancs az előzőeken kívül még nagyon sok lehetőséggel rendelkezik. Például a parancs segítségével lehetséges saját csomag készítése is. Elsősorban olyankor használatos, amikor egyedileg történik csomagok telepítése vagy eltávolítása.

Egy csomag telepítésekor a következő történik :

- a csomag vezérlő információinak kiszedése a csomagból
- amennyiben a csomagnak egy korábbi változata telepítve van a rendszerben, akkor a korábbi csomag prerm szkriptjének futtatása
- a csomagban lévő preinst szkript futtatása, hogyha létezik
- a csomag tartalmának a kicsomagolása (a régi fájlok mentése)

- amennyiben a csomagnak egy korábbi változata telepítve volt a rendszerben, akkor a korábbi csomag postrm szkriptjének futtatása
- az új csomag konfigurációs fájljainak kicsomagolása
- a csomagban lévő postinst szkript futtatása, hogyha létezik

A dpkg parancs esetében definiálhatunk különböző állapotokat. Ezek lehetnek a kiválasztással összefüggőek, illetve ettől függetlenek. Az állapot mindegyik csomagra jellemző.
A kiválasztástól független állapotok:

- installed — a csomag sikeresen kicsomagolva és konfigurálva lett
- half-installed — a csomag telepítése elkezdődött, de nem fejeződött be
- not-installed — a csomag nincs telepítve
- unpacked — a csomag sikeresen kicsomagolva lett, de nincs konfigurálva
- half-configured — a csomag konfigurálása elkezdődött, de nem fejeződött be
- config-files — csak a csomaghoz tartozó konfigurációs fájlok léteznek a csomag esetében

A kiválasztás segítségével adhatjuk meg, hogy egy csomag egy művelet elvégzése után másik állapotba kerüljön. Itt a következő lehetőségek vannak:

- install — a csomag telepítésre van kijelölve
- deinstall — a csomag törlésre van kijelölve (a konfigurációs fájlok nem törlődnek)
- purge — a csomag törlésre van kijelölve a konfigurációs fájlokkal együtt
- hold — a csomag jelenlegi állapotának a megtartása

Debian GNU/Linux rendszerben léteznek a dpkg-re épülő magasabb szintű csomag kezelők (frontend-ek). Ezek közül a leggyakrabban használt az APT (Advanced Packaging Tools).

Az APT segítségével van arra lehetőségünk, hogy hálózati csomag forrásokat használjunk. Az APT esetében fontos konfigurációs fájl a /etc/apt/sources.list. Ebben a fájlban adhatjuk meg, hogy milyen hálózati erőforrásokból akarunk csomagokat telepíteni. A fájl egy sorának felépítése a következő sémát követi:

```
type uri distribution section1 section2 ...
```

Az egyes mezők jelentése a következő:

- type: az erőforrás típusát jelenti, értéke a **deb** vagy a **deb-src** lehet. A deb utal a bináris csomagok lelőhelyére, a deb-src a forráscsomagokéra.
- uri: tetszőleges URI (Uniform Resource Identifier)
- distribution: egy disztribúció vagy változat neve
- section[12...]: csomagok licenc szerinti csoportja

Példák erőforrás megadásra:
deb file:/mnt/package/debian stable main contrib non-free
deb ftp://ftp.hu.debian.org/debian/ lenny main non-free contrib
deb http://kami.duf.hu/debian/ lenny main contrib non-free
deb http://security.debian.org/ lenny/updates main contrib non-free

Az APT esetében használhatjuk az alábbi helyi és hálózati erőforrásokat:

- file
- cdrom
- ftp
- http

Amennyiben tehát szeretnénk elérni egy APT erőforrást, első lépésben a megfelelő bejegyzést kell elhelyeznünk a /etc/apt/sources.list nevű fájlba. Amennyiben szeretnénk saját tükörszervert létrehozni, azt egyszerűen megtehetjük a „debmirror" parancs használatával. A parancs képes arra, hogy adott helyről a paraméterként átadott disztribúciót tükrözze egy adott gépre.

```
debmirror -v --method=ftp --passive \
-a i386,amd64 -d lenny -s main \
--progress --host=ftp.hu.debian.org /mnt/data/debian
```

Az előző parancs az i386, illetve az amd64 architektúra esetében a lenny disztribúcióhoz tartozó main csoportba sorolható csomagokat tölti le ftp-n keresztül az ftp.hu.debian.org-ról. Amennyiben előállítjuk a csomagok tárolásánál bemutatott módon az egyes könyvtárakat és fájlokat, abban az esetben máris készen van a csomag lelőhely. Ezt pedig egyszerűen megoszthatjuk egy web szerver vagy egy ftp szerver felhasználásával.

Ahhoz, hogy csomagokat érjünk el APT forrásból, szükségünk van az apt-get parancsra. Az apt-get parancs fontosabb lehetőségei:

- `apt-get update` # csomag cache frissítése
- `apt-get upgrade` # új csomagok telepítése
- `apt-get dselect-upgrade` # új csomagok telepítése a dselect parancshoz hasonlóan
- `apt-get dist-upgrade` # disztribúció frissítése
- `apt-get install <package>` # adott csomag telepítése
- `apt-get remove <package>` # adott csomag törlése (konfigurációt nem)
- `apt-get remove --purge <package>` # adott csomag törlése konfigurációs fájlokkal együtt
- `apt-get source <package>` # forrás csomag letöltése és kicsomagolása
- `apt-get clean` # letöltött csomagok törlése

Miután az APT forrásokat beállítottuk, utána szükséges az `apt-get update` parancs használata. A parancs letölti a beállított források esetében a Packages és a Sources fájlokat. Ezután van arra lehetőség, hogy az újonnan beállított helyekről telepítsünk vagy a rendszerünket frissítsük.

Az APT működését befolyásolhatjuk a /etc/apt/apt.conf segítségével.

```
APT::Default-Release "5.0";
Acquire
{
        http {
                Proxy "http://proxy.duf.hu:3128";
                Proxy::kami.duf.hu "DIRECT"
        };
        ftp {
                Passive "true";
        };
};
APT::Cache-Limit 16777216;
```

Abban az esetben, hogyha sok APT bejegyzésünk van előfordul, hogy meg kell növelnünk a cache méretét. A példában 16MB-ra van beállítva a cache mérete. Beállíthatjuk, hogy milyen http proxy-t használjon a hálózati csomagok eléréséhez, illetve mely szerverek esetében ne használja a proxy-t.
Van arra is lehetőségünk, hogy ún. kevert disztribúciót használjunk. Ez annyit jelent, hogy bizonyos csomagokat a stable, másokat a testing és/vagy az unstable elemei közül választunk. Ebben az esetben fontos lehet az elsőbbségi sorrend felállítása az egyes disztribúciók elemei között. Ezt a /etc/apt/preferences fájl segítségével tehetjük meg.

```
Package: *
Pin: release a=lenny
Pin-Priority: 1000
Package: *
Pin: release a=squeeze
Pin-Priority: 900
Package: *
Pin: release a=sid
Pin-Priority: 800
```

Ebben az esetben az összes csomagra vonatkozóan állítjuk a prioritást. A lenny-ben lévő csomagok rendelkeznek a legnagyobb prioritással, a squeeze ennél kevesebb, a sid (unstable) csomagjai pedig a legkevesebbel. Természetesen a működéshez az kell, hogy az adott disztribúciókra utaló hivatkozások legyenek a /etc/apt/sources.list-ben.
Az apt-get parancs esetében megadhatjuk azt is telepítéskor, hogy melyik disztribúcióból vegye a csomagokat (amennyiben több is van az APT források között).

```
apt-get -t squeeze install mc
```

Az APT-hez kapcsolódik meg néhány további parancs:

- apt-cache csomag cache manipuláció, csomag keresés
- apt-cdrom: telepítő CD/DVD felismertetése
- apt-file csomag keresés és listázás
- apt-key a csomagok ellenőrzésénél használt kulcsok kezelése

Az apt-get mellett egyéb eszközök is léteznek a csomagok menedzselésére. Az egyik ilyen alkalmazás a dselect. Ez képes együttműködni az APT-vel, ugyanazokat a forrásokat tudjuk felhasználni mindkét esetben. A dselect esetében egy menüből választhatjuk ki az elvégzendő tevékenységeket:

0. [E]lérés Válassz elérési módot
1. [F]rissít Frissíti az elérhető csomagok listáját, ha lehetséges.
2. [V]álaszt Csomagok választása
3. [T]elepít Kért csomagok telepítése, frissítése.
4. [B]eállít Beállítatlan csomagok beállítása.
5. [L]eszedés Nem kívánt szoftverek eltávolítása.
6. [K]ilépés Kilépés a dselect programból.

A program esetében ajánlott az egyes menüpontokon a felsorolt sorrendben végigmenni. Annyi előnye van az apt-get paranccsal szemben, hogy itt egyéb parancs nélkül el tudjuk olvasni az adott csomagra vonatkozó leírás mezőt, amely alapján eldönthetjük, hogy szükségünk van-e az adott csomagra.

Miután a csomagkezelő megfelelő módon konfigurálva lett, úgy elérhetjük a rendelkezésre álló csomagokat. Ahhoz, hogy a rendszerünk képes legyen adott hálózati feladatok ellátására ahhoz a megfelelő csomagok telepítésére van szükségünk. Illetve természetesen ezeket a szoftvereket a saját felhasználási igénynek megfelelően konfigurálnunk kell.

5. A diszk adminisztráció és parancsai

5.1. A diszkkezelés és lehetőségei

Egy rendszer telepítése előtt döntenünk kell a diszkjeink felhasználása felől. A könyvtár hierarchia egyes részei különböző méretű partíciókat igényelnek. A pontos méretek természetesen függnek az adott felhasználási területtől. Amikor egy sémát tervezünk, akkor legalább a következő könyvtárakat célszerű figyelembe venni a telepítése esetében:

- / mérete legalább néhány GB (mérete függ attól, hogy a felhasználói programokat külön helyre rakjuk-e)
- /boot max. 100 MB, a kernel, initrd és a boot loader számára szükséges hely
- /home a felhasználók adatai számára fenntartott hely igény szerint
- /usr a felhasználói programok számára fenntartott hely igény szerint
- /tmp átmeneti fájlok tárolása, legalább 500 MB javasolt, felhasználás függvénye
- /var változó adatok, napló fájlok, igény szerint

Természetesen dönthetünk úgy, hogy az előzőekben felsorolt könyvtárakat nem külön partíciókra helyezzük, hanem bizonyosakat összevonunk. Vigyáznunk kell azonban arra, nehogy „hátrányosan befolyásolják” egymás működését.
Néhány példa, amely igazolja a partíciók külön választását:

A /var-t célszerű külön választani a /-től, nehogy a naplófájlok kitöltsék a teljes partíciót, és ebben az esetben nem lehetséges semmilyen új program telepítése, vagy bármilyen fájl létrehozása.
A /tmp méretének kellően nagynak kell lenni, hogy a különböző átmeneti fájloknak elegendő helyet biztosítson. Vigyázni kell vele is, mert könnyen megtelhet.
A felhasználók könyvtárai is célszerű külön választani, mert ellenkező esetben az adott lemez nagy részét kitölthetik az felhasználóink dolgai, és esetleg kevés hely marad az alkalmazásaink számára.

Amellett, hogy figyelembe vesszük az egyes könyvtárak méret igényeit, figyelnünk kell a virtuális memória (swap) méretére is. Általános szabályt nehéz mondani a méret megválasztására vonatkozóan. Általában legalább annyit, mint a fizikai memória, de nem többet, mint a fizikai memória kétszerese. A mai memória méretek és árak mellett ezzel nem érdemes spórolni. A nagyobb méretű memória nagyon megdobja a rendszer teljesítményét, és csak nagyon ritkán lesz szüksége a rendszernek a tárcserére.

A diszk kezelés esetében megkülönböztethetünk különböző módszereket:

- csak **fizikai kötetek (partíciók)** használata. Ebben az esetben hagyományos módon partíciókat hozunk létre, és ezek csatlakoztatjuk.
- létrehozunk fizikai köteteket, de ezeket nem közvetlenül használjuk. A Debian esetében van arra lehetőség, hogy **titkosított fájlrendszer**t használjunk. Ebben az esetben az adatok titkosítva kerülnek a partícióra,

és csak megfelelő jelszó vagy kulcs megadása után érhetők el. A kernel device mapper speciális eszköze (dm-crypt) segítségével érhetjük el a rajta lévő adatokat.
- **LVM (Logical Volume Management)** használata. Ebben az esetben a fizikai kötetekből kötetcsoportokat képezünk, majd ezeket felosztjuk logikai kötetekre. Ezeket a logikai köteteket használjuk az adataink tárolására. Fontos előnye, hogy a logikai kötetek átméretezhetők.
- **Szoftveres RAID (md)** használata. A Linux lehetővé teszi azt, hogy több fizikai kötetet összefogjunk, és szoftveres úton megoldjuk a redundáns adattárolást.
- **EVMS (Enterprise Volume Management System)** használata. A Debian telepítője ezt a speciális módszert nem támogatja. Akár parancssoros, ncurses alapú, vagy grafikus felületen keresztül is megoldható a segítségével a diszk adminisztráció. Képesen vagyunk a RAID illetve LVM logikai köteteket is felhasználni a lemezkezelés során. Főként sok diszket tartalmazó robusztus rendszerek esetében ajánlott.

5.2. A Szoftveres RAID (md)

A RAID (Redundant Array of Independent Disk) egy fontos technológia az adatok tárolása esetében. A számítógépekben levő merevlemez könnyen és gyakran meghibásodhat. Ezért fontos, hogy az adataink ne csak egy eszközön, hanem több helyen legyenek tárolva. A RAID a redundáns tárolás mellett az I/O műveletek gyorsítására is kiválóan alkalmas.

A RAID-nek létezik hardveres és szoftveres megvalósítása is. A hardveres implementáció esetében a vezérlőkártya feladata a RAID tömb kezelése és a párhuzamos írási műveletek végrehajtása. Legfőbb előnye, hogy ebben az esetben a lemez I/O művelet nem terheli a CPU-t. Hátránya viszont az, hogy az ilyen kártyák ára százezer forint fölött kezdődik.
A RAID-nek létezik szoftveres megvalósítása is. Ebben az esetben az operációs rendszer egy összetevőjének a feladata, hogy megvalósítsa a RAID funkcióit. A Linux kernel esetében az MD (Multiple Device) driver látja el ezt a feladatot.

A RAID esetében különböző szinteket (RAID level) definiálhatunk:

- **RAID 0** (striping): A merevlemezt vagy partíciót meghatározott méretű részekre (sávokra) bontjuk. A RAID 0 esetében felváltva történik ezen sávok összefésülése. Így az adatok tárolása egyszerre több eszközre történik. Ezáltal az I/O műveletek gyorsulására számíthatunk. A RAID 0 azonban nem biztosít hibatűrést. Ezért ha az eszközeink közül bármelyik meghibásodik, akkor az adatunk egy része elvész.
- **RAID 1** (mirroring): Ebben az esetben történik az adatok duplikált tárolása. Ehhez legalább kettő diszk szükséges (ajánlott, hogy egyformák legyenek). Ebben az esetben az eszközök tárolási kapacitása nem adódik össze úgy, mint a RAID0 esetében. A RAID1 tömb beállítása után egy diszk kapacitásának megfelelő terület lesz elérhető. A RAID1 esetében az olvasási sebesség megnőhet, hiszen több lemezről az olvasás egyszerre megtehető. Az írási művelet esetében azonban nem eredményez gyorsulást. Egy meghajtó meghibásodását képes túlélni.

- **RAID 4** (striped set with dedicated parity): Ebben az esetben több merevlemez egymástól függetlenül működik. Legalább három diszkre van szükségünk a megvalósításához. Az egyik diszken történik a paritás információk tárolása. Lemez meghibásodás esetében ennek a felhasználásával lehetséges az adatok visszaállítása. Olvasási sebessége jó, viszont az írási művelet esetében a paritás diszk jelenti a szűk keresztmetszetet.
- **RAID 5** (striped set with distributed parity): Hasonlóan működik, mint a RAID 4, azzal a különbséggel, hogy a paritás információk nem egyetlen diszken, hanem az összesre szétosztva tárolódnak. Így kiküszöböli a paritás meghajtó által jelentett szűk keresztmetszetet. Legalább három meghajtó szükséges hozzá, és kettő meghajtónyi tároló kapacitást biztosít. Az egyik legelterjedtebb RAID szint, mivel egyszerre biztosítja a redundáns tárolást, illetve az olvasási sebesség többszörözését.
- **RAID 6** (striped set with dual distributed parity): Annyiban különbözik a RAID 5 -től, hogy ebben az esetben a paritást nemcsak soronként, hanem oszloponként is képezik. Így három diszk esetében is csak egy meghajtónyi kapacitást biztosít, viszont a segítségével kettő merevlemez meghibásodás is kiküszöbölhető. A többletköltségek miatt nem terjedt el.
- **RAID 1+0** vagy **RAID 10** (mirrored striping): Ebben az esetben legalább négy eszközre van szükségünk. Először párban kettőt-kettőt RAID1-be kapcsolunk, majd a ezeket kapcsoljuk össze RAID0-vá. Ezáltal ötvözzük a RAID0 és a RAID1 előnyeit. Kettő (nem tetszőleges) diszk meghibásodást is elvisel.

Az előző felsorolásban nem került bele az összes RAID szint, csak azok, amelyek a Linux kernel esetében léteznek, és felhasználhatók. Ezek mellett létezik például RAID 2, 3, 50 is.
A telepítő kernele az előző felsorolásban szereplő szintek közül csak a RAID 0,1,5,6,10 szinteket támogatja.

A RAID esetében néhány fontos fogalommal tisztában kell lennünk:
- szinkronizálás: Amennyiben a RAID1 tömb valamelyik eleme meghibásodik, úgy a hibás merevlemezt pótolni kell. Miután megtörtént a pótlása fontos, hogy a másik diszk tartalma rá kerüljön. Amíg a két diszk között a szinkronizáció folyik, nem célszerű I/O igényes folyamatokat futtatni, mert ez önmagában is megterhelő a gép számára.
- tartalék (spare) diszk: A RAID0 kivételével van arra lehetőség, hogy a működő winchesterek mellé elhelyezzünk egy tartalék diszket. Ez alapesetben nincsen használatban. Szerepe akkor fontos, amikor valamelyik diszkünk meghibásodik. Ebben az esetben, a tartalék eszköz a meghibásodott helyébe léphet, és megkezdődhet a diszk szinkronizálása és „beépítése” a RAID-be.
- RAID állapot: Ez utal a RAID illetve a benne lévő eszközök működésére. A „clean” állapot jelenti, hogy minden rendben működik. Amikor egy diszk meghibásodik, akkor a rendszer továbbra is működik, de átkerül „dirty” állapotba. Amennyiben a hibás diszket eltávolítottuk, úgy „degraded” állapotban kerülünk.

Ahhoz, hogy szoftveres RAID-et használjunk, szükségünk van az mdadm nevű

csomagra, illetve a benne lévő ugyanolyan nevű parancsra.

Példa RAID használatára az mdadm parancs segítségével

RAID1 tömb létrehozása

```
mdadm --create --verbose /dev/md0 --level=1
--raid-devices=2 /dev/sd[cd]
--spare-devices=1 /dev/sde
```

RAID leállítása

```
mdadm --stop /dev/md0
```

Eszköz eltávolítása

```
mdadm --fail /dev/md0 /dev/sdc
mdadm --remove /dev/md0 /dev/sdc
```

Eszköz hozzáadása

```
mdadm --add /dev/md0 /dev/sdc
```

A RAID-ben lévő eszközök állapotáról a /proc/mdstat fájl alapján tájékozódhatunk:

```
Personalities : [raid1]
md0 : active raid1 sdc[0] sdd[1]
      4194240 blocks [2/2] [UU]
```

Az adott md eszközre vonatkozó beállításokat megtekinthetjük:

```
mdadm --detail /dev/md0
```

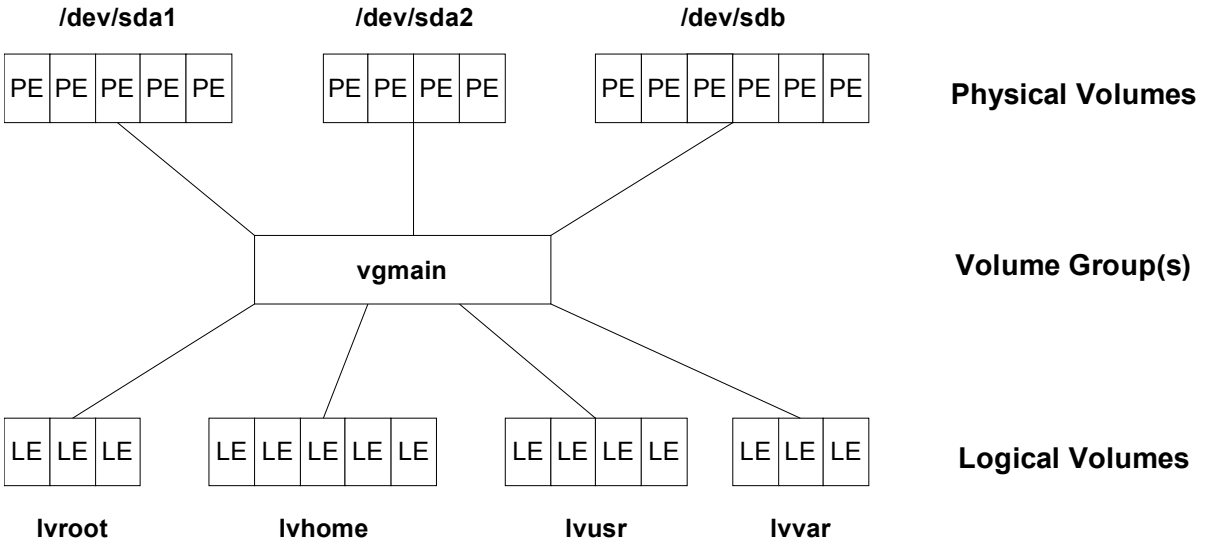
```
/dev/md0:
        Version : 00.90.03
  Creation Time : Wed Feb 27 07:37:10 2008
     Raid Level : raid1
     Array Size : 4194240 (4.00 GiB 4.29 GB)
    Device Size : 4194240 (4.00 GiB 4.29 GB)
   Raid Devices : 2
  Total Devices : 2
Preferred Minor : 2
    Persistence : Superblock is persistent

    Update Time : Sat Mar  8 00:24:43 2008
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 0

           UUID : b17bc19f:8da2da7b:fab91c8b:5e7c77de
         Events : 0.2

    Number   Major   Minor   RaidDevice State
       0       8       32        0      active sync   /dev/sdc
       1       8       48        1      active sync   /dev/sdd
```

A szoftveres RAID egy olcsó megoldás arra, hogy adatainkat redundáns módon tároljuk. Éppen ezért használata bármilyen telepítés esetében javasolt, ahol rendelkezésre áll több diszk. Elsősorban SATA vagy SCSI diszkekkel ajánlott használni, az IDE esetében gyakran tapasztalhatunk például sebességbeli problémákat.

5.3. Az LVM

Az LVM egy magasabb absztrakciós réteget definiál, mint a hagyományos diszkek vagy partíciók. A Linux kernel Device mapper támogatása teszi lehetővé a használatát. A 2.6-os kernelekben az LVM2 támogatás érhető el. Ahhoz, hogy a logikai kötetkezelést használjuk szükségünk van userspace-beli binárisokra (lvm2 nevű csomag).
Az LVM működének megértéséhez definiálnunk kell fogalmakat, rövidítéseket:

- PE (Physical Extent): adott eszköz vagy partíció egy előre meghatározott méretű része. Egy fizikai kötet több azonos méretű ilyen egységre bontható.
- PV (Physical Volume): fizikai kötet (partíció)
- VG (Volume Group): kötet csoport. Egy vagy több fizikai kötet egyesítésével képezhető.
- LV (Logical Volume): logikai kötet, amely egy kötetcsoporton belül helyezkedhet el
- LE (Logical Extent): egy logikai kötet előre meghatározott méretű része



Minta logikai kötetkezelésre 1. sz. ábra

Az előző ábra bemutatta, hogy a logikai kötetek és a fizikai kötetek hogyan kapcsolódnak egymáshoz.

Az LVM legfontosabb előnyei, tulajdonságai:

- több lemez összefűzhető egy „nagy méretű" kötetté
- a kötetcsoportok működés közben átméretezhetők, új diszkek adhatók a kötetcsoporthoz, illetve meglévők vehetők el belőle

- a logikai kötetek átméretezhetők újabb LE-k hozzáadásával vagy elvételével
- lehetséges ún. pillanatfelvétel (snapshot) készítése a logikai kötetekről, amelyet az LVM2 esetében írni is és olvasni is lehet
- a logikai kötetek áthelyezhetők működés közben másik fizikai kötetre
- a logikai köteteket sávokra lehet bontani és ezeket a RAID0-hoz hasonlóan el lehet osztani a fizikai kötetek között
- egy logikai kötetet lehet tükrözni, és ezzel a RAID1-hez hasonló működést lehet elérni
- van arra is lehetőség, hogy kötetcsoportokat osszunk több részre vagy többet egyesítsünk. Ebben az esetben feltétel az, hogy a kötetcsoporton belül ne legyenek hozzárendelt logikai kötetek.

Az LVM használata egyszerűsíti és hatékonyabbá teszi a lemezkezelést főleg abban az esetben, amikor sok diszkkel rendelkezünk.

A rendszerünk sebességét befolyásolja az, hogy milyen a viszony a fizikai illetve a logikai kötetek között.
- A lineáris leképezés (linear mapping) esetében egy logikai kötet első része egy fizikai köteten található, míg a második fele egy másik fizikai köteten. Vagyis tekinthetünk rá úgy, mint kettő fizikai partíció egymás után történő összefűzése. Így könnyen előfordulhat, hogy a lemezeink közül csak az egyik dolgozik, a másik nem végez semmilyen műveletet adott fájl íráskor vagy olvasáskor.
- A sávokra való leképezés (striped mapping) esetében a logikai kötet egymás után következő részei nem egy diszkről kerülnek ki, hanem felváltva a rendelkezésre álló fizikai kötetek közül. Ez jelentős mértékben javíthatja a rendszerünk teljesítményét.

A következőkben szeretnék bemutatni egy példát LVM használatára:
- A partíciók és eszközök felhasználási sémájának megtervezése
- Fizikai kötetek létrehozása: `pvcreate /dev/sda2 /dev/sdb`
- Kötetcsoport(ok) létrehozása: `vgcreate vg1 /dev/sda2 /dev/sdb`
- Kötetcsoport(ok) frissítése: `/etc/init.d./lvm restart`
- Logikai kötetek létrehozása: `lvcreate -L 5g -n vol1 vg1`
- Fájlrendszer létrehozása: `mkfs.ext3 /dev/mapper/vg1_vol1`
- Fájlrendszer csatlakoztatása: `mount /dev/mapper/vg1_vol1 /mnt/vol1`

Az LVM-el kapcsolatban fontos megemlíteni, hogy nem biztosít redundáns adattárolást. Éppen ezért a gyakorlatban valamilyen hibatűrő RAID szinttel együtt ajánlott használni.
Egy logikai kötet maximális mérete 2.6-os kernel esetében 32 bites rendszerben 16TB, 64 bites rendszerben pedig 8EB.

A szoftveres RAID esetében gyakorlatilag az mdadm parancs segítségével meg tudjuk oldani a RAID-el kapcsolatos műveleteket. Ezzel szemben az LVM2 esetében több parancsunk is van. A parancsokat csoportosíthatjuk aszerint, hogy fizikai kötetekhez, kötetcsoportokhoz vagy logikai kötetekhez kapcsolódnak. A parancsok nevéből erre egyszerűen következtethetünk.

A fizikai kötetekhez kapcsolódó parancsok:

- pvs: fizikai kötetekre vonatkozó információk összegzése
- pvscan: fizikai kötetek keresése a diszkeken
- pvdisplay: fizikai kötetek jellemzőinek megtekintése
- pvcreate: fizikai kötet létrehozása
- pvchange: fizikai kötet attribútumainak megváltoztatása
- pvmove: PE-k mozgatása másik fizikai kötetre
- pvremove: fizikai kötet megszüntetése. Ez után a partíció már nem használható az LVM-el.
- pvresize: fizikai kötet átméretezése (miután pl. az fdisk-el a partíciót átméreteztük)

A kötetcsoportokhoz kapcsolódó parancsok:

- vgs: kötetcsoportokra vonatkozó információk összegzése
- vgscan: kötetcsoportok keresése a diszkeken
- vgdisplay: jellemzőinek megtekintése
- vgcreate: kötetcsoport létrehozása
- vgchange: kötetcsoport attribútumainak megváltoztatása
- vgrename: kötetcsoport átnevezése
- vgsplit: kötetcsoport két részre való hasítása
- vgmerge: kettő kötetcsoport egyesítése
- vgextend: kötetcsoport méretének növelése
- vgreduce: kötetcsoport méretének csökkentése
- vgck: kötetcsoportra vonatkozó metaadatok ellenőrzése
- vgmknodes: a /dev könyvtárban a kötetcsoportokhoz tartozó eszközfájlok létrehozása
- vgconvert: LVM1 kötet konvertálása LVM2 kötetté
- vgexport: kötetcsoport elrejtése
- vgimport: rejtett kötetcsoport újra ismertté tétele
- vgremove: kötetcsoport törlése

A logikai kötetekhez kapcsolódó parancsok:

- lvs: logikai kötetekre vonatkozó információk összegzése
- lvscan: logikai kötetek keresése a diszkeken
- lvdisplay: logikai kötetek jellemzőinek megtekintése
- lvcreate: logikai kötet létrehozása
- lvchange: logikai kötet attribútumainak megváltoztatása
- lvrename: logikai kötet átnevezése
- lvextend: logikai kötet méretének növelése
- lvreduce: logikai kötet méretének csökkentése
- lvresize: logikai kötet átméretezése
- lvconvert: logikai kötet konvertálása (linear, mirror, snapshot módok között)
- lvremove: logikai kötet megszüntetése

A parancsok neveiből látszik, hogy a fejlesztők egy előre definiált sémát követtek a parancsok neveinek meghatározásakor.

5.4. Egy RAID1-et, illetve LVM2-t együttesen használó rendszer telepítése

Amikor egy rendszert telepítünk nem tudjuk pontosan azt, hogy az új rendszert mire fogjuk használni a későbbiekben. Előfordulhat az, hogy a használt fizikai partíciók méretét nem megfelelően választjuk meg. Az előző fejezet alapján láthatjuk, hogy az LVM segítségével az efféle problémák kiküszöbölhetőek. Sőt egyes fájlrendszerek esetében megoldható az is, hogy működés közben átméretezzük a fájlrendszert.

Tegyük fel, hogy rendelkezünk négy darab diszkkel, amelyeket a /dev/sda, /dev/sdb, /dev/sdc, /dev/sdd eszközfájlokon keresztül érhetünk el. A diszkek mérete és típusa legalább páronként megegyezik. A /dev/sda és a /dev/sdb eszközök kettő partícióra lennének osztva:

/dev/sda1	100M	RAID típusú (FD)
/dev/sda2	fennmaradó szabad hely	RAID típusú (FD)

A /dev/sdc és /dev/sdd nem lenne particionálva, hanem csak RAID1-be összekapcsolva. A /dev/sda1 a /dev/sdb1-el lenne RAID1-be kapcsolva, a /dev/sda2 pedig a /dev/sdb2-vel. Ez utóbbi pedig LVM segítségével össze lenne kapcsolva a /dev/sd[cd]-ből képzett RAID1-el.

A Debian 5.0 telepítő menürendszere nem támogatja azt, hogy diszkeket használjunk partíciós tábla nélkül. Azonban parancsok segítségével ez is kivitelezhető.

Telepítsük a Debian GNU/Linux 5.0-t a szokásos módon, majd jussunk el a telepítő menüből a „Lemezek particionálása” menü pontig (De ezt még ne futtassuk.). Ajánlott expert módban végezni a telepítést. Ebben az esetben az előzőekben definiált sémát nem a telepítő segítségével, hanem parancsok felhasználásával fogjuk megoldani.
ALT-F2 billentyűkombinációval átválthatunk egy második virtuális konzolra, ahol ENTER megnyomására kapunk egy root shellt. Itt a következő parancsokkal az előzőekben definiált séma kivitelezhető:

- `fdisk /dev/sda` a kettő partíció létrehozása és típusának beállítása FD-re (RAID auto detect)
- `sfdisk -d /dev/sda | sfdisk /dev/sdb` a /dev/sda diszken lévő partíciós tábla átmásolása /dev/sdb eszközre
- ```
  mknod /dev/md0 b 9 0
  mknod /dev/md1 b 9 1
  mknod /dev/md2 b 9 2
  ```
  mivel nem léteznek az egyedi md eszközöknek megfelelő fájlok, ezért azokat létre kell hozni
- ```
  chmod 660 /dev/md[0-2]
  chown root.6 /dev/md[0-2]
  ```
  a létrehozott eszközfájlok tulajdonjogának és jogosultságainak beállítása. Mivel a disk nevű csoport itt még nem létezik, ezért a GID-jét használom.
- ```
  modprobe dm-mod
  modprobe raid1
  ```
  a Device mapper és a RAID1 támogatás betöltése
- ```
  mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sd[ab]1
  mdadm --create /dev/md1 --level=1 --raid-devices=2 /dev/sd[ab]2
  ```

`mdadm --create /dev/md2 --level=1 --raid-devices=2 /dev/sd[cd]`
három RAID1 létrehozása

- `pvcreate /dev/md[12]`
  fizikai kötet létrehozása a RAID1 eszközökön
- `vgcreate vgmain /dev/md[12]`
  a két fizikai kötetből a „vgmain” kötetcsoport létrehozása
- `lvcreate -L 2000m -n lvroot vgmain`
  `lvcreate -L 1000m -n lvvar vgmain`
  `lvcreate -L 500m -n lvswap vgmain`
  három logikai kötet létrehozása a „vgmain” kötetcsoporton belül. Az első neve „lvroot”, mérete 2000MB. A második neve „lvvar”, mérete 1000MB. A harmadik neve „lvswap”, mérete 500MB.

Miután ezeket a parancsokat kiadtuk, ALT-F1 megnyomásával visszaválthatunk a telepítő menüjébe. Amikor a „Lemezek particionálása” menüt kiválasztjuk, akkor a telepítő automatikusan érzékeli a parancsok segítségével létrehozott RAID-et, illetve logikai köteteket. Így csak annyi a dolgunk, hogy az egyes logikai köteteknél kiválasszuk a felhasználás módja esetében a megfelelő fájlrendszert, és ezután a többi szokásos telepítési lépést elvégezzük.

A /dev/md0 legyen a /boot-ban csatlakoztatva, a /dev/mapper/vgmain-lvroot (vgmain kötetcsoporton belüli lvroot logikai kötet) pedig a / fájlrendszer. A /dev/mapper/vgmain-lvvar a /var alá legyen csatlakoztatva, a /dev/mapper/vgmain-lvswap pedig virtuális memóriaként legyen használva.

Miután a telepítés befejeződött, és az új rendszer elindult, egy dolog van még hátra. Alapértelmezésként a telepítő a GRUB-ot az első diszk (/dev/sda) MBR-jébe telepítette. Felteszem, hogy ennél a kérdésnél nem változtattuk meg a telepítő által felajánlott beállítást. Abban az esetben, hogyha a /dev/sda eszközünk meghibásodna, akkor a rendszer nem lenne képes önállóan elindulni, mivel a GRUB csak a /dev/sda eszköz elejére lett telepítve. Azért, hogy a rendszerünk ez ellen is védve legyen a /dev/sdb eszköz esetében is telepítenünk kell a GRUB-ot. Ezt például a következő módon tehetjük meg:

- Bejelentkezünk root-ként a működő rendszerbe.
- Kiadjuk a `grub` parancsot, ezzel elindul a grub shell.
- Ezután pedig a következő parancsokat kell kiadnunk:
  ```
  device (hd0) /dev/sdb
  root (hd0,0)
  setup (hd0)
  ```
- Miután ez végrehajtódott, utána kiléphetünk a grub shell-ből a `quit` paranccsal.

A grub esetében a device paranccsal módosítottuk, hogy melyik eszköz legyen a hd0. Ez egészen addig marad érvényben, amíg ki nem lépünk a grub shellből. A root paranccsal adjuk meg, hogy melyik diszk melyik partíciójára települjön a GRUB. A GRUB 0-től kezdi az elsődleges partíciók sorszámozását, nem pedig 1-től ahogyan a Linux kernel. Ennélfogva a GRUB ide helyezi a saját fájljait, amelyek a setup parancs kiadása után másolódnak erre a helyre, illetve a GRUB a diszk MBR-jébe települ. (Mivel RAID1 van, ezért a GRUB fájljai már amúgy is elérhetők ebben a könyvtárban.)
A telepített rendszerünk tehát megvalósítja a redundáns adattárolást, és a logikai

kötet kezelést egyszerre. Amellett, hogy többszörösen vannak tárolva az adataink, arra is van lehetőségünk, hogy kihasználjuk az LVM adta lehetőségeket, előnyöket.

5.5. A diszk particionálás és a fájlrendszer létrehozás

5.5.1. A diszk particionálás

Amikor egy winchestert szeretnénk használni akkor nem feltétlenül egyben szeretnénk látni a teljes diszket. Éppen ezért az eszközt több részre, partícióra oszthatjuk. A lemezen lévő partíciókat jellemzőit a partíciós tábla tárolja. Ennek a típusa is többféle lehet. Mi a továbbiakban csak az msdos típusú partíciós táblával fogunk foglalkozni.

A lemezek particionálásának alapvető eszköze az fdisk parancs.

```
fdisk device
```

Az fdisk parancs indítása után a paraméterként átadott blokkos eszközön lehetséges a partíciós tábla módosítása. Miután a parancs elindul, akkor egy saját felületet biztosít a parancsok begépeléséhez. Az itt elérhető fontosabb parancsok:

- a: a boot flag (aktív vagy nem aktív) változtatása egy partíció esetében
- c: DOS kompatibilis mód ki-be kapcsolása
- d: partíció törlése sorszám alapján
- l: az ismert partíció típusok listázása
- m: az elérhető parancsok listájának kiírása
- n: új partíció létrehozása
- o: üres partíciós tábla létrehozása
- p: a partíciós tábla jelenlegi állapotának kiírása
- q: kilépés mentés nélkül
- t: partíció típusának módosítása sorszám és típuskód alapján
- v: a partíciós tábla ellenőrzése
- w: a változások mentése

Az fdisk-nek létezik egy ncurses alapú frontend-je (cfdisk). Ez annyit jelent, hogy karakteres képernyőn menü-vezérelt módon képesek vagyunk a partíciós tábla manipulációjára. Ennélfogva a legtöbb ember számára jóval kényelmesebb a használata. A program annyit tesz, hogy az egyes menübeli eseményeknek megfelelően fdisk parancsokat hoz létre, majd ezeket futtatja le az fdisk segítségével.
Az sfdisk parancs szintén a partíciós tábla manipulációjára használható. Legfontosabb felhasználási lehetősége a partíciós tábla mentése illetve visszaállítása.

Partíciók mentése: `sfdisk -d /dev/sda > sda_PART`
Partíciók visszaállítása: `sfdisk /dev/sda < sda_PART`

Az sfdisk saját formátumot használ a mentésre. Az előző alfejezetben használt telepítési példa esetében a tartalma például a következő:

```
# partition table of /dev/sda
unit: sectors

/dev/sda1 : start=       63, size=   208782, Id=fd, bootable
/dev/sda2 : start=   208845, size=  8177085, Id=fd
/dev/sda3 : start=        0, size=        0, Id= 0
/dev/sda4 : start=        0, size=        0, Id= 0
```

A Linux szempontból fontos partíciós tábla bejegyzés típusok a következők:

- 0x83 Linux
- 0x82 Linux swap
- 0x85 Linux extended
- 0x8e Linux LVM
- 0xfd Linux raid autodetect

Az előző felsorolásban a 0x jelzi, hogy hexadecimális értékről van szó.

A lemezek particionálására használható meg a parted nevű program, vagy annak valamelyik frontend-je (gparted, qtparted).

5.5.2. A fájlrendszer létrehozás

Egy winchester esetében a partíció létrehozása önmagában még kevés. Ahhoz, hogy az adott területen adatokat tudjunk tárolni, szükségünk van arra, hogy az adott partíción valamilyen Linux által ismert fájlrendszert hozzunk létre. Más operációs rendszerek esetében a fájlrendszer létrehozást inkább formázásnak szokás hívni. Fájlrendszert nemcsak fizikai köteten lehet létrehozni, hanem bármilyen blokkos eszközön. Ilyenek lehetnek akár a szoftveres RAID által létrehozott /dev/md[01...] eszközök, vagy pedig az LVM által definiált logikai kötetek.
A fájlrendszer létrehozás általános parancsa az mkfs (make filesystem). Ahhoz, hogy a parancs különböző fájlrendszereket tudjon létrehozni, szükségünk van a megfelelő Debian csomagokra.

ext2/ext3	e2fsprogs
reiserfs (3.6)	reiserfsprogs
reiserfs (4.0)	reiser4progs
xfs	xfsprogs
jfs	jfsutils

A parancs fontosabb paraméterei:

```
mkfs [ -t fstype ] [ fs-options ] device [ blocks ]
```

Ahol:

- `-t fstype`: fájlrendszer típusa
- `fs-options`: az adott fájlrendszerre vonatkozó speciális beállítások
- `device`: az eszközfájl neve
- `blocks`: a fájlrendszernek szánt blokkok száma

Amennyiben ext2 vagy ext3 típusú fájlrendszert szeretnénk létrehozni, abban az esetben használhatjuk az mke2fs parancsot. Erre mutató hard link lesz az mkfs.ext2 és az mkfs.ext3. Mivel a két fájlrendszer belső felépítése megegyezik, ezért nincsen külön parancs a létrehozásra. A legfontosabb különbség, hogy az ext2 nem naplózó fájlrendszer.

Az mke2fs parancs fontosabb opciói:

```
mke2fs [ options ] device [ blocks ]
```

Ahol:

- `-b block-size`: blokkméret megadása
- `-c`: ellenőrzés
- `-i bytes-per-inode`: byte/i-node arány megadása
- `-j`: ext3 fájlrendszer létrehozása
- `-J`: ext3 journal opciók megadása
- `-L label`: lemezcímke megadása
- `-m number`: a rendszergazda számára fenntartott blokkok aránya (5%)
- `-N number-of-inodes`: az i-node-ok száma
- `-T fs-type` (news (4k), largefile (1M), largefile4 (4M))

Adott fájlrendszer létrehozásra használható további parancsok:

- `mkfs.reiserfs, mkreiserfs` reiserfs 3.6
- `mkfs.reiser4, mkreiser4` reiserfs 4.0
- `mkfs.xfs` xfs
- `mkfs.jfs, jfs_mkfs` jfs
- `mkfs.cramfs` cramfs
- `mkswap` swap
- `mkinitramfs` initramfs (gzippel tömörített cpio)
- `mkisofs` iso9660 fs
- `mkfs.msdos, mkfs.vfat, mkdosfs` FAT12, FAT16, FAT32
- `mkfs.ntfs, mkntfs` NTFS
- `mkfs.gfs, gfs_mkfs` GFS (Global FS)
- `mkfs.ocfs2` OCFS2 (Oracle Cluster FS)

Miután az adott fájlrendszert létrehoztuk, utána birtokba is vehetjük. Ahhoz, hogy az adott fájlrendszer használható legyen adatok tárolására (része legyen a rendszerünknek) csatlakoztatnunk kell.

5.6. A fájlrendszerek ellenőrzése és csatlakoztatása

5.6.1. A fájlrendszer ellenőrzése

Mielőtt egy fájlrendszert csatlakoztatunk ajánlott, hogy ellenőrizzük, hogy a fájlrendszer nyilvántartása rendben van-e. Ha ugyanis a fájlrendszerben valamilyen hiba keletkezett, abban az esetben a további műveletek csak tetézik a bajt. Fájlrendszer hiba bekövetkezhet például lemez, lemezvezérlő vagy memória hiba miatt. De egy szabálytalan leállás (áramszünet, tápegység meghibásodás) esetében is megtörténhet.
A fájlrendszer ellenőrzés általános parancsa az fsck (**f**ile**s**ystem **c**hec**k**).
Az fsck parancs fontosabb paraméterei:

```
fsck [ -t fstype ][ options ] device
```

Az opció közül fontos a `-a`, amely az automatikus javítást jelenti.

Az mkfs parancshoz hasonlóan léteznek olyan parancsok, amelyeknek a neve az fsck szóval kezdődik. Ezek esetében már nem kell megadni a -t kapcsoló után a fájlrendszer típusát. Ilyen például az fsck.ext2, fsck.reiserfs, ...

5.6.2. Fájlrendszer csatlakoztatása

Amennyiben a fájlrendszer ellenőrzése sikeresen végrehajtódott nyugodtan csatlakoztathatjuk a fájlrendszert. Bármilyen a kernel által támogatott fájlrendszert képesek vagyunk csatlakoztatni. Abban az esetben, hogyha az automatikus kernel modul betöltés engedélyezett, akkor a csatlakoztatás előtt nem kell a fájlrendszer típusának megfelelő kernel modult betölteni. Ellenkező esetben ezt meg kell tennünk, mivel ez szükséges feltétele a csatlakoztatásnak. Ahhoz, hogy valamilyen fájlrendszert csatlakoztassunk, szükségünk van egy létező könyvtárra. Azt a könyvtárat, ahova egy fájlrendszert csatlakoztatunk hívjuk csatolási pontnak (mount point).

A fájlrendszer csatlakoztatás általános parancsa a mount.
Az mount parancs leggyakrabban használt alakja:

```
mount [ -t fstype ] device dir [ -o options ]
```

A device jelenti az eszközt, amit csatlakoztatni kell, a dir pedig a csatlakoztatási pontot. A mount parancsnak rengeteg paramétere van. Ezen paraméterek egy része függ a fájlrendszer típusától, míg a többi általános lesz a Linux „saját” fájlrendszereire. Az utóbbi alatt minden olyan fájlrendszert értek, amely alatt működik a Linux jogosultsági rendszere és a speciális fájl típusok létrehozhatók.

A mount parancs fontosabb opció párjai (a -o után vesszővel elválasztva írandók):

- `atime, noatime`: hozzáférési idő frissítése az i-node táblában
- `auto, noauto`: automatikusan kell-e csatlakoztatni (-a opció esetében)
- `dev, nodev`: engedélyezettek-e a speciális eszközfájlok

- `exec, noexec`: engedélyezettek-e a bináris fájl végrehajtása
- `ro, rw`: csak olvasható vagy írható-olvasható mód
- `sync, async`: szinkron vagy aszinkron I/O használata
- `suid, nosuid`: setuid bit engedélyezett-e
- `user, nouser`: normál felhasználó csatlakoztathatja-e

Azért, hogy ne kelljen minden esetben felsorolni mindegyik opciót, kitalálták azt, hogy legyen egy alapértelmezés. A defaults opció az előzőekben felsoroltak közül a következőket tartalmazza: `atime, auto, dev, exec , rw, async, suid, nouser`. Abban az esetben, hogyha az alapértelmezéstől csak egy-két opcióban térünk el használhatjuk a következő módon:

- `defaults,noatime`
- `defaults,nodev,nosuid`

Az előzőekben felsorolt párok mellett a mount esetében az alábbi opciókat is használhatjuk:

- `bind`: egy adott könyvtár más helyről történő elérése (a device paraméter helyett ebben az esetben egy létező könyvtárat kell megadni)
- `loop`: egy fájl tartalmának csatlakoztatása egy loopback device segítségével (pl.: iso fájl tartalmának elérése)
- `remount`: egy már korábban csatlakoztatott fájlrendszer újra csatlakoztatása
- `grpquota, usrquota`: lemezkvóta beállításához

Az alábbi kapcsolók is lehetségesek:

- `-t`: fájlrendszer típusának megadása
- `-a`: automatikus csatlakoztatás a /etc/fstab alapján
- `-n`: nem próbálja meg írni a /etc/mtab-ot
- `-r`: megegyezik a `-o ro` opcióval
- `-w`: megegyezik a `-o rw` opcióval

Példák mount használatára:

- `mount -n -o remount,rw /`
- `mount -t reiserfs /dev/sdb1 /boot`
- `mount -t iso9660 debian.iso /mnt/iso -o loop,ro`

Egyes fájlrendszerek esetében további opciók is lehetségesek:

- `mount -t ext3 /dev/sdb2 /home -o rw,user_xattr,acl`
- `mount -t tmpfs tmpfs -o size=100M /var/tmp`
- `mount -t jfs /dev/sdb3 /mnt/test -o remount,rw,resize`

Miután egy fájlrendszert sikeresen csatlakoztatunk, utána képesek vagyunk rajta adatokat tárolni. Figyeljünk arra, hogy a naplózó fájlrendszerek esetében amennyiben nem külső naplót használunk, az csökkenteni fogja a fájlrendszer méretét.

Amikor egy rendszert telepítünk azért is fontos lehet az egyes könyvtárakat külön helyre tenni, mert abban az esetben van arra lehetőség, hogy a rendszer egyes részeit különböző mount opciókkal csatlakoztassuk.

Erre példa a következő felsorolás:

- `/ defaults`
- `/boot rw,nodev,noexec,nosuid`
- `/home rw,nodev,noexec,nosuid`
- `/tmp rw,noatime,nodev,noexec,nosuid`
- `/usr rw,nodev`
- `/var rw,noatime,nodev,noexec,nosuid`

Amennyiben valamelyik csatlakoztatott fájlrendszerre már nincsen szükségünk, abban az esetben leválaszthatjuk. A leválasztás parancsa az umount. A mount kapcsolói közül itt is használható a -a, -n illetve a -t.
Fontos dolog, hogy amikor a rendszer indul elsőként történik meg a root fájlrendszer csatlakoztatása, majd ezután jöhetnek az egyéb fájlrendszerek. Ahhoz, hogy valamilyen fájlrendszert le tudjunk csatolni elsőként gondoskodnunk kell arról, hogy az adott könyvtárat ahova a fájlrendszer csatlakoztatva van semmilyen alkalmazás ne használja. Tehát elsőként azokat az alkalmazásokat kell leállítani, amelyek a csatlakoztatási pont alatt valamilyen fájlt használnak. Csak ezután következhet be a leválasztás. A rendszerleállításkor a root fájlrendszer leválasztása történik meg legutoljára.
Példa umount használatára:

- `umount -a -t reiserfs`

Jellegét tekintve a swap partíció vagy swap file használata kapcsolódik a mount parancshoz. Annyi a különbség, hogy ennek a tartalmát nem szoktuk közvetlenül olvasni. Vagyis itt csak a használatot kapcsoljuk be illetve ki.

```
swapon /dev/sdb4
swapoff /dev/sdb4
```

Abban az esetben, hogyha nem áll rendelkezésre szabad partíció, de van valahol szabad helyünk, akkor foglalhatunk helyet tárcsere céljára. Ahhoz, hogy egy fájlt képesek legyünk swap területként használni fontos, hogy folyamatosan legyenek az egyes blokkjai tárolva a diszken. Példa 500MB-os swapfile létrehozására és használatba vételére:

- `dd if=/dev/zero of=swapfile bs=1M count=500`
- `mkswap swapfile`
- `sync`
- `swapon swapfile`

5.7. A fájlrendszerek átméretezése

Amikor egy fájlrendszert létrehozunk, akkor az egy rögzített mérettel jön létre. Általában a fájlrendszer teljes mértékben kitölt egy partíciót vagy logikai kötetet. A LVM esetében láttuk, hogy logikai kötetek átméretezhetők, méretük parancsok segítségével növelhető és csökkenthető.

5.7.1. Fájlrendszer méretének a növelése

Amennyiben egy fájlrendszer méretét szeretnénk növelni, abban az esetben elsőként a fizikai vagy logikai kötet méretét kell növelni. Az LVM esetében ez nem okoz problémát, viszont a partíciók esetében igen. Ahhoz, hogy egy winchester partíciós tábláját módosítani tudjuk, elsőként le kell csatolni az összes

partíciót. Ez ugyanis feltétele annak, hogy a kernel módosítás után újra tudja olvasni a partíciós táblát. Ezután le tudjuk törölni a partíciót, majd a helyére egy ugyanolyan típusú, csak nagyobb méretűt hozhatunk létre. Nagyon fontos, hogy az új partíció kezdete megegyezzen a korábbi partíció kezdetével. Ezután kezdődhet a fájlrendszer méretének növelése.

Az egyes fájlrendszerek esetében ez a különböző parancsok segítségével történhet. Tegyük fel, hogy a /dev/sdb2 partíció méretét növeltük meg, és most a rajta lévő fájlrendszer méretét szeretnénk megnövelni:

ext2, ext3 `resize2fs /dev/sdb2`
reiserfs 3.6 `resize_reiserfs /dev/sdb2`
xfs `xfs_growfs /mnt/sdb2` (a csatolási pont a paraméter)
jfs `mount -t jfs /dev/sdb2 /mnt/sdb2 -o remount,rw,resize`

Mindegyik parancs a partíció méretéig megnöveli a fájlrendszer méretét. Amennyiben nem ez a célunk paraméterek segítségével megadható a növelés mértéke.

5.7.2. Fájlrendszer méretének a csökkentése

Amennyiben helyet szeretnénk felszabadítani egy diszken vagy kötetcsoporton belül, úgy szükségünk van a partíció vagy logikai kötet méretének a csökkentésére. Csak akkora méretcsökkentést tehetek, hogy az aktuálisan tárolt adatok a csökkentett fájlrendszerben is elférjenek. A csökkentés esetében fordított sorrendben kell eljárni, mint azt a növeléskor tettük. Vagyis elsőként kell a fájlrendszert, majd pedig a tároló kötetet csökkenteni.
Az xfs és a jfs fájlrendszerek esetében nincsen arra lehetőség, hogy működés közben, vagy offline módon csökkentsük a fájlrendszer méretét.
Tegyük fel, hogy van egy 500MB-os partiónk (/dev/sdb1). Szeretnénk a méretét csökkenteni kb. 400MB-ra:
A ext2, ext3 esetében a `resize2fs /dev/sdb2 [size]` parancs segítségével lehet a fájlrendszer méretét csökkenteni akár működés közben is. A `[size]` paramétereként meg kell adnunk a blokkméretet, amire akarjuk a fájlrendszer csökkenteni. 1kB-os blokkméretet feltételezve 400000 körüli értéket kell itt megadnunk.
A reiserfs (3.6) estében csak akkor tehetjük meg az átméretezést, hogyha a partíció nem aktív. Ebben az esetben a `resize_reiserfs -s 100M /dev/sdb2` parancs segítségével a feladat kivitelezhető.

5.8. A fájlrendszerekhez kapcsolódó konfigurációs fájlok

Amikor a Linux indul, akkor általában a kernelnek parancssorban átadásra kerül, hogy melyik eszköz, melyik partícióját csatlakoztassa fel, mint root fájlrendszer. A GRUB esetében ilyen beállítások például:

```
kernel /boot/vmlinuz root=/dev/sda2
kernel /vmlinuz root=/dev/mapper/vgmain-lvroot
```

Az adott eszköz esetében szükségünk van egy konfigurációs fájlra, amely a további fájlrendszer csatolási paramétereket adja meg. Ez a fájl a /etc/fstab.

Ebben beállíthatjuk, hogy hova és milyen csatlakoztatási paraméterekkel legyenek az egyes fájlrendszerek csatlakoztatva. Természetesen a sikeres rendszerindulás feltétele, hogy kernel azt a fájlrendszert, amelyet a rendszerbetöltő esetében megkap mint root fájlrendszert, azt el tudja érni és a rajta lévő fájlrendszer típusát ismerje. Amennyiben ez a feltétel teljesül, akkor a /etc/fstab alapján folytatódhat a fájlrendszerek automatikus csatlakoztatása.

Példa /etc/fstab-ra:

```
# /etc/fstab: static file system information.
#
proc                    /proc    proc         defaults            0  0
/dev/md0                /        reiserfs     defaults,acl        0  1
/dev/sda1               /boot    ext3         defaults            0  2
/dev/md1                /home    reiserfs     rw,acl,nosuid,nodev 0  2
/dev/mapper/vg-lvvar    /var     reiserfs     defaults            0  2
/dev/scd0               /cdrom   udf,iso9660  user,noauto         0  0
/dev/fd0                /floppy  auto         rw,user,noauto      0  0
```

A konfigurációs fájl esetében megkülönböztethetünk hat mezőt. A fájl egy sora egy fájlrendszer csatlakoztatásához szükséges paramétereket állítja be. A mezők jelentése balról jobbra haladva:

- 1. mező: a speciális eszköz vagy távoli fájlrendszer, amit csatlakoztatni kell
- 2. mező: a csatlakoztatási pont
- 3. mező: a fájlrendszer típusa
- 4. mező: a fájlrendszer csatlakoztatási módja (mount parancs -o kapcsolója esetében használható opciók)
- 5. mező: kell-e biztonsági mentés (dump)
- 6. mező: kell-e ellenőrzést futtatni

Amennyiben a /etc/fstab nevű fájl adott sorában össze van rendelve egy csatlakoztatási pont egy eszközzel, illetve fájlrendszerekkel és azok csatlakoztatási paramétereivel abban az esetben a mount parancs kevesebb paraméterrel is használható. Ekkor csak az eszköz nevét vagy a csatlakoztatási pontot kell megadni a mount parancs után.
Példák mount parancs használatára az előző példában szereplő /etc/fstab nevű fájlt alapul véve:

- `mount /cdrom`
- `mount /dev/md1`

Hasonló módon az umount parancs is használható. Amennyiben a mount parancsot paraméterek nélkül használjuk, úgy megmutatja a rendszer jelenleg csatlakoztatott eszközeit, illetve a csatlakoztatás paramétereit.

Példa mount parancs kimenetére:

```
/dev/md0 on / type reiserfs (rw,acl)
tmpfs on /lib/init/rw type tmpfs (rw,nosuid,mode=0755)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
procbususb on /proc/bus/usb type usbfs (rw)
udev on /dev type tmpfs (rw,mode=0755)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=620)
/dev/sda1 on /boot type ext3 (rw)
/dev/md1 on /home type reiserfs (rw,nosuid,nodev,acl)
/dev/mapper/vg-lvvar on /var type reiserfs (rw)
binfmt_misc on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
```

A felcsatlakoztatott fájlrendszerek jellemzői megtekinthetők a /etc/mtab vagy a /proc/mounts segítségével is.

6. Saját kernel fordítása

6.1. Előkészületek a kernel fordításához

Az „Operációs rendszerek – Linux” jegyzetben a 7. fejezet a kernelről, illetve a általánosságban a kernel fordításról szól. Az olvasóról feltételezem, hogy az ott leírtakat ismeri, vagy más forrásból rendelkezik megfelelő szinttű ismeretekkel. Ebben a fejezetben részletesen megismerkedünk a kernel lehetőségeivel, illetve megnézzük azokat a beállításokat, drivereket amelyek kapcsolódnak a számítógépes laborokban használt VMware virtuális gépekhez.

6.1.1. Mikor szükséges kernelt fordítani?

Általános szabályt nem túl egyszerű mondani. Néhány dolog, amely indokolhatja a saját kernel előállítását:

- a disztribúció készítői által kiadott kernel még nem támogatja az adott hardver eszközt
- a használt driverek közül valamelyikben biztonsági vagy működési hiba van
- az adott szolgáltatás, protokoll, hálózati támogatás nem lett belefordítva a kernelbe
- saját, speciális működési környezet, amely az „általános célú” kernellel nem működne (pl. hálózati rendszerindulás, Boot CD, ...)
- valamilyen kiegészítő patch-e(ke)t szeretnénk használni (pl.: grsecurity)
- nem akarjuk, hogy más is ismerje a kernelünk beállításait

Ha az előzőek közül több dolog is fennáll, akkor valószínűleg szükségünk van saját kernel fordítására. A saját kernel előnyei:

- többféle hardvereszközt támogathat
- optimalizálni lehet a hardverhez (pl. CPU utasításkészlet)
- kevesebb memória használat
- kevesebb driver, vélhetően kisebb hibalehetőség
- gyorsabb működés
- az újabb kernelben javítva lettek bizonyos biztonsági hibák

Természetesen az operációs rendszer magjának a fordítása az egész rendszer működésére komoly hatást gyakorol. A dokumentációk átolvasása után nyugodtan nekifoghatunk. Néhány fontos dolog, amelyet mindig szem előtt kell tartanunk:

- a kernelnek el kell tudnia érnie azt hardver eszközt, illetve azt a fájlrendszert, ahol a root fájlrendszer található. (sikeres rendszerindulás feltétele)
- a konfigurálás előtt el kell döntenünk, hogy a rendszerindulást initrd (initial RAM disk) segítségével, vagy nélküle szeretnénk-e elvégezni
- azokról a támogatásokról, amelyekről nem tudjuk pontosan, hogy mi, de alapértelmezésként ajánlja a belefordítását, fogadjuk meg a tanácsát

6.1.2. A kernel fordítás szükségletei

Ahhoz, hogy kernelt tudjunk kicsomagolni, foltozni és fordítani Debian GNU/Linux rendszerben, szükségünk van néhány csomagra:

- libc6-dev általános C header fájlok
- libncurses-dev a make menuconfig használatához
- bzip2 bzip2 tömörítő
- patch javítások, módosítások futtatása
- binutils bináris segédprogramok (pl.: ld, as, ...)
- gcc, cpp GNU C fordító, C++ előfeldolgozó
- make GNU make a fordítási folyamat irányításához

Ezek az előfeltételek az

```
apt-get install gcc make patch bzip2 libncurses-dev
```

parancs hatására települnek a függőségi viszonyoknak köszönhetően. Az előző parancsban felsoroltak tehát minden esetben szükségesek. Ezek mellett a kernel fordítás használhat egyéb programokat is, melyek közül néhány alapértelmezésként telepítve is van a rendszerben.
A 2.6.31.9-as kernel esetében a fordítási függőségek a következők:

5. sz. táblázat

Program (csomag)	***Verzió***	***Ellenőrzés parancsa***
GNU C (gcc)	3.2	gcc --version
GNU make	3.8	make --version
binutils	2.12	ld -v
util-linux	*2.10o*	*fdformat --version*
module-init-tools	0.9.10	depmod -V
e2fsprogs	1.41.4	tune2fs
jfsutils	1.1.3	fsck.jfs -V
reiserfsprogs	3.6.3	reiserfsck -V 2>&1| grep reiserfsprogs
xfsprogs	2.6.0	xfs_db -V
pcmciautils	*004*	*pccardctl -V*
quota-tools	*3.09*	*quota -V*
PPP	*2.4.0*	*pppd --version*
isdn4k-utils	*3.1pre1*	*isdnctrl 2>&1|grep version*
nfs-utils	*1.0.5*	*showmount --version*
procps	3.2	ps --version
oprofile	*0.9*	*oprofiled --version*
udev	081	udevinfo -V
grub	0.93	grub –version

Az előző táblázatban szereplő csomagok egy része már a rendszer telepítését követően elérhető. Ilyenek például a fájlrendszerek kezeléséhez kötődő csomagok. Míg a listában szereplő egyéb csomagok csak speciális opciók

választása esetén szükségesek. Az opcionális csomagokat dőlt betűtípussal szerepelnek a táblázatban.

6.1.3. A kernel forrás előkészítése és foltozása

A továbbiakban a 2.6.31.9-es verziószámú vanilla kernel forráskódjának előkészítését és konfigurálását fogom bemutatni. A kernelről tudni kell, hogy állandóan változik, fejlődik. Az egyes verziók esetében újabb és újabb lehetőségek jelennek meg vagy avulnak el. Némely esetben az is előfordul, hogy a támogatások helye változik, tehát máshol találjuk az adott beállítást, mint a korábbi verziók esetében.

Az előkészítés, letöltés és ellenőrzés lépései:

- `cd /usr/src`
- `mv linux `uname -r``
- `wget \`

```
  kernel.org/pub/linux/kernel/v2.6/linux-2.6.31.tar.bz2 \
ftp://ftp.hu.kernel.org/pub/linux/kernel/v2.6/linux-2.6.31.tar.bz2.sign \
ftp://ftp.hu.kernel.org/pub/linux/kernel/v2.6/patch-2.6.31.9.tar.bz2 \
ftp://ftp.hu.kernel.org/pub/linux/kernel/v2.6/patch-2.6.31.9.tar.bz2.sign
```

Miután a forrást, illetve a verzió javító patch-et és a digitális aláírásukat letöltöttük, ajánlatos ellenőrizni, hogy rendben vannak-e.
Ehhez a következőket kell tennünk:

- `gpg --keyserver pgp.mit.edu --recv-keys 0x517D0F0E`

Le kell töltenünk valamelyik publikus gpg kulcsszerverről az 0x517D0F0E azonosítóhoz tartozó publikus kulcsot, amely az <ftpadmin@kernel.org>-hoz tartozik.
Majd következhet az ellenőrzés:

- `gpg --verify linux-2.6.24.tar.bz2.sign linux-2.6.24.tar.bz2`
- `gpg --verify patch-2.6.24.3.bz2.sign patch-2.6.24.3.bz2`

Amennyiben rendben van ezt látjuk:

```
gpg: Signature made 2009. szept. 10., csütörtök, 00.40.47 CEST using DSA key ID
517D0F0E
gpg: Good signature from "Linux Kernel Archives Verification Key
<ftpadmin@kernel.org>"
```

Ezután következhet a forráskód kicsomagolása és a /usr/src/linux beállítása:

- `tar xjf linux-2.6.31.tar.bz2`
- `ln -s linux-2.6.31 linux`
- `cd /usr/src/linux`

Amennyiben 2.6.31-es kernelt szeretnénk fordítani, úgy a forráskód már elő lett készítve. Ezután következhet igény szerint a foltozás (patch-elés).
Amennyiben a 2.6.31-es forrása ki van csomagolva úgy egy patch segítségével növelhetjük a verzióját 2.6.31.9-re.

- `bzcat ../patch-2.6.31.9.bz2 | patch -p1`

Ezután az adott verziószámhoz megfelelő egyéb foltokat is használhatunk:

- `patch -p1 < ../grsecurity-2.1.14-2.6.31.9-200912191011.patch`

Egy folt alkalmazásakor minden esetben a patch parancs standard bemenetére

kell a „szerkesztési parancsokat” átadni. A -p paraméter után megadott számot az határozza meg, hogy hány darab könyvtárat kell törölni a fájlhivatkozások elejéről.
A grsecurity patch első sora például:

```
diff -urNp linux-2.6.31.9/arch/alpha/include/asm/atomic.h ...
```

Mivel mi a /usr/src/linux-2.6.31 könyvtárban vagyunk, ezért a hivatkozás elejéről a linux-2.6.31.9 könyvtárat el kell távolítani. Ezért kell a -p1, vagyis egy könyvtárat kell törölnünk a fájl hivatkozásból. Az előző grsecurity patch-et alkalmazhattuk volna a következőképpen is:

- `cd /usr/src`
- `ln -s linux-2.6.31 linux-2.6.31.9`
- `patch -p0 < grsecurity-2.1.14-2.6.31.9-200912191011.patch`

Ebben az esetben semmilyen fájlhivatkozást nem kell átalakítani. De fontos létrehozni a linux-2.6.31.9 könyvtárat (szimbolikus link is elég), mivel a patch ezen könyvtáron belüli fájlokat akar majd módosítani.

6.2. A hardver eszközök és szoftver szükségletek vizsgálata

Miután a forráskód elő lett készítve fontos dolog, hogy tisztában legyünk a rendszerünk feladatával és a hardver eszközök paramétereivel. Tehát figyelembe kell vennünk a következőket:

- **hardver paraméterek**: tudnunk kell, hogy a rendelkezésre álló hardver milyen képességekkel rendelkezik és ehhez milyen eszközmeghajtók szükségesek
- figyelembe kell venni a **telepítési szituációt**: attól függően, hogy milyen fájlrendszereket, illetve milyen egyéb diszk kezelési módszereket használunk (RAID, LVM, ...) különböző fájlrendszer támogatások szükségesek
- a használni kívánt protokollok, eljárások támogatását is előre végig kell gondolni
- a gép védelme érdekében a kernel szintű tűzfal használni kívánt szűrési lehetőségei
- szeretnénk-e initramfs/initrd-t használni a rendszerindulás során

Amikor az operációs rendszert telepítettük, akkor a disztribúció készítői egy általános kernelt adtak. Amely a legtöbb szituációban működik. A saját kernel esetében nekünk csak egy szituációt kell figyelembe vennünk, hogy a mi esetünkben el tudja látni a megfelelő funkciókat.
A továbbiakban az ingyen elérhető VMware Server által emulált virtuális gép paramétereit fogom bemutatni. Vagyis számunkra a továbbiakban ez meghatározza, hogy az egyes driverek közül mire is lesz szükségünk. A kernel konfiguráció során, amelyről a következő alfejezet szól, van arra lehetőségünk, hogy az összes lehetőség közül kiválasszuk a számunkra fontosakat.

6.2.1. A VMware virtuális gép hardver paraméterei

Amennyiben rendelkezünk egy működő operációs rendszerrel, úgy a hardver eszközök feltérképezésre esetében segítségül hívhatjuk az lspci parancsot. Amennyiben a parancs esetében léteznek ismeretlen eszközök, akkor ajánlott a

helyi adatbázis frissítése. Az lspci parancs ugyanis kiolvassa az egyes eszközök PCI azonosítóját, és ezek alapján egy saját adatbázisból megpróbál hozzájuk egy eszközt rendelni. Az update-pciids paranccsal frissíthetjük a szóban forgó adatbázist.
Példa lspci parancs kimenetére:

```
00:00.0 Host bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX Host
bridge (rev 01)
00:01.0 PCI bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX AGP
bridge (rev 01)
00:07.0 ISA bridge: Intel Corporation 82371AB/EB/MB PIIX4 ISA (rev 08)
00:07.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:07.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:0f.0 VGA compatible controller: VMware Inc [VMware SVGA II] PCI Display
Adapter
00:10.0 SCSI storage controller: LSI Logic / Symbios Logic 53c1030 PCI-X
Fusion-MPT Dual Ultra320 SCSI (rev 01)
00:11.0 Ethernet controller: Advanced Micro Devices [AMD] 79c970 [PCnet32
LANCE] (rev 10)
```

A VMware virtuális gép fontosabb hardverei: 6. sz. táblázat

Hardver	***Típus***
CPU	a hoszt gép processzora
Memória	a virtuális gép paraméterei esetében beállított
Alaplapi chipset	Intel PIIX4
Videókártya	VMware SVGA II PCI
Diszk vezérlő	SCSI: lsilogic vagy buslogic típusú IDE: alaplapi PIIX4 típusú vezérlő
Diszk	a létrehozáskor beállított típusú és méretű
CDROM	IDE vagy SCSI típusú (csak IDE-ről tud bootolni), ISO image használható CD-ként
Egér	PS/2-es
Soros, Párhuzamos, USB portok	a hoszt gép megfelelő portjai
Floppy	a hoszt gép floppy eszköze, vagy image fájl

Az előzőekben felsorolt eszközök közül a kernel működése szempontjából a legkritikusabbak: a CPU utasításkészlet, a diszk vezérlő és diszk típusok támogatása

A géptermi laborgyakorlatok esetében használt virtuális gépek jellemzői, és a működésükhöz szükséges drivereket mutatja be a következő táblázat. Csak a virtuális gép azon komponenseit tartalmazza, amelyekre vonatkozó információk feltétlenül szükségesek a kernel konfigurálásához:

A VMware virtuális gép működéshez szükséges kernel driverek: 7. sz. táblázat

Hardver	***Kernel driver***
Pentium 4 CPU HT támogatással	Processor family (Pentium-4/Celeron(P4-based)/Pentium-4 M/older Xeon)
512MB Memória	alapértelmezésként támogatott
Intel PIIX4	Intel PIIX/ICH chipsets support
VMware SVGA II	VGA text console karakteres módban (X alatt saját driver vmware típusú csatolóhoz)
SCSI: lsilogic diszk vezérlő	Fusion MPT ScsiHost drivers for SPI
SCSI: buslogic diszk vezérlő	BusLogic SCSI support
IDE alaplapi PIIX4 típusú vezérlő	Intel PIIX/ICH chipsets support
SCSI diszk	SCSI disk support
IDE diszk	ATA/ATAPI/MFM/RLL support + generic ATA/ATAPI disk support + ATA disk support
SCSI CDROM	SCSI CDROM support
IDE CDROM	ATA/ATAPI/MFM/RLL support + Include IDE/ATAPI CDROM support
Floppy	Normal floppy disk support

A kernel konfigurálása című alfejezetből kiderül, hogy az előző táblázatban felsorolt támogatások pontosan mely menüpontok alatt lesznek elérhetőek. A következő fejezetben bemutatom a kernel fontosabb opcióit. Ebben az esetben egy olyan VMware virtuális gépet veszek alapul, amely lsilogic típusú SCSI diszk vezérlőt és SCSI diszket használ, csak ext3 fájlrendszerrel, és nem kell a rendszerinduláshoz initramfs/initrd image.

6.3. A kernel konfigurálása

6.3.1. A konfigurálás lehetőségei és a használt jelölési módok

A kernelfordítás egy szöveges konfigurációs fájl alapján történik. Ebben a fájlban meghatározott nevek segítségével lehetséges adott támogatásokat ki-be kapcsolni.
Minta a konfigurációs fájl soraira:

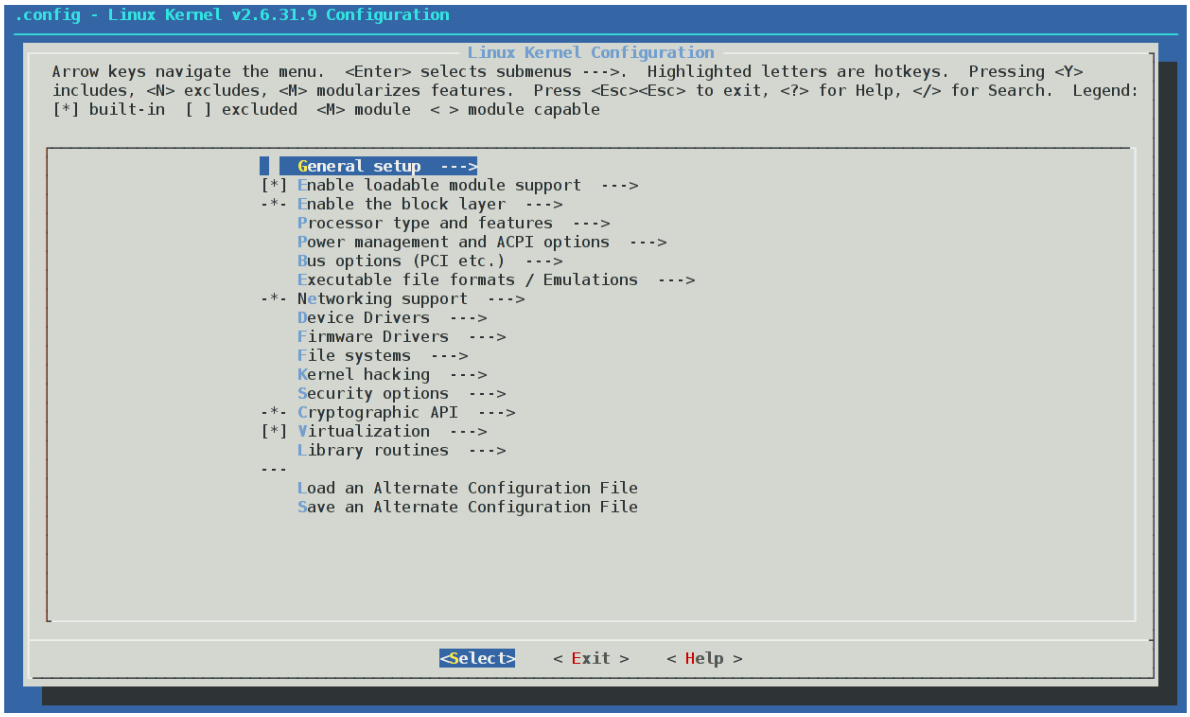
```
CONFIG_NET=y
CONFIG_PACKET=m
# CONFIG_IP_ADVANCED_ROUTER is not set
```

A konfigurációs fájl könnyebb elkészítésére találták ki a következő lehetőségeket:

```
make config             #sor orientált módon
make menuconfig         #ncurses alapú felület
make xconfig            #Qt alapú felület
```

make gconfig #Gtk alapú felület

Az utolsó kettő futtatásához tehát X grafikus felület feltétlenül szükséges. A parancsokat a kernel forrást tartalmazó könyvtárban kell kiadni.



A „make menuconfig” főmenüje 2. sz. ábra

A „make menuconfig” jelölései:

- A [ ] két választási lehetőséget jelöl
  - [*] a támogatást be kell építeni a kernel image-be
  - [ ] adott támogatás nem szükséges
- A < > három választási lehetőséget jelöl
  - <*> a támogatást be kell építeni a kernel image-be
  - <M> a támogatás külső modulként legyen
  - < > adott támogatás nem szükséges
- A { } azt jelzi, hogy függőség miatt kiválasztva, de választhatunk két lehetőség közül
  - {*} a támogatást be kell építeni a kernel image-be
  - {M} a támogatás külső modulként legyen
- A -*- jelzi, hogy a támogatás függőség miatt automatikusan kiválasztva és nincsen más választási lehetőségünk
- A normál zárójel esetében valamilyen sztringet vagy számot adhatunk meg.

A „make menuconfig” fontosabb billentyűparancsai:
- kurzor mozgató billentyűkkel navigáció
- ENTER adott almenüpont kiválasztása

- kettő ESC: egy szinttel feljebb a menük között
- Y beépíteni valamit
- M modulként használni valamit
- N nem szükséges a támogatás
- SPACE Y => N => M => Y ... a választás módosítása
- ? az adott menüponthoz tartozó leírás megmutatása

6.3.2. A konfigurálás menüpontok szerint

A továbbiakban a „make menuconfig” főmenüje alapján fogom a kernel egyes funkcióit értelmezni. A kernelnek nagyon sok lehetősége van, amely ismertetése meghaladja ezen mű terjedelmi lehetőségeit. Azokra az opciókra fogok koncentrálni, amelyek az alap működéshez kellenek. Ezen felül igyekszem utalásokat tenni, amellyel segítem az eligazodást.

General setup ---> (általános beállítások)

```
[*] Prompt for development and/or incomplete code/drivers
```

Mutassa az egyes kódok „stabilitási állapotát”:

- NEW új
- DEPRECATED elavult
- EXPERIMENTAL kísérleti jellegű

```
(-noinitrd) Local version - append to kernel release
[*] Automatically append version information to the version string
```

Saját verzió hozzáfűzése a verzió sztringhez

```
[*] Support for paging of anonymous memory (swap)
```

Virtuális memória (swap) támogatása

```
[*] System V IPC
```

SVR rendszerektől örökölt folyamatok közti adatcsere

```
[*] POSIX Message Queues
```

POSIX típusú üzenet sor, az IPC egy speciális formája

```
[*] BSD Process Accounting
```

Folyamatokról szóló információk tárolása

```
[ ] Auditing support
```

A kernel működésének vizsgálata, ellenőrzése

```
< > Kernel .config support
```

Saját konfiguráció tárolása

```
(15) Kernel log buffer size (16 => 64KB, 17 => 128KB)
```

Kernel napló puffer mérete

```
[ ] Group CPU scheduler
```

Folyamatok csoportosítása, csoportként való CPU allokáció

```
[ ] Kernel->user space relay support (formerly relayfs)
```

Nagy mennyiségű adat mozgatása kernel és felhasználói tér között

```
[ ] Initial RAM filesystem and RAM disk (initramfs/initrd) support
```

Kezdeti ramdiszk támogatás, a kernel a valódi root fájlrendszer beállítása előtt ezt használja mint kezdeti root fájlrendszer. Innét kernel modulok tölthetők be illetve alkalmazások indíthatók el igény szerint.

```
[*] Optimize for size
```

A kernel méretre való optimalizálása

```
[ ] Configure standard kernel features (for small systems)  --->
```

Beágyazott rendszerekben használható kernelt készít. Jó néhány választási opciót eleve letilt.

```
Choose SLAB allocator (SLAB)  --->
```

Gyakran használt fájl objektumok cache-elésének módszere

[*] Enable loadable module support ---> (betölthető modul támogatás)

```
[*]    Forced module loading
```

Kernel modul betöltésének „kierőszakolása”

```
[*]   Module unloading
[*]     Forced module unloading
```

Kernelmodulok eltávolítása normál illetve „kierőszakolt” módon.

```
[*]   Module versioning support
```

Modulok verziószámozásának támogatása

```
[ ]   Source checksum for all modules
```

A forrásra utaló ellenőrző összeg használata.

[-*- Enable the block layer ---> (blokkos eszközök támogatása)

```
[ ]   Support for Large (2TB+) block devices and files
```

Nagy méretű blokkos eszközök és fájlok támogatása >= 2TB (max. 16TB 32bites rendszerben)

```
IO Schedulers  --->
<*> Anticipatory I/O scheduler
<*> Deadline I/O scheduler
<*> CFQ I/O scheduler
Default I/O scheduler (CFQ)  --->
```

Az I/O műveletek esetében használatos ütemező (elevator) támogatása.

Processor type and features ---> (processzor típusok és jellemzőik)

```
[ ] Tickless System (Dynamic Ticks)
```

Az időzítő nem folyamatosan jár, hanem csak igény szerint.

```
[ ] High Resolution Timer Support
```

Nagy pontosságú időzítő támogatás.

```
[*] Symmetric multi-processing support
```

Több processzoros (SMP) rendszerek támogatása

```
[ ] Paravirtualized guest support  --->
```

Paravirtualizáció támogatása (pl.: Xen)

```
Processor family (Pentium-4/Celeron(P4-based)/Pentium-4 M/older Xeon)
```

Processzor család kiválasztása. Itt választhatunk 386-tól felfelé, a CPU-nk típusa szerint. A mintában szereplő Pentium-4 család mellett elérhető az Opteron/Athlon64/Hammer/K8 és a Core 2/newer Xeon támogatása is.

```
[ ] Generic x86 support
```

Általános x86 optimalizálás az utasításkészlet mellett.

```
[ ] HPET Timer Support
```

Nagy pontosságú időzítő támogatása

```
(2) Maximum number of CPUs (2-255)
```

A processzorok maximális száma

```
[*] SMT (Hyperthreading) scheduler support
```

A Hyperthreading-et figyelembe vevő CPU ütemező.

```
[ ] Multi-core scheduler support
```

Több magot figyelembe vevő CPU ütemező.

```
Preemption Model (No Forced Preemption (Server))
```
A kernel szálak preemptívek legyenek-e. Egy szerver esetében nem kell, mivel nem kell gyorsan „felhasználói eseményekre” reagálni.
```
[ ] Machine Check Exception
```
A CPU jelzi a kernelnek például ha valamilyen melegedési hiba van.
```
< > /dev/cpu/microcode - Intel IA32 CPU microcode support
```
CPU microcode frissítésének lehetősége.
```
< > /dev/cpu/*/msr - Model-specific register support
```
Modellfüggő regiszterek támogatása speciális eszközökön keresztül.
```
< > /dev/cpu/*/cpuid - CPU information support
```
CPU típusának azonosítása
```
High Memory Support (4GB)  --->
```
„Felső memória” támogatása. Ezáltal nemcsak az 1G alatti rész, hanem maximum 4 vagy 64GB memória is elérhető lesz.
```
Memory model (Flat Memory)  --->
```
Memória belső kezelésének modellje
```
[*] MTRR (Memory Type Range Register) support
```
A processzor memória hozzáféréseinek vezérlésére. Grafikus környezet esetében ajánlott használni, mert jelentős sebesség növekedést is eredményezhet.
```
 Timer frequency (250 HZ)  --->
```
Az időzítő frekvenciája.

Power management options ---> (teljesítmény menedzsment)

```
[*] Power Management support
```
A teljesítmény menedzsment támogatása
```
[*] Suspend to RAM and standby
```
A gép „elaltatása” (ACPI S3 állapot)
```
[ ] Hibernation (aka 'suspend to disk'
```
A gép állapotának mentése (hibernálása) adott partícióra
```
[*] ACPI (Advanced Configuration and Power Interface) Support  --->
```
Az ACPI funkciók támogatása (CPU hőmérséklet, elemek állapota, AC adapter érzékelése, bekapcsoló gomb használata, ...
```
< > APM (Advanced Power Management) BIOS support  --->
```
Az APM BIOS támogatása. Az újabb gépek tudják az ACPI-t.
```
CPU Frequency scaling  --->
```
CPU sebességének változtatása „meghatározott szabály” alapján.
performance: maximális teljesítmény, powersave: teljesítmény megőrzése, userspace: felhasználói térből állítható, ondemand: igény szerinti, conservative: hasonlít az ondemand-ra annyi a különbség, hogy lépcsőnként növekszik a sebesség, nem pedig egyszerre.

Bus options (PCI etc.) ---> (sín rendszerek támogatása)

```
[*] PCI support
```
PCI busz támogatása
```
[ ] PCI Express support
```
PCI Express busz támogatása
```
[ ] ISA support
```
AT busz támogatása
```
< > PCCard (PCMCIA/CardBus) support
```
PCMCIA/CardBus támogatása

Executable file formats / Emulations ---> (végrehajtható formátumok)

```
[*] Kernel support for ELF binaries
```
Az ELF típusú binárisok támogatása. Feltétlenül kell ahhoz, hogy a kernel alkalmazást tudjon indítani.
```
< > Kernel support for a.out and ECOFF binaries
```
Régebbi bináris formátumok támogatása.
```
< > Kernel support for MISC binaries
```
Egyéb binárisok (Java, .NET, ...) futtatási lehetősége.

Networking support ---> (hálózati beállítások)

```
Networking options  --->
```
Hálózati opciók
```
<M> Packet socket
```
A hálózati eszköz és kernel közvetlen kommunikációjához kell.
```
[*]   Packet socket: mmapped IO
```
A gyorsabb kommunikáció érdekében memory mapped I/O.
```
 <M> Unix domain sockets
```
Standard módszer a hálózat elérésére Unix operációs rendszerben, az IPC egy speciális formája.
```
[*] TCP/IP networking
```
TCP/IP alapú hálózati kommunikáció
```
 [ ]   IP: multicasting
```
IP alapú csoportos küldés támogatása
```
 [ ]   IP: advanced router
```
IP alapú útválasztó funkciók
```
 [ ]   IP: kernel level autoconfiguration
```
Lehetővé teszi a kernel indulásakor az IP paraméterek beállítását. Lemez nélküli munkaállomások esetében kiválóan használható.
```
< >   IP: tunneling
```
IP alapú alagút típusú átvitel
```
[*]   IP: TCP syncookie support (disabled per default)
```
A SYN elárasztástól véd, ami a DoS típusú támadások egyik fajtája.
```
<M>   INET: socket monitoring interface
```
Hálózati forgalom megfigyelésére való programok használják.
```
[ ]   TCP: advanced congestion control  --->
```
TCP torlódás vezérlő algoritmusok
```
< >   The IPv6 protocol
```
IP version 6 protokoll
```
[ ] Network packet filtering framework (Netfilter)  --->
```
Hálózati csomagszűrő tűzfal beállításai.
```
< > 802.1d Ethernet Bridging
```
Ethernet hálózati híd
```
< > 802.1Q VLAN Support
```
Virtuális LAN-ok támogatása
```
 [ ] QoS and/or fair queueing  --->
```
Quality of Service, hálózati csomagok küldési sorrendjének szabályozása

```
[ ]   Amateur Radio support  --->
```
AX.25 rádiós protokoll támogatása
```
< >   IrDA (infrared) subsystem support  --->
```
Infrared kommunikáció támogatása
```
 < >   Bluetooth subsystem support  --->
```
Bluetooth kommunikáció támogatása
```
Wireless  --->
```
Wifi hálózati stack
```
< >   RF switch subsystem support  ---
```
Wifi és Bluetooth kártyákon található rádiófrekvenciás kapcsolók támogatása.

Device Drivers ---> (eszközmeghajtó programok)

```
Generic Driver Options
```
általános driver opciók, firmware használatának lehetősége
```
< > Memory Technology Device (MTD) support  --->
```
Flash és egyéb speciális memóriák támogatása. Ezekkel főként beágyazott rendszerekben találkozhatunk.
```
< > Parallel port support  --->
```
Párhuzamos port támogatása
```
-*- Plug and Play support  --->
```
Plug and Play eszközök támogatása

```
[ ] Block devices  --->
```
blokkos eszközök támogatása
```
< >   Normal floppy disk support
```
Floppy támogatása
```
< >   Loopback device support
```
„Visszahurkoló eszköz” támogatása. Segítségével egy fájlt ugyanúgy lehet használni, mint egy partíciót. pl.: CD iso image csatlakoztatásakor használatos.
```
< >   Network block device support
```
Hálózaton keresztüli erőforrások elérése blokkos eszközként
```
 < >   RAM disk support
```
A memória egy részét úgy használhatjuk a segítségével, mint egy diszket. Abban az esetben, hogyha a kernel initrd-t (nem initramfs-t) használ, akkor feltétlenül szükséges.

```
[ ] Misc devices  --->
```
Egyéb eszközök támogatása, amely egyik kategóriába sem sorolható.

```
<M> ATA/ATAPI/MFM/RLL support  --->
```
Általános IDE/ATA támogatás
```
<M>   Generic ATA/ATAPI disk support
```
Általános ATA/ATAPI diszk támogatás
```
< >     ATA disk support
```
újabb ATA módok támogatása
```
<M>     Include IDE/ATAPI CDROM support
```
IDE CDROM támogatása
```
<M>     Intel PIIX/ICH chipsets support
```
Az Intel PIIX és ICH alaplapi chipset támogatása

```
SCSI device support  --->
```
Általános SCSI támogatás
```
<*> SCSI device support
```
SCSI eszközök támogatása
```
<*> SCSI disk support
```
SCSI lemez támogatása
```
< > SCSI CDROM support
```
SCSI CDROM támogatása
```
SCSI Transports  --->
```
SCSI eszközök átviteli jellemzőinek megtekintése
```
-*-   Paralell SCSI (SPI) Transport Attributes
```
Párhuzamos SCSI transzport támogatása
```
[*] SCSI low-level drivers  --->
```
Alacsony szintű SCSI eszközök
```
<*>   BusLogic SCSI support
```
BusLogic típusú SCSI vezérlő támogatása (a VMware ilyet is tud emulálni)

```
< > Serial ATA (prod) and Parallel ATA (experimental) drivers
```
ATA protokollt használó hoszt adapterek elérése. pl. SATA lemezek esetében. A SCSI diszk illetve a SCSI CDROM támogatás szükséges a használathoz.

```
[*] Multiple devices driver support (RAID and LVM)  --->
```
RAID és LVM használata
```
<M>   RAID support
```
RAID szintek támogatása
```
<M>     RAID-0 (striping) mode
<M>     RAID-1 (mirroring) mode
```
A megfelelő RAID szintek támogatása
```
<M>   Device mapper support
```
LVM támogatása

```
[*] Fusion MPT device support  --->
```
Fusion Message Passing Technology eszközök támogatása
```
<*>   Fusion MPT ScsiHost drivers for SPI
```
A VMware által emulált lsilogic típusú vezérlő támogatása

```
IEEE 1394 (FireWire) support  --->
```
Firewire csatoló támogatása (pl. külső diszk vagy digitális videókamera)

```
< > I2O device support  --->
```
Intelligent Input/Output eszköz támogatása. Lehetővé teszi, hogy a drivereket két részre bontsuk. Egy operációs rendszer függő illetve egy hardver függő részre.

```
[*] Network device support  --->
```
Hálózati eszközök támogatása
```
< >   Bonding driver support
```
Több eszköz összekapcsolása (trönk)
```
< >   Universal TUN/TAP device driver support
```
Csomagküldés felhasználói programoknak. Lehetővé teszi a hálózati csomagok továbbítását pl.: egy virtuális gép felé.

```
[*]   Ethernet (10 or 100Mbit)  --->
```
10 vagy 100 Mbites Ethernet kártyák támogatása
```
[*]   EISA, VLB, PCI and on board controllers
```
EISE, VLB, PCI és alaplapi kártyák támogatása
```
<M>     AMD PCnet32 PCI support
```
AMD PCnet32 alapú ethernet kártyák támogatása (a VMware ilyet emulál)
```
[ ]   Ethernet (1000 Mbit)  --->
[ ]   Ethernet (10000 Mbit)  --->
```
Gigabites, 10 Gigabites ethernet kártyák támogatása
```
Wireless LAN  --->
```
Wifi eszközök támogatása
```
< >   PPP (point-to-point protocol) support
```
Pont-pont kapcsolat támogatása

```
< > ISDN support  --->
```
ISDN eszközök támogatása

```
< > Telephony support  --->
```
Telefon kártya támogatás. Lehetővé teszi VoIP alkalmazások használatát.

```
Input device support  --->
```
Bemeneti eszközök támogatása
```
-*- Generic input layer (needed for keyboard, mouse, ...)
```
Általános billentyűzet, egér támogatás
```
-*-   Mouse interface
```
/dev/input/mice eszközfájl használata
```
-*-   Keyboards  --->
```
Billentyűzetek
```
[*]   Mice  --->
   <M>   PS/2 mouse
```
PS/2-es egér támogatása

```
Character devices  --->
```
karakteres eszközök támogatása
```
-*- Virtual terminal
```
Virtuális terminál támogatás
```
    Serial drivers  --->
```
Soros port használata
```
-*- Unix98 PTY support
```
Unix98 pszeudo terminál támogatás
```
< > Enhanced Real Time Clock Support
```
Valós idejű óra
```
< > I2C support  --->
```
I2C busz támogatása
```
[ ] SPI support  --->
```
Serial Peripheral Interface protokoll támogatása
```
-*- Power supply class support  --->
```
Az elemek illetve a tápfeszültség monitorozása

```
< > Hardware Monitoring support  --->
```
Hardver eszközök feszültségének, hőmérsékletének, ventilátorok forgási sebességének monitorozása
```
[ ] Watchdog Timer Support  --->
```
A gép működésének figyelése, hiba esetén az újraindítása
```
Multimedia devices  --->
```
Multimédiás eszközök támogatása (pl. TV tuner kártya, rádiós kártya)

```
Graphics support  --->
```
Grafikai támogatások
```
 < > /dev/agpgart (AGP Support)  --->
```
AGP port támogatása
```
 < > Support for frame buffer devices  --->
```
Frame buffer console támogatása
```
Console display driver support  --->
 -*- VGA text console
```
VGA console támogatása
```
Sound  --->
```
Hangkártyák támogatása
```
[*] HID Devices  --->
```
Human Interface Device támogatása

```
[*] USB support  --->
```
Általános USB támogatás
```
<M>   Support for Host-side USB
```
USB Host támogatás
```
<M>     EHCI HCD (USB 2.0) support
```
USB 2.0 támogatása
```
< >   USB Printer support
```
USB csatlakozós nyomtató támogatása
```
< >   USB Mass Storage support
```
USB-s tárolóeszközök támogatása

```
< > MMC/SD card support  --->
```
Kártyaolvasók támogatása
```
< > InfiniBand support  --->
```
Infiniband protokoll támogatása (tároló eszközök)

Firmware Drivers ---> (firmware eszközmeghajtók)

File systems ---> (fájlrendszerek támogatása)

```
< > Second extended fs support
```
EXT2 fájlrendszer támogatása
```
<*> Ext3 journalling file system support
```
EXT3 naplózó fájlrendszer támogatása
```
< > The Extended 4 ext4 filesystem
```
EXT4 naplózó fájlrendszer támogatás

```
< > Reiserfs support
< > JFS filesystem support
< > XFS filesystem support
```
Reiserfs (3.6), JFS, XFS fájlrendszerek támogatása
```
< > GFS2 file system support
< > OCFS2 file system support
```
Global FS klaszter fájlrendszer és Oracle klaszter fájlrendszer 2-es verziójának támogatása

```
[*] Inotify file change notification support
[*]   Inotify support for userspace
```
Fájl változások jelzésére használható

```
< > Kernel automounter support
```
Távoli fájlrendszerek automatikus csatlakoztatása
```
< > FUSE (Filesystem in Userspace) support
```
FUSE támogatás (felhasználói térben fájlrendszerek implementálását teszi lehetővé)

```
CD-ROM/DVD Filesystems  --->
```
CD/DVD fájlrendszerek
```
<M> ISO 9660 CDROM file system support
[*]   Microsoft Joliet CDROM extensions
```
ISO9660 fájlrendszer Joliet kiterjesztéssel
```
< > UDF file system support
```
UDF fájlrendszer támogatása
```
DOS/FAT/NT Filesystems  --->
```
DOS, FAT, NTFS fájlrendszerek támogatása

```
Pseudo filesystems  --->
```
pszeudo fájlrendszerek
```
-*- /proc file system support
```
/proc fájlrendszer támogatása
```
[*] Virtual memory file system support (former shm fs)
```
tmpfs fájlrendszer támogatása (mindent a virtuális memóriában tárol)

```
Miscellaneous filesystems  --->
```
Egyéb fájlrendszerek (Apple HFS, OS/2 HPFS, UFS, ...)
```
< > Compressed ROM file system support (cramfs)
```
Initrd esetében (nem initramfs) szükség van a cramfs támogatásra, ugyanis az initrd image ezt a fájlrendszer használja.

```
[ ] Network File Systems  --->
```
Hálózati fájlrendszerek támogatása
```
< >   NFS file system support
< >   NFS server support
```
Network FS eléréséhez illetve saját kötetek megosztásához
```
< >   SMB file system support (OBSOLETE, please use CIFS)
< >   CIFS support (advanced network filesystem SMBFS successor)
```
Windows alatti megosztások, vagy Samba megosztásokhoz való hozzáférés

```
< >   NCP file system support (to mount NetWare volumes)
```
Novell Netware kötetek elérése
```
< >   Coda file system support (advanced network fs)
```
Coda kliensként való működés

```
Partition Types  --->
```
Partíció típusok választása
```
PC BIOS (MSDOS partition tables) support
```
Hagyományos MSDOS típusú partíciós tábla elérése
```
[ ]   Windows Logical Disk Manager (Dynamic Disk) support
```
Az „ablakozós” operációs rendszerek Logical Disk Manager programjával létrehozott partíciók kezelése

```
{*} Native language support  --->
```
Karakter készletek, kódlapok támogatása (UTF-8, Latin-1, Latin-2, Windows-1250, Codepage 852, ...)

```
Kernel hacking  --->
```
kernel „hackelése”
```
[*] Magic SysRq key
```
Alt+PrintScreen és egyéb billentyűk megnyomása segítségével egy kernel hiba után ki lehet üríteni a buffer cache-t és a rendszert újra lehet indítani.

```
Security options  --->
```
Biztonsági beállítások

```
[ ] Cryptographic API  --->
```
Titkosítási algoritmusok megvalósításai főként az IPSec számára

```
[ ] Virtualization  --->
```
KVM (Kernel-based Virtual Machine) támogatások

```
Library routines  --->
```
Általános célú könyvtári rutinok
```
{M} CRC32 functions
```
CRC ellenőrzése eszközök és hálózati szolgáltatások esetében

6.4. A konfigurálás néhány fontos beállítása

Amikor kernelt konfigurálunk van néhány fontos beállítás, amely függvénye a telepítési szituációnak. A VMware virtuális gép esetében a következő táblázat foglalja össze ezeket a beállításokat. A táblázatban a – jelenti, hogy nincsen szükség az adott támogatásra, az M jelöli, hogy modulként, a * pedig, hogy be kell építeni a kernelbe az adott támogatást. A táblázat a minimális támogatásokat tartalmazza, ahol a -, M, * felel meg a növekvő sorrendnek.

Az egyes támogatások minimális szükséglete 8. sz. táblázat

Eszköz	***Kernel támogatás(ok)***	***initrd***	***noinitrd***
SCSI támogatás	SCSI device support	M	*
LSI Logic SCSI vez.	Fusion MPT device support	M	*
	Fusion MPT ScsiHost drivers for SPI	M	*
BusLogic SCSI vez.	BusLogic SCSI support	M	*
SCSI diszk	SCSI disk support	M	*
IDE támogatás	ATA/ATAPI/MFM/RLL support	M	*
IDE vezérlő (PIIX4)	Intel PIIX/ICH chipsets support	M	*
IDE diszk	ATA/ATAPI/MFM/RLL support	M	*
	Generic ATA/ATAPI disk support	M	*
	ATA disk support	M	*
RAID 0,1,5,10	RAID support	M	*
	megfelelő RAID szint támogatása pl.: RAID-1 (mirroring) mode	M	*
LVM	Device mapper support	M	!
Initramfs/Initrd	Initial RAM filesystem and RAM disk (initramfs/initrd) support	*	-
	RAM disk support (csak initrd)	*	-
/ fájlrendszer	a megfelelő fájlrendszer (ext2, ext3, reiserfs, xfs, jfs) támogatása pl.: Ext3 journalling file system support	M	*
további fájlrendszerek	a megfelelő helyi vagy távoli fájlrendszer támogatása	M	M
CRAMFS (csak initrd)	Compressed ROM file system support (cramfs)	*	-

A táblázat utolsó két oszlopa tartalmazza azt, hogy mi a különbség abban az esetben, hogy initrd image-t szeretnénk használni, illetve hogyha nem akarunk használni. A felkiáltójel jelzi, hogy initrd nélkül az LVM használata nem támogatott. A logikai kötetkezeléshez ugyanis feltétlenül szükség van a megfelelő userspace-beli programokra. Ezek nélkül a kernel nem képes a megfelelő / fájlrendszert megtalálni.

6.5. Fordítás előzetes konfiguráció alapján

Amennyiben végignéztük a konfigurálás lehetőségeit és beállítottuk a számunkra szükséges opciókat úgy ezeket a beállításokat el kell mentenünk. Amikor az „exit” menüpontot választjuk, akkor rákérdez a mentésre:

```
Do you wish to save your new kernel configuration?
```

Ebben az esetben válaszoljunk igennel, mivel a korábbi kijelöléseink csak ebben az esetben lesznek mentve. Alapértelmezésként a .config nevű fájlba kerül a mentés a kernel forrás gyökérkönyvtárába (/usr/src/linux). Amennyiben szeretnénk egy adott konfigurációt menteni és a későbbiekben felhasználni, akkor ezt a fájlt másoljuk át egy külön könyvtárba.
A mentést és a konfiguráció visszatöltését menüpontok segítségével is el tudjuk végezni:

```
Load an Alternate Configuration File
Save an Alternate Configuration File
```

Amennyiben verzió váltás van előfordulhatnak problémák abban az esetben, hogyha egy régebbi konfigurációs fájlt töltöttünk vissza. Ennek oka lehet az, hogy például megváltoztak egy korábbi verzióhoz képest a konfigurációs bejegyzések nevei. Éppen ezért visszatöltés után ajánlott egyszer végigszaladni az összes menüponton, hogy áttekintsük az újabb lehetőségeket és a szükségesek beállítását ellenőrizzük.

Ezután kiadhatjuk a fordítás parancsait.

Hagyományos módon (működik a 2.6-nál korábbi kernelek esetében is):

- `make dep` függőségek vizsgálata
- `make clean` törlés
- `make bzImage (vagy make zImage)` kernel image bzip2-vel vagy gzippel tömörítve
- `make modules` kernel modulok fordítása
- `make modules_install` kernel modulok telepítése

Egyszerűsített módon:

- `make clean` törlés
- `make` kernel image (bzip2-vel tömörítve) és modulok
- `make modules_install` modulok telepítése

A `make clean` parancs végrehajtása csak abban az esetben szükséges, hogyha már volt egy korábbi fordítás. Ez törli a kernel image-t illetve az egyes object fájlokat. A `make help` paranccsal tájékozódhatunk a make egyéb lehetőségeiről. A fordítás parancsai architektúránként különbözhetnek. Manapság ajánlott több szálon indítani a fordítást, ezzel jelentősen csökkenthető a fordítási idő. (pl.: `make -j 4`)

Amennyiben a parancsok sikeresen végrehajtódnak elkészül a kernel image, illetve az egyes kernel modulok. Előfordulhat, hogy bizonyos driverek kiválasztása fordítási problémát okoz (pl.: 2.6.22.9 RxRPC session socket), de ez nem jellemző.

A fordítás eredményeként létrejön:

- `arch/i386/boot/bzImage` kernel image (x86 architektúra esetében)

- `System.map` kernel szimbólum tábla
- `/lib/modules/"version"` a telepített kernelmodulok a verziónak megfelelő könyvtárban (pl.: `"version" = 2.6.31.9-kami`)

Ezután következhet az elkészült fájlok másolása a /boot könyvtárba:
- `cp arch/i386/boot/bzImage /boot/vmlinuz-2.6.31.9-kami`
- `cp System.map /boot/System.map-2.6.31.9-kami`

Abban az esetben, hogyha a kernel initramfs/initrd image-t is használ a rendszerinduláshoz, akkor azt létre kell hozni a rendelkezésre álló modulok felhasználásával. Például:
- `mkinitramfs -o /boot/initrd.img-2.6.31.9 2.6.31.9-kami`

A parancs esetében a -o kapcsoló után adjuk meg a kimeneti fájl nevét, a következő paraméter pedig a kernel verzió. Ez is nagyon fontos, hiszen a kernel csak a megfelelő verziójú modulok segítségével működhet megfelelően.

Ezután következhet a rendszerbetöltő konfigurálása. A GRUB esetében például a következő módon állíthatjuk be a szükséges paramétereket:

```
title           Debian GNU/Linux, kernel 2.6.31.9
root            (hd0,0)
kernel          /vmlinuz-2.6.31.9-kami root=/dev/sda1 ro
savedefault
```

A legegyszerűbb, hogyha az alapértelmezett kernelhez tartozó részt lemásoljuk. Majd ezt követően módosítjuk az image fájlok neveit és a menüpont nevét. Célszerű az új kernelhez tartozó menüpontot az első helyre tenni, hiszen ebben az esetben ő lesz az alapértelmezett és a következő rendszerinduláskor a GRUB azt próbálja majd meg automatikusan indítani. Ajánlott meghagyni a korábbi verzió(ka)t, hiszen előfordulhat, hogy valamilyen konfigurációs hiba miatt az új kernelünk nem lesz működőképes. Ebben az esetben a korábbi kernel segítségével még tudunk rendszert indítani és a konfigurálást és fordítást újra megpróbálni.

Amennyiben a kernel indulásához szeretnénk initrd-t is használni, úgy azt létre kell hozni, majd elhelyezni a rá mutató hivatkozást a /boot/grub/menu.lst-ben. Például a kernel-el kezdődő sor után berakhatjuk a következő sort:

```
initrd          /initrd.img-2.6.31.9
```

Ezután a paraméterként átadott fájl lesz a kezdeti root fájlrendszer, amely betöltődik a ramdiszkbe és onnét a kernel drivereket tud betölteni és programokat tud elindítani.

A Debian 5.0 támogatja a GRUB 2-t is, ennek használatával egyelőre nem fogunk foglalkozni.

6.6. Kernel fordítás „Debian módra”

Debian GNU/Linux operációs rendszerben a kernel-package nevű csomag a kernel fordítás elősegítésére készült. Ebben a csomagban található a make-kpg parancs, amely segítségével kernelt lehet fordítani, és az elkészült kernelből .deb csomagot készíteni. Használatának a legfontosabb előnyei:

- a kernelfordítás kimenete .deb formában jelenik meg (kernel-image-*${version}_${revision}_${arch}*.deb), amelyet a későbbiek csomagkezelővel tudunk telepíteni, akár másik gépre is
- a kernellel együtt lehet ún. külső modulokat is fordítani (pl.: aufs, nvidia, ...), amelyekből szintén csomagok készülnek (a module-assistant (m-a) csomagot (parancsot) felhasználva)
- nem kell szerkeszteni a GRUB beállításait, a megfelelő sorok a kernel-image csomag telepítéskor automatikusan létrejönnek
- a dpatch segítségével saját foltok használhatók fel

Ilyenkor az alábbiak szerint végezhetjük el a kernel fordítását:

- kernel forrás letöltés, ellenőrzés, kicsomagolás a korábbiak szerint
- a */usr/src/modules* könyvtárban kicsomagolni azokat a modulokat, amelyeket szeretnénk a kernellel együtt fordítani. Ezek rendre a *-source* -ra végződő csomagokban vannak (pl. aufs-source, nvidia-kernel-source, ...)
- a kernel konfigurálása a korábbiak szerint
- a kernel fordítása: `make-kpkg --initrd kernel_image`
- a külső modulok fordítása (opcionális): `make-kpkg modules_image`
- header fájlok csomagolása (opcionális): `make-kpkg kernel_headers`

Egy egyszerű szkripttel tetszőleges verziószámot adhatunk hozzá a csomagokhoz, illetve beállíthatjuk azt is, hogy párhuzamosan hány szálon történjen a fordítás.

```
#!/bin/bash
export CONCURRENCY_LEVEL=4
V=-kami
R=1
make-kpkg --append-to-version $V --revision $R --initrd kernel_image
make-kpkg --append-to-version $V --revision $R modules_image
make-kpkg --append-to-version $V --revision $R kernel_headers
```

Egy sikeres fordítás eredményeként létrejönnek az alábbi fájlok: *linux-image-2.6.31.9-kami_1_i386.deb*, *linux-headers-2.6.31.9-kami_1_i386.deb*. Ezt követően a csomagokat a dpkg paranccsal telepíthetjük. és ellenőrizhetjük a kernelünk működését.

A module-assistant csomaggal „utólagosan” is lehet egy kernelhez modulokat fordítani. (Feltéve, hogy ugyanaz a C fordítónk van, és rendelkezésre állnak a kernel header fájlok.) Egy példa a használatára:

```
m-a -k /usr/src/linux-2.6.31 -l 2.6.31.9-kami build nvidia
```

A példában az nvidia driverből készítünk kernel modult. A *-k* opcióval a kernel header fájlok helyét, a *-l* opcióval pedig a kernel verziót adjuk meg.

7. Felhasználó adminisztráció és a PAM

7.1. A felhasználó adminisztráció parancsai

Amikor egy rendszert telepítünk, akkor a rendszergazda (root) felhasználó jelszavát meg kell adnunk. A jelszó megadásánál figyelnünk kell arra, hogy az minél nehezebben legyen kitalálható az ún. jelszó feltörő alkalmazások segítségével. Így ajánlott kis és nagybetűket is használni, emellett számokat és speciális karaktereket is. Célszerű a billentyűzet kiosztások adta különbségekre is odafigyelni.
A Debian GNU/Linux használ ezen kívül jó néhány beépített csoportot és felhasználót is. Ezeknek az egyes eszközök jogosultságai esetében van fontos szerepük.
Például:

```
brw-rw---- 1 root disk 8, 0 2008-03-18 07:05 /dev/sda
brw-rw---- 1 root disk 8, 1 2008-03-18 07:06 /dev/sda1
```

Vagyis csak azok a felhasználók írhatják a /dev/sd{a,a1} eszközöket, akik tagjai a disk nevű csoportnak vagy rendszergazdák. Ennélfogva a jogosultság elhatárolás egy fontos eszköze. Ezen „beépített” felhasználókra illetve felhasználói csoportokra jellemző, hogy 0-től fölfelé növekednek az azonosítóik.

A beépített felhasználók másik fontos felhasználási területe az, hogy egyes szolgáltatások ezeket a meglévő azonosítókat használják fel a működésük során.
Részlet egy: `ps -u daemon,www-data -o pid,user,cmd` parancs kimenetéből:

```
  PID USER     CMD
 2345 daemon   /sbin/portmap -i 127.0.0.1
 2821 daemon   /usr/sbin/atd
14349 www-data /usr/sbin/apache2 -DSSL -k start
```

A parancs kimenetéből kiolvashatjuk, hogy a daemon nevű felhasználóként kettő szolgáltatás (portmap, atd) fut a lekérdezés pillanatában, míg www-data felhasználóként az apache2 szolgáltatás.

Egyes .deb csomagokban találhatók olyan scriptek, amelyek a csomag telepítésekor automatikusan létrehozzák a szükséges felhasználókat illetve felhasználói csoportokat. Általában 100-999 között szokás a „rendszerfelhasználókat” (programok végrehajtásához kellenek), míg 1000-től felfelé a normál felhasználókat.

Az Operációs rendszerek – Linux jegyzetben bemutatásra került a felhasználói adatok tárolásának módszere (/etc/passwd, /etc/shadow, ...), ezen fájlok részletes felépítése. Így ennek ismeretét feltételezem.

Amikor saját felhasználói csoportokat vagy felhasználókat szeretnénk létrehozni, akkor többféle lehetőség közül választhatunk. A továbbiakban csak az ide kötődő parancsokat szeretném bemutatni. Manapság sok lehetőség kínálkozik annak a megoldására, hogy a felhasználók adminisztrációját kényelmesen megoldjuk (pl.:

Webmin segítségével böngészőn keresztül). Ezekkel a továbbiakban nem szeretnék foglalkozni, csak a parancssori lehetőségekkel, amely bárhol használható és nincsenek egyéb szoftver követelményei.

Debian GNU/Linux operációs rendszerben a passwd illetve az adduser nevű csomagban találhatók azok a parancsok, amelyek segítségével az adminisztráció megvalósítható. A továbbiakban példákon keresztül fogom bemutatni a parancsok fontosabb lehetőségeit.

Felhasználói csoportok felvétele:

- `addgroup --gid GID group`
- `adduser --group GID group`
- `groupadd -g GID group`

Az előző parancsok esetében a `GID` jelenti a csoport azonosítót, a `group` pedig a csoport nevét.

Felhasználói csoportok törlése:

- `delgroup group`
- `deluser --group group`
- `groupdel group`

Felhasználói csoportok jellemzőinek módosítása:

- `groupmod -g GID -n group_name group`

A módosítás esetében két fő lehetőségünk van. A felhasználói csoport azonosítóját tudjuk megváltoztatni és/vagy a csoport nevét.

Felhasználók felvétele:

- `adduser --uid UID --gid GID user`
- `useradd -u UID -g GID user`

Természetesen a felhasználók egyéb jellemzői is a megfelelő paraméterek segítségével beállíthatók. Az adduser segítségével megvalósítható a felhasználók „interaktív” felvétele. Ebben az esetben rákérdez a felhasználó egyes jellemzőire (GECOS field) a felvétel folyamán.

```
adduser tux
Adding user `tux' ...
Adding new group `tux' (1001) ...
Adding new user `tux' (1001) with group `tux' ...
Creating home directory `/home/tux' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: a jelszó sikeresen frissült
tux felhasználói információinak cseréje
Add meg az új értéket vagy üss ENTER-t az alapértelmezetthez
        TELJES Név []: Tux
        Szobaszám []:
        Munkahelyi telefon []:
        Otthoni telefon []:
        Egyéb []:
Is the information correct? [Y/n] Y
```

A parancs pontos kimenete függ a nyelvi beállításoktól. Amennyiben ez magyar, akkor részben angol, részben pedig magyar lesz a parancs kimenete.

Felhasználók törlése:

- `deluser --remove-all-files user`
- `userdel -r user`

A /etc/adduser.conf illetve a /etc/deluser.conf segítségével megadhatók az adott parancs esetében az alapértelmezések. Amennyiben nem szeretnénk ezektől eltérni, akkor nem szükséges az ott található bejegyzéseket a megfelelő opció segítségével felüldefiniálni. A useradd parancs működésére pedig a /etc/default/useradd nevű fájl van hatással. A felhasználó törlési példákban szeretnénk a felhasználó összes fájlját a felhasználóval együtt törölni a rendszerből.

Felhasználók jellemzőinek módosítása:

- `usermod [options] user`

Néhány módosítási opció:

- `-d HOME_DIR` saját könyvtár módosítása
- `-g GROUP` alapértelmezett csoport
- `-G GROUPS` csoport tagságok vesszővel elválasztott listaként
- `-l USER_NAME` átnevezés
- `-s SHELL` alapértelmezett shell módosítása
- `-u UID` UID módosítása

Felhasználók jelszavainak és jelszó beállításainak a módosítása

Rendszergazdaként fontos feladat a jelszavak kezeléséhez tartozó szabályozások definiálása. A felhasználóink könnyen elfelejtik a ritkán használt jelszavakat, és ilyenkor egy másikat kell definiálnunk helyette. Egy másik probléma az, hogy meddig biztonságos egy jelszó? Amennyiben előbb megváltoztatjuk a jelszót, mint azt bárki vissza tudná fejteni, akkor mondhatjuk azt, hogy biztonságos. Azonban a mai számítási teljesítmény mellett ezt nem igazán teljesíthető. Éppen ezért nagy gondot kell fordítanunk a tárolt jelszavak megfelelő védelmére.

Rendszergazdaként a saját jelszavunkat illetve bárki más jelszavát is megváltoztathatjuk:

- `passwd [options] user`

Néhány használható opció:

- `-S` a jelszó állapotának megtekintése (/etc/shadow „flag”-ek)

```
tux P 03/18/2008 0 99999 7 -1
```

- `-l` lock (a fiók zárolása)
- `-u` unlock (a fiók zárolásának feloldása)
- `-x N` jelszó módosítási kötelezettség N nap után
- `-n N` jelszó csak N nap eltelte után módosítható
- `-w N` jelszó lejárta előtt N nappal figyelmeztet

- `-i N` fiók inaktívvá tétele N nappal a lejárat után

Abban az esetben, hogyha sok felhasználó jelszavát szeretnénk egyszerre módosítani, ajánlott a chpasswd nevű csomag telepítése. A csomagban lévő ugyanolyan nevű parancs segítségével több felhasználó jelszavát tudjuk módosítani. Ha a kódolt vagy a kódolatlan jelszóval rendelkezik adott felhasználók esetében, akkor készíthetünk egy listát. A lista tartalmazza soronként a felhasználók neveit illetve jelszavait kettősponttal elválasztva. Amennyiben ezt a chpasswd parancs standard bemenetére irányítjuk úgy az összes felhasználó jelszavát meg tudjuk változtatni.

A felhasználói bejelentkezéshez kötődik a /etc/login.defs nevű fájl. Ebben tudjuk szabályozni a felhasználókra vonatkozó bejelentkezési paramétereket, környezeti változókat, könyvtárakat.

7.2. A PAM (Pluggable Authentication Modules)

Linux operációs rendszerben több alkalmazásnak lehet arra szüksége, hogy a felhasználókat azonosítsa. Ilyen lehet például a login, amely a felhasználói bejelentkezést kezeli vagy pedig egy képernyőzár. Mindegyik programnak meg kell tudnia állapítani, hogy a felhasználó megfelelő jelszót adott-e meg. Azért, hogy egy alkalmazásba tárolási módtól függetlenül tudja kezelni a hitelesítést találták ki a PAM-ot. Vagyis az alkalmazásom számára teljesen mindegy, hogy a felhasználói adatok helyileg vannak tárolva vagy hálózaton egy központi szerveren. A PAM „megoldja” a megfelelő adatok elérését, az alkalmazásnak csak a PAM-tól kell „megkérdezni-e”, hogy például létezik-e adott felhasználó a megadott jelszónak megfelelően.

A PAM legfontosabb jellemzői:
- API-t nyújt az alkalmazások számára, amely ezen függvényeket használhatja fel a felhasználók hitelesítésére
- moduláris felépítésű, az egyes modulok dinamikus programkönyvtárként érhetők el és használhatók igény szerint
- „transparent authentication and authorization”: megvalósítja tehát a tárolási módtól független hitelesítést
- az alkalmazások részéről támogatás szükséges a használatához

Manapság a legtöbb Linux disztribúció támogatja a használatát. Így a Debiannak is része a 2.2-es verzió óta.

A rendszerben rendelkezésre álló modulok a /lib/security könyvtár alatt érhetők el. A fájlok nevei egységesen pam_ -al kezdődnek.

A PAM beállításait a /etc/pam.d könyvtárban találhatjuk. A Debian azt a konvenciót követi, hogy ezen belül a konfigurációs fájlok nevei megegyeznek annak a szolgáltatásnak a nevével, amelyre a beállítások vonatkoznak. Ezen felül léteznek olyan fájlok, amelyek általánosak (common-*) és több fájlban is ezekre a beállításokra hivatkoznak.

A konfigurációs fájlok esetében a következő séma szerint felépülő utasításokat használhatjuk:

```
type  control_flag  module-path  module-arguments
```

A `type` jelenti a hitelesítés típusát. Van arra lehetőség, hogy megadjuk azt, hogy egy felhasználó több kritériumnak is meg kell, hogy feleljen. A típus lehetséges értékei:

- `account`: a fiókhoz való hozzáférésre vonatkozó szabályok
- `auth`: a hitelesítés módjának a beállítása
- `password`: a jelszó változtatására vonatkozó szabályok
- `session`: a felhasználó hitelesítése előtt és/vagy után (a munkamenethez kapcsolódó) végrehajtandó feladatok

A `control_flag` jelenti annak a szabályozását, hogy mi történjek, abban az esetben, ha valamelyik modul sikertelen eredményt ad vissza. Ez rendkívül fontos a végső eredmény szempontjából. A lehetséges értékei:

- `requisite`: a hitelesítés azonnali megtagadása, hogyha adott modul sikertelen eredménnyel tér vissza
- `require`: a hitelesítés megtagadását jelenti, de ezután a beállított többi PAM modult is meg fogja hívni a megtagadás előtt
- `sufficient`: ha ezen modul alapján sikeres a hitelesítés, akkor megadja azt, függetlenül a korábbi moduloktól
- `optional`: a sikeresség/sikertelenség csak akkor fontos, hogyha kizárólag ilyen jellegű modul van beállítva

A `module-path` jelenti a PAM modul nevét vagy elérési útvonalát, hogyha nem az alapértelmezett helyen található.
A `module-arguments` pedig a PAM modul számára átadandó fontosabb paramétereket tartalmazhatja.

Az általános beállítások a következők:

/etc/pam.d/common-account
```
account required        pam_unix.so
```

/etc/pam.d/common-auth
```
auth    required        pam_unix.so nullok_secure
```

/etc/pam.d/common-password
```
password   required   pam_unix.so nullok obscure min=4 max=8 md5
```

/etc/pam.d/common-session
```
session required        pam_unix.so
```

A pam_unix.so modul jelenti a Unix rendszerekben általánosságban használt /etc/passwd, és /etc/shadow alapján történő hitelesítést.

A `nullok` beállítás megengedi, hogy azok a felhasználók, akiknek nincsen jelszavuk, azok is be tudjanak jelentkezni. Az `obscure` a jelszó „erősségének” ellenőrzésére vonatkozik, a `min` és `max` a jelszó hosszára. Az `md5` segítségével állíthatjuk be, hogy a jelszavaink az MD5 hash algoritmus alkalmazásával legyenek tárolva.

Gyakran használt PAM modulok:

- pam_unix.so standard Unix hitelesítés
- pam_access.so hozzáférés szabályozása (/etc/security/access.conf)
- pam_cracklib.do jelszó ellenőrzése
- pam_deny.so hozzáférés megtagadása
- pam_env.so környezeti változók beállítása (/etc/environment)
- pam_lastlog.so az utolsó bejelentkezés helye és ideje
- pam_limits.so erőforrások korlátozása (/etc/security/limits.conf)
- pam_listfile.so fájl tartalom alapján való hitelesítés
- pam_ldap.so LDAP szerver alapján való hitelesítés
- pam_mail.so mailbox információk mutatása bejelentkezés után
- pam_mkhomedir.so a felhasználók saját könyvtárának létrehozása bejelentkezéskor
- pam_motd.so a /etc/motd kiíratása bejelentkezés után
- pam_nologin.so ha a /etc/nologin fájl létezik, akkor csak root-ként lehet bejelentkezni
- pam_permit.so feltétel nélkül garantálja az elérést
- pam_rootok.so amennyiben root-ként vagyunk bejelentkezve nem szükséges adott alkalmazás esetében a hitelesítés
- pam_securetty.so root-ként való bejelentkezés engedélyezése a /etc/securetty alapján
- pam_shells.so valós shell vizsgálata /etc/shells alapján

Az előző felsorolásból látható, hogy számos feladat elvégzésére léteznek PAM modulok. Mindezek mellett természetesen van arra is lehetőség, hogy saját modulokkal kiegészítsük a rendszert.

Példa több PAM modult is használó fájlra (/etc/pam.d/ssh):

```
# PAM configuration for the Secure Shell service
# Read environment variables
auth       required     pam_env.so # [1]
auth       required     pam_env.so envfile=/etc/default/locale
@include common-auth

# Disallow non-root logins when /etc/nologin exists.
account    required     pam_nologin.so
@include common-account

# Standard Un*x session setup and teardown.
@include common-session
session    optional     pam_motd.so # [1]
session    optional     pam_mail.so standard noenv # [1]
session    required     pam_limits.so
```

```
# Standard Un*x password updating.
@include common-password
```

A PAM esetében fontos, hogy miként kezelje a számára ismeretlen alkalmazásokat. Itt azokra az alkalmazásokra gondolok, amelyek PAM hívásokat használnak, csak nem tartozik hozzájuk definiált konfigurációs fájl. Ezen alkalmazásokra a /etc/pam.d/other fájl tartalma fog vonatkozni.

8. A TCP/IP alapú hálózat beállítása

Amikor Debian GNU/Linux-ot telepítünk, akkor van arra lehetőségünk, hogy a hálózati paramétereket beállítsuk. Természetesen ezeket a későbbiek folyamán igény szerint módosíthatjuk. A továbbiakban bemutatom, hogy milyen módon célszerű a hálózati működést beállítani és ellenőrizni. A továbbiakban csak ethernet-tel és IPv4-el foglalkozom.

8.1. A hálózati beállítás folyamata

Amikor egy gép hálózatban működik, abban az esetben szükség van bizonyos paramétereket beállítani a hálózati eszközön és természetesen az operációs rendszerben. A kettő sokszor nem kezelhető egymástól függetlenül. A beállításnál ajánlott az alábbi sorrendben ellenőrizni a hálózat működését:

- A megfelelő eszközmeghajtók betöltése. Ahhoz, hogy a hálózat működhessen a kernelnek ismernie kell az adott hálózati eszközt. A /proc/net/dev fájlból tájékozódhatunk a rendelkezésre álló interfészekről. A Debian 5.0 használja az udev-et. Ami alapértelmezésként megjegyzi az egyes hálózati eszközök MAC címét, és e szerint rendel hozzájuk interfész neveket. Vagyis lehetséges, hogy az interfész létezik, csak nem az alapértelmezett néven. (Az 5.0 már nem jegyzi meg a VMware-es MAC címeket.)
- Az eszközmeghajtók mellett természetesen szükséges a kernel részéről a használni kívánt protokollok támogatása. A /proc/net/protocols fájlból tájékozódhatunk.
- Miután meggyőződtünk arról, hogy rendszerünk alkalmas a hálózati kommunikációra ellenőriznünk kell, hogy létezik-e fizikai kapcsolata (linkje) hálózati eszköznek. Lehetséges ugyanis, hogy a kábellel vagy a csatlakozóval van a probléma, vagy például a switch adott portja van rosszul konfigurálva.
- Amennyiben az előzőek rendben vannak, következhet az IP cím és egyéb hálózati paraméterek beállítása.
- Miután a hálózat működik, utána következhet bármilyen TCP/IP-t használó szolgáltatás telepítése.

Mielőtt az ethernet interfészeket ellátnánk IP paraméterekkel ajánlatos ellenőriznünk a használt beállításaikat. Némely esetben a hálózati kártya és a switch „nem érti meg automatikusan egymást” (autonegotiation). Ennek a tünete lehet például a link időnkénti szakadása és/vagy a maximális átviteli sebesség töredéke. Ekkor a hálózati kártya manuális beállítása segíthet:

`ethtool -i eth0`
: ethernet kártya driverének lekérdezése

`ethtool eth0`
: ethernet kártya paramétereinek lekérdezése (pl.: link, sebesség, duplexitás, auto-negotiation)

`ethtool -s eth0 speed 100 duplex full autoneg off`
: 100MB/s full duplex üzemmód beállítása, autoneg. kikapcsolása

Amennyiben a driver MII-t (Media Independent Interface) használ, akkor az mii-tool parancs segítségével is megtekinthetjük és beállíthatjuk a paramétereket:

```
mii-tool eth0
```

```
eth0: negotiated 100baseTx-FD, link ok
```
ethernet kártya paramétereinek lekérdezése

```
mii-tool -F 100baseTx-FD eth0
```
100MB/s full duplex üzemmód beállítása

Amennyiben a hálózati kapcsolónk használ VLAN-okkal megvalósított szeparációt, úgy az adott sorszámú virtuális LAN-ba fel kell vennünk a hálózati eszközünket.

```
vconfig add eth0 305
```
az eth0 interfész felvétele a 305-ös sorszámú VLAN-ba

```
vconfig remove eth0.305
```
az eth0 interfész eltávolítása a 305-ös sorszámú VLAN-ból

A továbbiakban a TCP/IPv4-es hálózat esetében a következő eszközök lesznek fontosak:

- `lo` loopback
- `eth0, eth1, ...` ethernet kártyák
- `ppp0, ppp1, ...` pont-pont kapcsolatok
- `wlan0, wlan1, ...` wifi kártyák

8.2. A hálózati beállítása a /etc/network/interfaces alapján

A Debian GNU/Linux a /etc/network/interfaces fájlt használja a hálózati paraméterek beállítására.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
        address 192.168.31.2
        netmask 255.255.255.0
        network 192.168.31.0
        broadcast 192.168.31.255
        gateway 192.168.31.1

allow-hotplug eth1
iface eth1 inet dhcp
```

Az lo interfész helyi kommunikációra használható 127.0.0.1-es IP címmel. Az eth0 interfész esetében fix IP cím (static) beállítására mutatok példát. A fontosabb beállítási lehetőségek:

- `address` IP cím megadása
- `netmask` alhálózati maszk
- `network` alhálózat címe
- `broadcast` az üzenetszórás esetében használt IP cím
- `gateway` az átjáró IP címe

Az előző felsorolásból nem kötelező megadni az alhálózat címét illetve a

broadcast címet, amennyiben ezek automatikusan képezhetők az IP cím és a netmask felhasználásával. Amennyiben adott interfészen keresztül nem érhető el külső hálózat, akkor nem szükséges az átjáró megadása sem.

Az eth1 interfész esetében dinamikusan történik az IP cím kiosztása. Ebben az esetben a hálózati kommunikációhoz szükséges paramétereket egy DHCP szervertől kaphatja meg a számítógép. Ebben az esetben feltétlenül szükséges egy DHCP kliens alkalmazás. pl.: dhclient, dhclient3, pump, ...

A konfigurációs fájl esetében tehát az iface-el kezdődő sorok jelenti egy adott interfészre vonatkozó beállítások elejét. Ezután következhet az interfész neve, majd pedig a címosztály, amely IPv4 esetében az inet. Végezetül pedig az interfész esetében használt módszer az paraméterek hozzárendelésére.
Ezen lehetőségek mellett tetszőleges parancsokat is végrehajthatunk bizonyos szituációk esetében:

- `pre-up,up,post-up` elindítás előtt, elindításkor, elindítás után
- `pre-down,down,post-down` leállítás előtt, leállításkor, leállítás után

Az auto-val kezdődő sorok megadják azokat az eszközöket, amelyeket a gép bekapcsolását követően automatikusan konfigurálni kell és el kell indítani. Az allow-hotplug esetében csak akkor kerül sor az eszköz elindítására, amennyiben a fizikai link létrejött.

Részlet egy VLAN-t is tartalmazó konfigurációs fájlból

```
iface eth1 inet static
        address 172.30.0.1
        netmask 255.255.255.0
        post-up vconfig add eth1 731 && ifconfig eth1.731 up
        pre-down vconfig rem eth1.731
```

A Debian „lenny” alapból tartalmazza az IPv6 támogatást is az IPv4 mellett. Amennyiben erre nincsen szükségünk nyugodtan ki is kapcsolhatjuk. Ha a /etc/modprobe.d/aliases nevű fájl esetében a

```
alias net-pf-10 ipv6
```

sort lecseréljük

`alias net-pf-10 off` -ra, úgy a rendszer újraindítását követően már nem töltődik be automatikusan az ipv6 kernel modul, és rajta keresztül a protokoll támogatása.

A /etc/init.d/networking szkript segítségével leállíthatjuk vagy elindíthatjuk a hálózat működését. Mielőtt a konfigurációs fájlt szerkesztenénk ajánlott leállítani a hálózat működését, majd pedig a módosítások után újra elindítani.

8.3. A hálózat beállításához kapcsolódó parancsok

`ifup, ifdown`

Amennyiben egy hálózati eszköz paraméterei be lettek állítva a /etc/network/interfaces fájlban, de nem történt meg az automatikus indítása, úgy az ifup parancs segítségével az eszköz indítható. Az ifdown parancs

felhasználásával pedig egy működő interfész állítható le.

`ifconfig`
Hálózati interfész konfigurálására alkalmas.
Példák a használatára:

- `ifconfig eth0 up`
  interfész UP állapotba hozása IP paraméterek definiálása nélkül
- `ifconfig eth0 192.168.31.1 up`
  interfész indítása adott IP címmel és alapértelmezett netmask és broadcast címmel
- `ifconfig eth0 192.168.31.1 netmask 255.255.255.192 up`
  interfész indítása adott IP címmel, netmask-al és alapértelmezett broadcast címmel
- `ifconfig eth0 192.168.31.1 netmask 255.255.255.192 broadcast 192.168.31.60 up`
  interfész indítása adott IP címmel, netmask-al, és broadcast címmel.
- `ifconfig eth0 hw ether 00:50:56:00:02:02`
  MAC cím módosítása (az eszköz működése előtt)

Az előzőekben felsorolt lehetőségek mellett még lehetséges az eszközt úgynevezett promiscuous (promisc) módba kapcsolni. Ebben az esetben az eszköz a hálózatban menő összes csomagba képes lesz beletekinteni. Az ifconfig paranccsal interfész(ek) paramétereit is meg tudjuk tekinteni:

```
ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:50:8D:A4:12:BB
     inet addr:192.168.31.2 Bcast:192.168.31.255 Mask:255.255.255.0
     UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
     RX packets:69121567 errors:0 dropped:0 overruns:3 frame:0
     TX packets:110407338 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:1000
     RX bytes:1957228746 (1.8 GiB) TX bytes:3020905526 (2.8 GiB)
     Interrupt:18 Base address:0x4000
```

`route`
A interfészek paramétereinek megadása mellett nagy gondot kell fordítanunk az útválasztásra. A kernel útválasztási táblájának manipulálására használhatjuk a route parancsot.

Példák route parancs használatára:

- `route add -host 172.31.0.1 dev eth1`
  útvonal hozzáadás egy hoszthoz
- `route del -host 172.31.0.1 dev eth1`
  útvonal törlése egy hoszthoz
- `route add -net 172.31.0.0 netmask 255.255.0.0 dev eth1`
  útvonal hozzáadás egy hálózati tartományhoz
- `route del -net 172.31.0.0 netmask 255.255.0.0 dev eth1`
  útvonal törlése egy hálózathoz tartomány esetében
- `route add -net 172.31.0.0 netmask 255.255.0.0 gw 172.31.0.1 dev eth1`

útvonal hozzáadás egy hálózati tartományhoz saját átjáróval

- `route add default gw 192.168.31.1 dev eth0`
  alapértelmezett átjáró megadása
- `route del default gw 192.168.31.1 dev eth0`
  alapértelmezett átjáró törlése

A route parancs felhasználásával tehát az útválasztási tábla manipulálása lehetséges illetve természetesen az aktuálisan érvényben lévő szabályok megtekintése.

`route`

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.31.0    *               255.255.255.0   U     0      0        0 eth0
default         192.168.31.1    0.0.0.0         UG    0      0        0 eth0
```

8.4. A névfeloldás konfigurációja

Abban az esetben, hogyha az IP címet és az egyéb paramétereket helyesen beállítottuk, akkor a hálózatunk már működőképes. Tetszőleges program esetében IP címek felhasználásával tudunk kommunikálni. A gyakorlatban azonban az emberek zöme jóval könnyebben meg tud jegyezni szöveges információt, mint számokat. Főként ezért találták ki a DNS-t (Domain Name System), amely egy nagy elosztott adatbázisnak tekinthető. Mindegyik tartományon belül létezhetnek olyan számítógépek, amelyek a tartományban lévő gépek esetében oda-vissza irányban ellátja a névfeloldás feladatát. Ahhoz, hogy telepített rendszerünk képes legyen név alapján is kommunikálni néhány beállításra szükségünk van.

Egy gépre ún. FQDN (Full Qualified Domain Name) segítségével szokás hivatkozni. Például: www.duf.hu
Egy FQDN-t két részre oszthatunk:

- hosztnévre (www)
- tartománynévre (duf.hu)

A hosztnév beállítása egy adott gép esetében a /etc/hostname felhasználásával történik. Ebben a fájlban az adott gép nevét kell megadnunk.
Amikor a számítógépünkön a Linux indul, akkor a
`hostname --file /etc/hostname` parancs hatására az itt tárolt név automatikusan beállításra kerül.

A névfeloldáshoz kötődő konfigurációs fájlok:

- /etc/hosts

```
127.0.0.1           localhost
193.225.187.80      web-okt.duf.hu      web-okt
```

  Amennyiben az adott hálózatban nincsen, vagy nem működik a névszerver, akkor helyi beállítások alapján is megtörténhet a névfeloldás.
  Ezen fájl esetében három mezőt adhatunk meg tabulátorral elválasztva:
  `IP cím  FQDN vagy hosztnév  alias`
- /etc/host.conf

```
order hosts,bind
```

```
multi on
```

Ez a fájl határozza meg a névfeloldás sorrendjét. Amikor névfeloldást kell végezni, abban az esetben először a /etc/hosts fájlhoz kell fordulni, majd pedig a beállított névszerver(ek)hez. A multi on beállítás annyit jelent, hogy a hoszthoz tartozó összes bejegyzéssel vissza fog térni a resolver.

- /etc/resolv.conf

```
search duf.hu
domain duf.hu
nameserver 193.225.187.65
nameserver 193.225.187.202
```

Ebbe a fájlba kerül a keresési és az alapértelmezett tartomány, illetve a névszerver(ek) IP címe(i). Amikor névfeloldási feladatot kell végezni, akkor azt az operációs rendszer az itt beállított szerverek segítségével próbálja meg megvalósítani.

8.5. A hálózat működésének tesztelése

Miután úgy gondoljuk, hogy minden hálózati beállításunk rendben van célszerű a működés tesztelése. Ezt különböző alkalmazások segítségével tehetjük meg.

`ping`

ICMP echo-request (ping) üzenet küldése egy gépnek, és várja a tőle érkező ICMP választ (echo-reply). Amennyiben ez sikeresen visszaérkezett, akkor a gép hálózati elérése működik. Nem biztos azonban, az hogy bármilyen UDP vagy TCP fölötti szolgáltatás is működik. Olyan is előfordulhat, hogy a gép meg sem kapja az ICMP üzenetet vagy csak egyszerűen nem válaszol rá.

```
ping -c 4 192.168.31.1
PING 192.168.31.1 (192.168.31.1) 56(84) bytes of data.
64 bytes from 192.168.31.1: icmp_seq=1 ttl=63 time=0.488 ms
64 bytes from 192.168.31.1: icmp_seq=2 ttl=63 time=0.508 ms
64 bytes from 192.168.31.1: icmp_seq=3 ttl=63 time=0.512 ms
64 bytes from 192.168.31.1: icmp_seq=4 ttl=63 time=0.518 ms

--- 192.168.31.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.488/0.506/0.518/0.025 ms
```

`traceroute`

Ezzel a paranccsal megvizsgálhatjuk azt, hogy milyen útvonalon keresztül történik meg a célgép elérése. A Debian 5.0 esetében használható a parancs *tracert* néven is.

```
traceroute www.linux.hu
traceroute to www.linux.hu (212.40.125.217), 30 hops max, 40 byte packets
 1  193.225.187.254 (193.225.187.254)  0.554 ms  0.475 ms  0.451 ms
 2  c72-gbeth0-3.dunaujvaros.hbone.hu (195.111.106.217)  0.521 ms  0.439 ms
0.398 ms
 3  c6513-g2-3.vh.hbone.hu (195.111.97.90)  1.688 ms  1.605 ms  1.625 ms
 4  bix.externet.hu (193.188.137.43)  1.638 ms  1.627 ms  1.652 ms
```

```
 5  kata-vlan100.externet.hu (212.40.109.3)  2.016 ms  2.066 ms  2.035 ms
 6  abos.linux.hu (212.40.125.217)  1.708 ms  1.663 ms  1.673 ms
```

A parancs kimenetéből arra is tudunk következtetni, hogy meddig működik megfelelően a hálózat, illetve hol akadnak el az egyes csomagok.

`host, nsloohup, dig`
A névszerver működésének lekérdezésére bármelyik parancs felhasználható. A parancsok segítségével normál, illetve reverse DNS lekérdezések is megvalósíthatók.
`host www.linux.hu`
```
www.linux.hu has address 212.40.125.217
www.linux.hu mail is handled by 10 mail.linux.hu.
```

`host 212.40.125.217`
```
217.125.40.212.in-addr.arpa domain name pointer abos.linux.hu.
```

8.6. A hálózat megfigyeléséhez kapcsolódó parancsok

Nagyon sok olyan alkalmazás létezik, amely kapcsolódik a hálózat működéséhez, a nyitott kommunikációs portok vizsgálatához.

`netstat`
Statisztika a hálózati kommunikációról, illetve a megnyitott portokról.

A netstat parancs fontosabb paraméterei:
- -a az összes socket figyelése
- -i IF csak adott interfész figyelése
- -l a listening (figyelő) állapotban lévő socket-ek figyelése
- -n portszámok mutatása a szolgáltatásnév helyett
- -p socketet használó programok mutatása
- -r az útválasztási tábla mutatása
- -s összegzés (summary)
- -t csak TCP mutatása
- -u csak UDP mutatása
- -w raw socket (nyers adatfolyam)

Amennyiben a netstat parancsot paraméter nélkül használjuk, megmutatja a jelenleg aktív TCP/IP illetve Unix Domain socket kapcsolatokat.

`netstat -t`
```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address      Foreign Address         State
tcp   64717  0   laptop:44029    r24-live.duf.hu:8000    ESTABLISHED
tcp   0      0   laptop:41347    kami:ssh                ESTABLISHED
```

Részlet a netstat -ltp parancs kimenetéből:

```
Active Internet connections (only servers)
Local Address           Foreign Address         State       PID/Program name
localhost.localdo:mysql  *:*                    LISTEN       2723/mysqld
```

```
*:www                  *:*                    LISTEN        12326/apache2
*:ftp                  *:*                    LISTEN        2960/proftpd
*:ssh                  *:*                    LISTEN        2885/sshd
*:smtp                 *:*                    LISTEN        2872/master
```

Az előző listából kiolvasható, hogy a számítógépen nyitva van a www, ftp, ssh, smtp és a mysql alapértelmezett portja. A jobb szélső oszlopban láthatjuk azon alkalmazást, amely a megnyitott portot figyeli. A mysql szolgáltatás csak localhost-ról érhető el, az összes többi pedig bármely interfészen keresztül.

`nmap`

A nmap segítségével port pásztázást lehet végrehajtani egy távoli gépen. Képes a program a távoli gép operációs rendszerének a meghatározására. Egy hálózati támadás előtt ajánlott „felmérni a terepet”. Megnézni az ott futó szolgáltatásokat, az esetleges hibákat, biztonsági réseket figyelembe venni.

```
nmap -A -O 192.168.31.1
```

Részlet a parancs kimenetéből:

```
Interesting ports on 192.168.31.1:
PORT    STATE SERVICE VERSION
21/tcp   open     ftp         ProFTPD 1.3.0
22/tcp   open     ssh         OpenSSH 4.3p2 Debian 9 (protocol 2.0)
25/tcp   open     smtp        Postfix smtpd
80/tcp   open     http        Apache httpd
Uptime 13.218 days (since Thu Mar  6 07:50:14 2008)
Service Info: OSs: Linux, Unix
```

A parancs segítségével érzékeny információk is nyerhetők. Rendszergazdaként természetesen a saját rendszerünk biztonságát tesztelhetjük a segítségével és védelmi módszereket dolgozhatunk ki az egyes támadás típusok kivédésére.

`tcpdump`

A tcpdump egy elterjedt sniffer alkalmazás, amely segítségével megfigyelhetjük a hálózati forgalmat. Képesek vagyunk szűrők segítségével a számunkra hasznos információt tartalmazó csomagokat megkeresni.

Egyszerű példák a használatára:

- `tcpdump host 192.168.31.102`
  Az adott IP címmel rendelkező gép összes forgalmának figyelése
- `tcpdump -x port 80`
  A 80-as portot használó forgalmak mentése hexadecimális formában
- `tcpdump dst 192.168.31.102 and port 80`
  A 192.168.31.102 IP cím 80-as portjának figyelése
- `tcpdump icmp and src 192.168.31.1`
  A 192.168.31.102 forrás IP-vel rendelkező icmp csomagok figyelése.

A parancs rengeteg opcióval rendelkezik. A csomagokban lévő tetszőleges mező alapján megoldható a csomagok elkapása. Emellett a parancs képes fájlba menteni, majd azt offline módon analizálni.

Nagyon sok olyan alkalmazás létezik, amely kapcsolódik a hálózati kommunikáció megfigyeléséhez. Ilyen például a wireshark(ethereal), netcat, cheops, ettercap, nagios, nast, snort, hogy csak a legismertebbeket említsem.

Az inetd és tcpd
A hálózati szolgáltatások indításához kapcsolódik az inetd (internet superserver daemon). Ez egy olyan speciális szolgáltatás, amely képes több hálózati szerver szolgáltatás indítására. Az inetd figyeli a /etc/inetd.conf-ban beállított portokat, majd igény szerint a beállított hálózati szolgáltatásokat elindítja. Ezzel elsősorban a rendszer terheltségét lehet csökkenteni.

A /etc/inetd.conf esetében a következő mezőket különböztethetjük meg:
- service a szolgáltatás neve (/etc/services)
- type a szolgáltatás típusa: stream vagy dgram
- protocol a kommunikációs protokoll neve (/etc/protocols)
- wait/nowait a socket kezelésének módja
- user.group a felhasználó és felhasználói csoport
- server a futtatandó program
- cmdline a futtatandó program argumentumai

Példa inetd.conf beállításokra:
```
ftp stream tcp nowait root /usr/sbin/ftpd in.ftpd -i
talk dgram udp wait nobody.tty /usr/sbin/in.talkd in.talkd
```

A /etc/services fájlban találhatjuk a szolgáltatások neveinek és a hozzájuk tartozó portszámoknak az összerendelése. A fájl mezői:
- service a szolgáltatás neve
- port/protocol a portszám és a szállítási protokoll megadása
- alias álnév az adott szolgáltatásra (nem kötelező megadni)

Példa sorokra a /etc/services fájlból:
```
ssh       22/tcp                        # SSH Remote Login Protocol
www       80/tcp          http          # WorldWideWeb HTTP
nfs     2049/udp                        # Network File System
```

A /etc/protocols nevű fájlban pedig a protokollokra vonatkozó adatok vannak tárolva. A fájl mező:
- protocol a protokoll neve
- azonosító az IP fejlécben a protokoll azonosítója
- alias alnév az adott protokollra

Példa sorokra a /etc/protocols fájlból:
```
icmp    1       ICMP            # internet control message protocol
tcp     6       TCP             # transmission control protocol
udp     17      UDP             # user datagram protocol
```

Ezen konfigurációs fájlok alapján megoldható az, hogy a portszámok helyett a szolgáltatás nevét mutassák meg az egyes alkalmazások. Erre láthattunk példát a netstat parancs esetében.

A tcpd (tcp wrapper daemon) segítségével megvalósítható a szerver szolgáltatások esetében a hozzáférés vezérlése illetve a szolgáltatás naplózása syslog-on keresztül.
Az inetd.conf-ban annyit kell tenni, hogy a futtatandó program nevét le kell cserélni a /usr/sbin/tcpd-re, és az ő argumentumaként kell megadni a futtatandó szerverprogram nevét. Ekkor az alkalmazás indítása tehát nem közvetlenül, hanem a tcpd felhasználásával történik.
Az előző példában szereplő szolgáltatás átalakítása tcpd-vel való futtatásra:

```
ntalk dgram udp wait nobody.tty /usr/sbin/tcpd /usr/sbin/in.ntalkd
```

A hozzáférés szabályozása a /etc/hosts.allow és a /etc/hosts.deny konfigurációs fájlok segítségével valósítható meg.
A fájlok esetében a következő szabályok szerint adhatjuk meg a jogosultságokat:

```
service : host list
```

Ezen kívül használhatunk speciális direktívákat is.

Példák lehetséges beállításokra:
ALL:ALL
in.talkd: ALL EXCEPT LOCAL, .duf.hu
ALL: .duf.hu EXCEPT web-okt.duf.hu
ALL: PARANOID

A jogosultságok értelmezése az alábbi szabályok szerint történik:

- Amennyiben adott szolgáltatásra vonatkozó engedélyezés megtalálható a /etc/hosts.allow nevű fájlban, akkor van hozzáférés, egyébként
- ha a /etc/hosts.deny fájlban meg van tiltva a hozzáférés, akkor nincsen, egyébként pedig van hozzáférés.

9. A hálózati biztonságról dióhéjban

A hálózat biztonság egy összetett fogalom, komoly elméleti háttérrel és vizsgálati módszerekkel rendelkezik. A továbbiakban nem elméletben szeretném megközelíteni a témakört, hanem szeretném a vele kapcsolatos gyakorlati tapasztalataimat bemutatni.
Ahhoz, hogy egy hálózati szolgáltatást nyújtó számítógép mennyire biztonságos, azt nagyban befolyásolják a körülmények. A mi esetünkben a Debian GNU/Linux operációs rendszert futtató számítógép egy hálózatban lát el különböző feladatokat. Így a hálózat biztonságát és az adott gép biztonságát nem lehet élesen különválasztani.

Egy hálózat esetében beszélnünk kell külön-külön az egyes eszközök (szerverek, switchek, routerek, ...) biztonságáról illetve ezeknek a kockázatairól. De nem szabad elfeledkeznünk ezeknek az egymásra gyakorolt hatásáról sem. A továbbiakban csak a Linux szerver biztonsági problémáira, illetve biztonság növelésére fogunk koncentrálni.

A biztonság esetében fontos tényező a fizikai biztonság. Egy szervert olyan helyen kell üzemeltetni, ahol csak az arra jogosultak férhetnek hozzá közvetlenül a hardver eszközökhöz. További fontos dolog, hogy a működés szempontjából fontos egyéb paramétereket (pl.: külső hőmérséklet) állandó szinten tartsunk.
A biztonság másik fontos tényezője az ún. ügyviteli biztonság. Egy hálózati program vagy alkalmazás révén egy konkrét feladatra tudunk megoldást találni. Ez természetesen lehet egy nagyobb rendszer egy kicsi építő eleme is. Mi a továbbiakban feltételezzük, hogy telepítettünk egy Debian GNU/Linux operációs rendszert, és ezt szeretnénk minél biztonságosabban használni és üzemeltetni.

Egy hálózati operációs rendszer esetében a következő tényezők befolyásolják a rendszer biztonságát:

- az operációs rendszer eszköz kezelése, jogosultságai
- az operációs rendszer hálózat kezelése (TCP/IP Stack)
- a használt hálózati szolgáltatások hozzáférései, illetve beállításai
- az alkalmazások által használt kommunikáció módszere
- a rendszert használó személyek (az ember a leggyengébb láncszem)

A biztonság tekintetében vannak általános szabályok:

- minden felhasználónk csak annyi jogosultsággal rendelkezzen, amennyire az ő munkájához feltétlenül szüksége van
- „nehezen kitalálható” jelszavak használata
- csak a feltétlenül szükséges protokollokat használjuk
- csak a feltétlenül szükséges hálózati szolgáltatásokat futtassuk
- korlátozzuk az egyes szolgáltatások elérhetőségét (pl.: hálózati interfész, MAC cím, IP cím, ... alapján) amennyiben ezt a szolgáltatás lehetővé teszi
- használjunk tűzfalakat az elérés korlátozására
- védekezzünk a DoS típusú támadások és az elárasztás (flooding) ellen
- a kommunikációs csatorna titkosítása

A Debian GNU/Linux operációs rendszer biztonsága fokozható az alábbiak segítségével:

- ne adjunk shellt a felhasználóinknak, hogyha az nem feltétlenül szükséges
- kerüljük a setuid bit használatát
- azokat a partíciókat, amelyeket elérhetővé teszünk lehetőség szerint csatlakoztassuk noexec, nosuid, nodev opcióval.
- ne futtassunk felesleges szolgáltatásokat
- amennyiben lehetséges az egyes szolgáltatásokat ne root-ként, hanem normál felhasználóként futtassuk
- alkalmazzunk biztonság növelő patch-eket a kernel esetében (pl.: grsecurity, RSBAC)
- ahol csak lehet, használjunk titkosított kommunikációt
- használjuk a TCP wrapper démont az inetd-vel vagy a xinetd-t, ahol van lehetőség a hálózati elérés korlátozására
- állítsuk be, hogy a szerver alkalmazások csak a működésükhöz szükséges fájlokhoz férjenek hozzá, semmi máshoz ne
- figyeljünk oda a puffer túlcsordulásos hibákra, használjunk ellenük védelmi módszereket (PaX, Exec Shield, ...)
- futtassuk az egyes szolgáltatásokat saját „chroot környezeten” belül
- csak hivatalos forrásból származó, digitálisan aláírt csomagokat használjunk.

Az előzőekben felsoroltak mellett fontos feladatok:

- a szolgáltatások megfelelő naplózása
  A naplófájlokat célszerű külön partícióra tenni. A syslog-ng segítségével megoldható a központosított naplózás, ahol több szerver naplóbejegyzései egy erre a célra kijelölt szerveren tárolódnak.
- a naplófájlokat monitorozni kell, amennyiben adott helyről megpróbálják a rendszer működését gátolni, vagy feltörni, akkor meg kell tenni az ellenlépéseket.
- fontos, hogy készítsünk biztonsági mentéseket a működő rendszerről, hogy a későbbiekben a mentésből vissza tudjuk állítani a rendszerünket. A redundáns adattárolás nem pótolja a biztonsági mentést, hiszen ha valakinek sikerül a rendszert feltörni, abban az esetben tetszőleges adatot törölhet. Ebben az esetben természetesen a RAID sem segít.
- időnként készítsünk biztonsági elemzést a rendszerünkről. Vizsgáljuk meg, hogy újabb technológiákkal fokozható-e a rendszerünk biztonsága.
- a használt alkalmazások esetében figyeljünk a biztonsági riasztásokra. Vizsgáljuk meg, hogy bennünket is érinthet-e a napvilágot látott biztonsági probléma.
- rendszeresen végezzünk szoftver frissítést, ezáltal biztosítsuk, hogy rendszerünk egyes komponensei naprakészek legyenek.

Biztonsági szempontból nagyon fontos a felhasználók megfelelő célra használják fel a kapott jogosultságaikat. Amennyiben a felhasználók képzetlenek, hiába bármilyen szuper biztonságos rendszer, akkor az ő jogosultságaikon keresztül a támadók könnyebben férhetnek hozzá a rendszerhez. Ezt felhasználva a biztonság jóval egyszerűbben kijátszható. Másrészt pedig könnyebben bedőlhetnek az internet felől érkező fenyegetéseknek, amelynek manapság már nem a tudás fitogtatása, hanem a haszonszerzés a legfőbb célja.

10. Fontosabb hálózati szolgáltatások Debian GNU/Linux operációs rendszerben

A hálózati operációs rendszerek legfontosabb feladata, hogy különböző szolgáltatásokat nyújtsanak, amelyek felhasználásával egy informatikai rendszer adott feladatot el tud látni. A továbbiakban a legnépszerűbb ilyen alkalmazásokat fogom bemutatni. Minden feladatra sokféle alkalmazás létezik. Az, hogy egy adott feladatra melyik alkalmas leginkább, az függ természetesen az alkalmazás lehetőségeitől, de nem szabad figyelmen kívül hagyni a rendszergazda személyes tapasztalatait. Ha valaki jól ismer egy adott programot, akkor biztonságosabban tudja megoldani az adminisztrálását, a nem várt problémákat is valószínűleg hamarabb meg tudja oldani.

10.1. Az OpenSSH

Az OpenSSH az OpenBSD projekt által fejlesztett alkalmazás. Hivatalos weboldala a http://www.openssh.com. A segítségével titkosított módon tudunk kapcsolatot kialakítani két számítógép között. Az alkalmazás kliens-szerver módon működik. Amennyiben egy Linux-ot futtató számítógépre telepítjük a kiszolgálót (sshd), abban az esetben a következő lehetőségeink lehetnek (a konfiguráció függvényében):

- Titkosított távoli bejelentkezés a szerverre az ssh protokoll felhasználásával. A bejelentkezés tetszőleges operációs rendszer alól megtehető, amelyre létezik ssh kliens program. Linux operációs rendszerben az `ssh` nevű parancs szolgál a távoli bejelentkezésre. Míg „ablakozós” operációs rendszerből például a putty.exe felhasználásával oldható meg a távoli bejelentkezés. A távoli rendszeradminisztrálás egyik fontos eszköze.
- Fájl átvitel lehetősége scp (secure copy) protokollon keresztül. A fájl átvitel esetében az sshd elindítja a szerveren a /usr/bin/scp parancsot, amely felhasználásával történik a fájl(ok) kiszolgálása. Linux operációs rendszer esetében az scp parancs segítségével oldható meg a fájl átvitel. „Ablakozós” operációs rendszerben erre a feladatra használható a WinScp.exe alkalmazás.
- Fájl átvitel lehetősége sftp (secure ftp) protokoll felhasználásával. Ennek a gyakorlati megvalósítását a /usr/lib/openssh/sftp-server végzi, amely az sshd egy alrendszerének tekinthető. Linux operációs rendszerben az sftp parancs használatával megoldható a fájl átvitel, míg az előző bekezdésben említett WinScp képes sftp alapú átvitelre is az scp használata mellett.
- Megvalósítható a segítségével a port forwarding. Ez annyit jelent, hogy képesek vagyunk egy önmagában titkosítást nem használó protokollt titkosított csatornán átvinni.
- Az ssh képes az X11 forwarding-ra. Ez annyit jelent, hogy a távoli gépen futó grafikus alkalmazás képét képesek vagyunk átvinni titkosítottan egy másik gépre.

Az OpenSSH egy nyílt forrású implementációja az ssh protokollnak. Számos elterjedt titkosítási algoritmust támogat. Ilyen például a DES, Blowfish, 3DES, AES, és ezeknek különböző méretű kulcsokon alapuló változataik.

A Debian GNU/Linux 5.0 az OpenSSH 5.1-et tartalmazza. Az openssh-client csomag tartalmazza a kliens alkalmazásokat, az openssh-server nevű csomag pedig a szervert. Amennyiben mindkettőt szeretnénk telepíteni, akkor telepíthetjük az ssh nevű csomagot. Ekkor automatikusan települni fog mind az ssh kliens, mind pedig a szerver a függőségek miatt.

10.1.1. Az SSH protokolljai

Az ssh működése a nyilvános kulcsú titkosításon alapul. Ez annyit jelent, hogy van egy kulcspárunk, amely egy titkos kulcsból (private key) és egy nyilvános kulcsból (public key) áll. Ezeknek fontos szerepük van a kommunikációs csatorna felépítésekor.

Az ssh esetében kétféle protokollt használhatunk:

SSH protocol version 1.
Ebben az esetben a szerver rendelkezik egy 1024 bites RSA kulcs párral, amelyet a /etc/ssh könyvtárban tárol (HostKey). Emellett rendelkezik egy másik RSA kulcs párral, amelyet alapértelmezésként óránként újragenerál.
Kapcsolatfelvételkor a következő történik:

- a kliens kapcsolódik a szerver megfelelő portjához (alapértelmezésként a 22-s TCP port)
- a szerver elküldi a publikus kulcsait a kliensnek
- a kliens ellenőrzi, hogy a kulcsok tárolva vannak-e, ha igen akkor összehasonlítja őket a tárolt változattal
- a kliens generál egy 256 bites véletlen számot, amelyet a publikus kulcsokkal titkosítva visszaküld a szervernek
- a szerver a privát kulcsa segítségével visszafejti a küldött véletlen számot, majd ezt a véletlen számot használják a továbbiakban a kommunikáció paramétereinek az egyeztetésére
- a kliens kiválasztja a titkosítás algoritmusát, amelyet a „normál adatforgalom” esetében használnak. A DES, Blowfish vagy a 3DES hagyományos titkosítási módszerek támogatottak, amely közül a 3DES az alapértelmezett.
- a kliens hitelesítése (authentication)

SSH protocol version 2.
Ebben az esetben a szerver rendelkezik egy RSA és egy DSA kulcs párral, amelyet a /etc/ssh könyvtárban tárol (RSA,DSA HostKey).
A különbség annyi az 1-es protokollhoz képest, hogy itt a kapcsolat felvétel nem véletlen szám alapján történik. Az ún. Diffie-Hellman kulcs csere protokoll segítségével történik meg egy kezdeti 128 bites kulcsnak a létrehozása (session key). Ezt mindegyik fél ismerni fogja, így nem fog titkosítatlan csatornán keresztül közlekedni. Ezután a kliens és a szerver megállapodnak a csatorna titkosítás módszerében. Ebben az esetben szimmetrikus titkosítást használnak AES, Blowfish, 3DES, CAST128, Arcfour módszerek valamelyikét, amelyet a szerver által felajánlottak közük a kliens kiválaszt. Ezután következhet a hitelesítés.

Manapság általában csak az újabb, a 2. protokoll a használatos. A másik támogatását ajánlott tiltani.

10.1.2. Az SSH fontosabb hitelesítési lehetőségei

Miután a kapcsolatfelvétel megtörtént a kliensnek valamilyen módon hitelesítenie kell magát. Amennyiben ez sikeres, csak abban az esetben szabad shellt adni az adott felhasználónak (vagy a távoli gépen a megfelelő parancsot futtatni). A hitelesítési módszerek után zárójelben látható, hogy melyik protokoll verzió esetében használhatók.

RhostsAuthentication (1):
Amennyiben az adott gép megtalálható a /etc/hosts.equiv vagy /etc/ssh/shosts.equiv fájlokban, illetve adott felhasználóként a .rhosts vagy a .shosts fájlokban, akkor jelszó megadás nélkül be lehet engedni.

RhostsRSAAuthentication (1):
HostbasedAuthentication(2):
Ugyanaz mint az előző, annyi a különbség, hogy ebben az esetben a kliens-nek rendelkezni kell a szerver RSA publikus kulcsával.

RSAAuthentication (1):
PubkeyAuthentication (2):
RSA kulcspár alapján történik a hitelesítés ebben az esetben. Az ssh-keygen parancs segítségével létrehozhatunk egy kulcspárt. Alapértelmezésként a .ssh/id_rsa lesz a privát, a .ssh/id_rsa.pub pedig a publikus kulcsunk. Amennyiben a publikus kulcsunkat hozzáadjuk a szerveren a .ssh/authorized_keys nevű fájlhoz, akkor a privát kulcsunk segítségével megtehetjük a bejelentkezést. Ekkor a generáláskor megadott jelszó felhasználásával tudunk bejelentkezni. Nem kötelező a jelszó megadása a kulcspár létrehozásakor.

PasswordAuthentication (1,2):
A rendszerbeli felhasználónév és jelszó alapján történik meg a hitelesítés.

Az előzőekben felsorolt lehetőségek mellett van arra lehetőség, hogy Kerberos szerver felhasználásával történjen meg a hitelesítés.

10.1.3. Az SSH-hoz kapcsolódó fontosabb parancsok

/usr/sbin/sshd	az SSH kiszolgáló
/usr/lib/openssh/sftp-server	az sftp kiszolgálásáért felelős alrendszer
/etc/init.d/ssh	a ssh indításáért felelős script
/usr/bin/ssh	az SSH kliens program
/usr/bin/scp	scp parancs a fájl átvitelre
/usr/bin/sftp	titkosított ftp kapcsolat ssh felhasználásával
/usr/bin/ssh-keygen	kulcspár létrehozása

10.1.4. Az SSH kiszolgáló néhány konfigurációs lehetősége

Az OpenSSH kiszolgáló a /etc/ssh/sshd_config fájl segítségével konfigurálható. A következőkben néhány fontosabb direktívát mutatok be:

```
Port 22
```
Melyik TCP porton „hallgatózzon”?

```
ListenAddress 172.31.0.1
```
Mely IP címen legyen elérhető. Amennyiben több interfészünk is van, akkor megadhatjuk, hogy melyeken történjen a port megnyitása. Több is szerepelhet belőle.

```
Protocol 2
```
Csak az SSH Protocol Version 2. legyen támogatva.

```
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
```
A telepített szerver privát kulcsait tartalmazó fájlok.

```
PermitRootLogin yes
```
A root-ként való bejelentkezés engedélyezése. Biztonsági kockázatot jelenthet, ha „PasswordAuthentication” engedélyezett. Ilyenkor lehetséges távolról a rendszergazdai jelszó próbálgatása.

```
AllowGroups oktato
AllowUsers user@host
DenyGroups hallgato
DenyUsers winokt
```
Bejelentkezés engedélyezése vagy tiltása csoport- vagy felhasználónév alapján.

```
RSAAuthentication yes
PubkeyAuthentication yes
PasswordAuthentication yes
RhostsRSAAuthentication no
HostbasedAuthentication no
```
Adott hitelesítési módszerek támogatása.

```
SyslogFacility AUTH
LogLevel INFO
```
A naplózás a /var/log/auth fájlba történjen INFO „naplózási szinttel”.

```
PermitEmptyPasswords no
```
Az üres jelszavak nem engedélyezettek.

```
MaxAuthTries 3
```
A hitelesítési próbálkozások száma.

```
StrictModes yes
```
Bejelentkezés előtt a saját könyvtár jogosultságainak az ellenőrzése.

`Subsystem sftp /usr/lib/openssh/sftp-server`
Az sftp alrendszer engedélyezése.

`UsePAM yes`
A PAM használatának engedélyezése.

10.1.5. Az SSH kliens használata

Linux operációs rendszerben az ssh parancs segítségével megtehetjük a távoli bejelentkezést illetve a port forward-ot. Az ssh kliens konfigurálása az alábbi fájlok segítségével történhet:
rendszer szinten: /etc/ssh/ssh_config
felhasználói szinten: ~/.ssh/config

Példa konfigurációs fájlra:
```
Port 22
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication yes
User user1
StrictHostKeyChecking yes
```

Alapértelmezésként az ssh parancs ugyanazzal a felhasználóval próbál meg távolra bejelentkezni, amellyel a helyi rendszerben jelen vagyunk. A User direktívával ezt definiálhatjuk felül. A `StrictHostKeyChecking` direktíva segítségével azt adhatjuk meg, hogy szigorúan ellenőrizze a ~/.ssh/known_hosts fájlban a szerver publikus kulcsát. Ha az nem egyezik, akkor ne engedélyezze a hitelesítést.

Példák ssh parancs használatára:
- `ssh 192.168.31.1`
  bejelentkezés az adott gépre az aktuális user-el
- `ssh -l alma 192.168.31.1`
- `ssh alma@192.168.31.1`
  bejelentkezés az adott gépre alma nevű felhasználóként
- `ssh -i pr_rsa.key -p 1022 -l alma 192.168.31.1`
  bejelentkezés az adott gépre alma nevű felhasználóként a pr_rsa.key RSA kulcs felhasználásával az 1022-es porton keresztül
- `ssh -f -N -L 2222:192.168.31.1:80 alma@192.168.31.1 true`
  a helyi 2222-es port forward-olása a távoli gép 80-as portjára az alma nevű felhasználóval
- `dd if=/dev/sda1 | ssh -l root 192.168.31.1 'cat > sda1_save'`
  egy partíció mentése ssh használatával egy távoli gépre

10.1.6. Fájlátvitel scp és sftp használatával

Az scp parancs általános szintaxisa:

```
scp [Options] [[user1@host1:]file1] [[user2@host2:]file2]
```

Példák scp parancs használatára:

- `scp alma@192.168.31.1:a* .`
  A távoli géphez alma nevű felhasználóval kapcsolódik és a felhasználó saját könyvtárából az a-val kezdődő fájlokat másolja át a helyi gép aktuális könyvtárába
- `scp -pr 192.168.31.1:/var/www /var/.`
  A távoli géphez a jelenlegi felhasználó azonosítóval kapcsolódik és onnét a /var/www könyvtár tartalmát másolja át a /var/www-ként a helyi gépre a fájl jellemzők megőrzésével
- `scp -p *tgz 192.168.31.1:.`
  Az aktuális könyvtárban lévő tgz-re végződő fájlokat másolja át a távoli gépre a jelenlegi felhasználói azonosítóval a felhasználó saját könyvtárába
- `scp -p /var/www/index.htm root@192.168.31.1:/var/www/index.htm`
  A /var/www/ -ben lévő index.htm nevű fájlt másolja át a távoli gépre ugyanilyen elérési úttal root-ként csatlakozva
- `scp -p -P 2222 *.html root@192.168.31.1:/var/www/.`
  Az aktuális könyvtárban lévő összes .html-re végződő fájlt másolja át a távoli gépre a /var/www könyvtárba root-ként csatlakozva a 2222-es tcp porton keresztül

Az sftp parancs használata hasonlít a hagyományos ftp parancshoz. Az scp-vel ellentétben ez egy önálló parancsfelületet biztosít. A parancs felületen belül az ftp kliensekben megszokott parancsokhoz hasonlóakat használhatunk.

Az sftp parancs használata:

```
sftp [Options] [[user@]host[:dir[/]]]
```

```
sftp alma@192.168.31.1
Connecting to 192.168.31.1...
alma@192.168.31.1's password:
sftp> ls
```

Miután sikeresen kapcsolódtunk a szerverhez, a kliens felületen belül a legfontosabb lehetőségek:

- `?, help` segítség kérés
- `ls, lls` listázás a távoli, helyi gépen
- `pwd, lpwd` aktuális könyvtár megtekintése a távoli, helyi gépen
- `cd, lcd` könyvtár váltás a távoli, helyi gépen
- `get, put` több fájl letöltése, feltöltése
- `rm, rmdir` távoli fájl, könyvtár törlése
- `quit` kilépés

Az OpenSSH segítségével lehetőség van távoli adminisztrációra illetve fájl átvitelre. Amennyiben csak rendszeradminisztrálásra használjuk korlátozzuk a szolgáltatás elérését IP cím alapján, illetve ne az alapértelmezett porton futtassuk. Amennyiben csak kulcspár alapján történő hitelesítést engedélyezzük úgy a távolról történő jelszó próbálgatásoknak is gátat szabhatunk. De természetesen ebben az esetben is fennáll annak a veszélye, hogy valamilyen szoftver hibát kihasználva visszaéljenek a rendszerrel. Tehát használjuk körültekintően, nehogy másnak is ezáltal legyen lehetősége a rendszer feltörésére. Sokszor éppen a biztonságosnak hitt szolgáltatások jelentik a legnagyobb veszélyt.
Amennyiben ezen keresztül szeretnénk megvalósítani a fájl átvitelt, akkor ajánlott, hogy a normál felhasználók csak a saját könyvtárukat tudják elérni. Erre az 5.1-es verzió esetében ott van a `ChrootDirectory` konfigurációs lehetőség. Amennyiben az itt megadott könyvtár esetében létezik egy megfelelő minimális könyvtárstruktúra, akkor a felhasználók csakis ezen könyvtár tartalmát tudják majd elérni. Az scponly csomag (parancs) felhasználásával megoldható, hogy csak scp elérés legyen, shellt ne kaphasson egy távoli felhasználó.

10.2. Web szerver használata Linuxon

10.2.1. Az Apache jellemzői

Manapság talán a leggyakrabban használt hálózati szolgáltatás a web kiszolgálás. Nagyon sok informatikai rendszer esetében egyértelmű követelmény, hogy legyen web-es felülete (is). Ahhoz, hogy a Linux-ot futtató számítógépünk ezt a feladatot el tudja látni, ahhoz szükséges, hogy valamilyen web kiszolgáló programot telepítsünk. Napjainkban a legelterjedtebb webszerver az Apache, amely a 2008-as Netcraft felméres szerint körülbelül a webszerverek 50%-n található. Linux alatt egy másik ígéretes kezdeményezés a lighttpd. Bár képességeiben még alulmarad az Apache-hoz képest, de gyorsabb kiszolgálást tesz lehetővé. Tehát a feladattól függ, hogy melyiket célszerű választani. Az kétségtelen, hogy az Apache jóval nagyobb felhasználói táborral rendelkezik.

Az Apache Software Foundation több nyílt forrású szoftvert is fejleszt, illetve támogat. Ezek közül az egyik legismertebb a web szerver. A projekt hivatalos web oldala a http://httpd.apache.org.

Az Apache webszerver legfontosabb jellemzői:

- Az utóbbi 10-12 év domináns webszervere. Népszerűségére jellemző, hogy folyamatosan 50-70%-át a webszervereknek az Apache tette ki.
- Jól bevált, jelentős hagyományai vannak.
- A web kiszolgálás TCP/IP-re épülő szolgáltatás. Alapértelmezésként a 80-as TCP portot használja a HTTP, a 443-as portot a HTTPS.
- Jelenleg három fő verziója van használatban: 1.3, 2.0, és a 2.2-es.
- Nyílt forráskódú, a legtöbb Linux disztribúcióban benne van.
- Kis erőforrás igényű, egy manapság elavultnak tekinthető Pentium II kategóriájú számítógép megfelelő lehet a webszerver feladatra. (Ez természetesen függ a felhasználók számától és az alkalmazás számításigényétől.)
- Az Apache sokoldalúan konfigurálható, rengeteg konfigurációs beállítással

rendelkezik.

- A webszerver felépítése moduláris. Amikor forráskódból fordítjuk a webszervert választhatunk, hogy milyen modulokat (támogatásokat) szeretnénk statikusan belefordítani a webszerver programba (httpd), illetve melyeket dinamikusan betölthető modulként. Ez utóbbi igény szerint a webszerver indulásakor betölthető és használható.
- A különböző script nyelveket maximálisan támogatja: Perl, PHP, Python, ... Ezen programozási nyelvek segítségével megvalósítható a dinamikus weboldal létrehozás, adatbázis alapú webes rendszerek létrehozása.
- Képesek vagyunk SSI (Server Side Include) használatára. Ennek a segítségével például mindegyik web oldalunkhoz tartalmába adott fájlok tartalmát be tudjuk illesztetni.
- Képesek vagyunk külső programokat a CGI (Common Gateway Interface)-n keresztül végrehajtani.
- Van lehetőség a Virtualhost használatára. Ami annyit jelent, hogy egy program képes ellátni több szerver kiszolgálását. Vagyis például ugyanaz az Apache szolgálja ki a mondjuk a www.a.hu és a www.b.hu szervereket.
- Sokféle operációs rendszert támogat, elsősorban Linux vagy Unix operációs rendszerben használatos.

Az Apache verziók legfontosabb jellemzői.

1.3-as verziók (a jegyzet írásakor az 1.3.42-es a legújabb)

- kizárólagosan process alapú működés
- több független gyermekfolyamat révén valósul meg a webkiszolgálás
- alapból nem tartalmaz SSL (Secure Socket Layer) támogatást

2.0-as verziók (a jegyzet írásakor az 2.0.63-as a legújabb)
2.2-es verziók (a jegyzet írásakor az 2.2.15-ös a legújabb)

- APR (Apache Portable Runtime): általános API és lib-ek az alkalmazások számára
- különböző MPM-ek (Multi Processing Modules) használata. A hagyományos process alapú modell mellett különböző szál alapú modellek használata (pl.: worker) használatára is van lehetőségünk.
- a 2GB-nál nagyobb fájlok támogatása (csak a 2.2-ben)
- alapból tartalmaz SSL támogatást (mod_ssl)
- többféle protokoll támogatása a HTTP mellett
- speciális szűrő (filter) modulok használata a kimenet vizsgálatára

10.2.2. Az Apache telepítése forráskódból

A továbbiakba a 2.2-es ággal fogok foglalkozni. Az Apache 2.2.9 benne van a disztribúcióban és csomagból is telepíthető. Az oktatás során a saját program fordítást és telepítést az Apache segítségével szoktam bemutatni évek óta. Így a jegyzetben is először ezzel a módszerrel fogom bemutatni a webszerver telepítését.

Ahhoz, hogy Apache-ot tudjunk fordítani, abban az esetben a kernel fordításhoz hasonlóan rendelkeznünk kell bizonyos csomagokkal. Amire biztosan szükségünk lesz az a libc6-dev, gcc és make nevű csomagok. Ezen felül a beállított

támogatások függvényében még szükségünk lehet néhány csomag telepítésére (pl.: zlib1g-dev, libssl-dev).

Miután az Apache forráskódját letöltöttük és kicsomagoltuk, első lépésként a ./configure scriptet kell futtatnunk. Ennek a segítségével beállíthatjuk, hogy a program mely összetevőire van szükségünk.

Miután beléptünk a forráskódot tartalmazó könyvtárba,
`./configure --help | less` parancs segítségével tekinthetjük át a konfigurálás különböző lehetőségeit.

Példa egy „./configure” parancsra:
```
./configure --prefix=/usr/local/httpd-2.2.15 \
--enable-modules=all \
--enable-mods-shared=all \
--enable-so \
--enable-cache \
--enable-cgi \
--enable-ssl \
--with-ssl=/usr/include \
--with-mpm=worker
```

Ebben a példában úgy konfigurálom a fordítót, hogy a /usr/local/httpd-2.2.15-ös könyvtárba kerüljenek majd a program különböző részei. Az összes modult engedélyezem, és mindegyiket dinamikusan. Ezen kívül engedélyezem, hogy legyen dinamikusan modul betöltés, az oldalak gyorsító tárba helyezését, a CGI végrehajtást és az SSL bővítményt. Ebben az esetben a worker MPM-et fogja használni a létrejött szerver.

Ahhoz, hogy az előző parancs sikeresen lefusson, ahhoz szükséges:
- zlib1g-dev csomag telepítése mod_deflate miatt.
- libssl-dev csomag telepítése mod_ssl miatt.

Miután a forráskód konfigurációja sikeresen lefutott, következhet az alkalmazáshoz tartozó binárisok, lib-ek és egyéb fájlok fordítása.
```
make [-j2]
```

Majd az elkészült fájlok telepítése a célhelyre (/usr/local/httpd-2.2.15). Alapértelmezésként a /usr/local/apache2 könyvtár a használatos. Amennyiben verzió frissítést végzünk, akkor ajánlatos, hogy a /usr/local/apache2 szimbolikus link legyen, amely a használt verziónak megfelelő könyvtárra mutat. Így a konfigurálást is tudjuk verziótól függetlenül végezni, amely megkönnyíti a verziófrissítést.
```
make install
```

10.2.3. Az Apache könyvtárai és parancsai

A telepítést követően a következő könyvtárak és fájlok jönnek létre a célhelyen:
- bin az Apache-hoz tartozó binárisok
  - ab Apache benchmark
  - apachectl az Apache indítása, leállítása, újraindítása

    - apxs — APache eXtenSion tool, bővítmények fordítása, telepítése
    - dbmmanage — DBM adatbázisban tárolt felhasználók kezelése, amely HTTP Basic hitelesítéshez használható
    - htdbm — DBM adatbázisban tárolt felhasználók létrehozás, amely HTTP Basic hitelesítéshez használható
    - htdigest — szövegfájlban tárolt felhasználók kezelése a HTTP Digest hitelesítéshez
    - htpasswd — szövegfájlban tárolt felhasználók kezelése a HTTP Basic hitelesítéshez
    - httpd — az kiszolgáló démon
    - httxt2dbm — hitelesítési információk szövegfájlból és DBM-be való átalakítása
    - logresolve — névfeloldás naplófájlban lévő IP címek alapján
    - rotatelogs — a naplófájlok „forgatása” idő vagy méret alapján
- build — a fordítási folyamat során használt szkriptek és makrók
- cgi-bin — a CGI-k alapértelmezett helye, ezen könyvtár tartalma érhető el a http://host/cgi-bin hivatkozás segítségével. Ahol a host az adott gép FQDN-je vagy IP címe.
- conf — konfigurációs fájlok
    - httpd.conf — a webszerver fő konfigurációs fájlja
    - mime.types — az egyes MIME típusok összerendelése az egyes fájl kiterjesztésekkel
- error — az egyes HTTP hibák esetén megjelenítendő fájlok
- htdocs — a webes dokumentumok helye. Amilyen fájlokat ebbe a könyvtárba rakunk, azok lesznek elérhetők weben keresztül. Ez a könyvtár a jelenti a webes dokumentumok gyökerét.
- icons — ikonok (pl.: listázáshoz)
- include — header fájlok
- lib — lib-ek
- logs — naplófájlok
    - access.log — a hozzáférések naplózása
    - error.log — a hibák naplózása
- man — parancsokhoz tartozó man oldalak
- manual — Apache dokumentáció
- modules — az elérhető dinamikus modulok

Amennyiben nem forráskódból telepítjük az Apache-ot, abban az esetben az előzőekben felsorolt könyvtárak és fájlok más helyen találhatók.
Például:

konfigurációs fájlok: /etc/apache2
web-es dokumentumok helye: /var/www
CGI-k helye: /usr/lib/cgi-bin

Természetesen a konfiguráció ugyanazon direktívák segítségével történhet, csak más helyen tudunk a program működésébe beavatkozni.

10.2.4. Az Apache néhány konfigurációs direktívája

Az Apache esetében konfigurálás szempontjából a fő konfigurációs fájl mellett egyéb konfigurációs fájlokat is használhatunk. Az include direktíva segítségével tetszőleges nevű külső konfigurációs fájl felhasználhatunk a szerver működésekor.
Amikor egy szervert konfigurálunk megkülönböztethetünk:

- globális konfigurációt a globális konfigurációs fájl felhasználásával tetszőleges direktívát megadhatunk. Hátránya, hogy bármilyen módosításkor a szerverrel újra kell olvastatni a konfigurációs fájlt.
- helyi konfigurációt (Alapértelmezésként .htaccess néven). Ebben a fájlban adott könyvtárra vonatkozó helyi beállításokat adhatunk meg. Aminek a módosítását előzetesen engedélyeztük (AllowOverride), annak a felüldefiniálását tehetjük meg helyileg.

A globális konfiguráció esetében léteznek úgynevezett blokk direktívák. Ezek segítségével bizonyos szempont alapján csoportosíthatjuk a direktívákat, adott feltétel vagy valamilyen fájl jellemző alapján.
A konfigurálás során használható fontosabb blokk direktívák:

- `<IfDefine SSL> ... </IfDefine>` a webszerver indításakor a -D kapcsló után definiált érték figyelembe vétele
- `<IfModule module> ... </IfModule>` adott dinamikus modul létezésekor alkalmazandó beállítások
- `<VirtualHost ip_address:port> ...</VirtualHost>` adott „virtuális hoszt-ra” vonatkozó beállítások
- `<Directory path> ... </Directory>` adott könyvtárra alkalmazandó beállítások elérési útvonal alapján
- `<DirectoryMatch regexp> ... </DirectoryMatch>` adott könyvtárra alkalmazandó beállítások reguláris kifejezés alapján
- `<Files filename> ... </Files>` adott fájl(ok)-ra vonatkozó beállítások fájlnév alapján (shell minta használható a fájlnévben)
- `<FilesMatch regexp> ... </Filesmatch>` adott fájl(ok)-ra vonatkozó beállítások reguláris kifejezés alapján
- `<Location URL> ... </Location>` adott URL-re vonatkozó beállítások név alapján (shell minta használható a névben)
- `<LocationMatch regexp> ... </LocationMatch>` adott URL-re vonatkozó beállítások reguláris kifejezés alapján
- `<Limit method> ... </Limit>` adott HTTP metódus(ok)ra vonatkozó beállítások név alapján
- `<LimitExcept method> ... </LimitExcept>` adott HTTP metódus(ok)ra nem vonatkozó beállítások név alapján

A blokkdirektívák tehát meghatározzák, az adott konfigurációs paraméterek érvényességi körét. A következőkben kiemelnék néhány konfigurációs lehetőséget. Az, hogy adott direktíva pontosan melyik blokkon belül lehetséges, az az Apache dokumentációban megtalálható. Ez utóbbi HTML formában is elérhető, és a /manual hivatkozáson keresztül a webszerver gyökerén belül.

`ServerRoot /usr/local/apache2`
Az Apache telepítésének a helyét adja meg. A különböző konfigurációs fájlok

helyét relatívan is megadhatjuk, amely esetében az itt definiált könyvtár lesz az alapértelmezés.

```
ServerType standalone     # vagy inetd
```
A webszerver futtatása önállóan vagy az inetd felhasználásával történjen.

```
ServerName www.apache_host.hu
```
Itt adhatjuk meg, hogy a szervert milyen név vagy IP cím segítségével érhessük el.

```
Listen 80
```
A 80-as TCP porton hallgatózzon a webszerver.

```
User www-data
Group www-data
```
A webszerver milyen felhasználó és milyen felhasználói csoport nevében működjön. Ahhoz, hogy a webszerver képes legyen dokumentumokat kiszolgálni, fontos, hogy legyen olvasási jogosultsága az adott fájlokra vonatkozóan. Ellenkező esetben nem lesz képes a fájlok elküldésére.

```
DocumentRoot /var/www/htdocs
```
Egy webszerver segítségével általában nem szeretnénk az összes fájlunkat elérhetővé tenni. Elég az, hogy csak bizonyos könyvtáron belüli fájlokat tesszük elérhetővé HTTP-n keresztül.

```
ServerAdmin webmaster@www.apache_host.hu
```
A webszerver karbantartásáért felelős személy vagy csoport e-mail címe.

```
LoadModule dir_module modules/mod_dir.so
LoadModule php5_module modules/libphp5.so
```
Dinamikus modul betöltése a szerver indításakor. Miután egy modul betöltésre került, akkor az általa nyújtott lehetőségek és bizonyos direktívák elérhetőek lesznek.

```
Timeout 300
```
Egy felépült kapcsolat időkorlátja.

```
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
```
Több kapcsolat engedélyezése adott IP címről és az erre vonatkozó maximális kapcsolatszám és időkorlát.

```
LogLevel warn
```
A naplózás szintjének megadása. A lehetséges értékek, amelyek esetében balról jobbra haladva egyre csökken a naplózott események száma: debug, info, notice, warn, error, crit, alert, emerg.

```
ErrorLog logs/error.log
```
Amennyiben olyan kérés érkezik a webszerverhez, amelyet valamilyen probléma miatt nem tud kiszolgálni, akkor ide bejegyzi a kérést és az arra vonatkozó fontosabb információkat.

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access.log common
```
Az elérések naplózásának formátuma és a naplófájl megadása. A %h jelenti például a távoli gép IP címét, a %t kérés idejét.

```
ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
```
Az egyes HTTP hibákhoz fájlokat társíthatunk, amely segítségével tetszőleges tartalmat megjeleníthetünk a hibák függvényében.

```
DefaultType text/plain
AddDefaultCharset UTF-8
```
Alapértelmezett MIME típus és karakter készlet.

```
ServerTokens Prod
```
A webszerver verziójának rejtése, csak az Apache név mutatása.

```
TraceEnable off
```
A HTTP TRACE metódusának tiltása.

```
ServerSignature Off
```
A szerver által generált listák alján szereplő sor kikapcsolása.

```
UserDir public_html
```
Felhasználóként a saját webtartalom elhelyezésének könyvtára a „HOME” könyvtáron belül. Amennyiben ez engedélyezett, akkor például a „tux” nevű felhasználó saját oldalai a `http://www.apache_host.hu/~tux` címen lesznek elérhetőek.

```
DirectoryIndex index.html index.shtml index.php
```
Amennyiben egy könyvtár esetében létezik az itt megadott nevű fájlok valamelyike, akkor olyan HTTP kérések esetében, amelyek könyvtár hivatkozással végződnek ezen fájlok lesznek automatikusan elküldve balról jobbra történő prioritással.

```
AccessFileName .htaccess
```
Megadja annak a fájlnak a nevét, amely segítségével helyi beállítások tehetők adott könyvtárakra vonatkozóan.

Hozzáférési beállítások

A webszerver esetében fontos, hogy milyen módon lehessen elérni bizonyos fájlokat vagy könyvtárakat. Szükség lehet például olyanra, hogy egy adott könyvtár esetében, ahol nem létezik a „DirectoryIndex”-nek megfelelő fájl, ott ne listázza ki automatikusan a könyvtár tartalmát.

A korábban ismertetett blokk direktívák közül ily módon használhatjuk a `Directory`, `DirectoryMatch`, `Files`, `FilesMatch`, `Location`, `LocationMatch` direktívákat.

Ebben az esetben a következő fontosabb lehetőségeink vannak a „jogosultságok” korlátozására:

- `Options` Az alábbi opciók közül választhatunk, amelyek a direktíva után írhatók:
  - `None` Semmilyen opció nincs bekapcsolva
  - `All` Az összes opció be van kapcsolva
  - `ExecCGI` CGI végrehajtás engedélyezése
  - `FollowSymLinks` szimbolikus linkek követése (mutathat a `DocumentRoot`-on kívülre is)
  - `SymlinksIfOwnerMatch` szimbolikus linkek követése csak akkor, hogyha ugyanaz a link és a link céljának tulajdonosa
  - `Includes` SSI engedélyezése
  - `Indexes` könyvtár tartalom listázásának engedélyezése

  Az egyes opciók öröklődnek. Vagyis, ha adott szinten beállítjuk őket, akkor az az alkönyvtárakra is érvényes lesz. Amennyiben az `Options` után egyszerűen felsorolunk értékeket, úgy csak a felsorolt opciók lesznek bekapcsolva. Van arra is lehetőség, hogy a korábbi öröklődő beállításokhoz adjunk új opciót hozzá, vagy vegyünk el belőle. Ezt úgy tehetjük meg, hogy az adott opció neve elé + vagy – jelet írunk.

- `AllowOverride` Azt határozza meg, hogy a .htaccess fájlokban milyen globális beállításokat definiálhassunk felül. Az alábbi lehetőségek közül választhatunk, amelyek a direktíva után írhatók:
  - `None, All` semmit, mindent
  - `AuthConfig` hitelesítésre vonatkozó beállításokat
  - `FileInfo` dokumentum típussal kapcsolatos beállításokat
  - `Indexes` könyvtár listázásának formátumát
  - `Limit` hozzáférések engedélyezését és megtagadását
  - `Options` az előzőekben szereplő opciók közül melyeket

A hozzáférések engedélyezéséhez kapcsolódik az Order, Allow és Deny direktíva.

```
Order allow, deny
Order deny, allow
```

A hozzáférés vezérlő beállítások érvényességének sorrendje.

```
Allow from host_list
Deny from host_list
```

Az elérés engedélyezése, tiltása adott IP címekről vagy hosztokról

Egyszerű példa opciók és jogosultságok beállítására:

```
<Directory "/usr/local/apache2/manual">
    Options Indexes
    AllowOverride None
    Order allow,deny
    Allow from localhost
    Deny from localhost
</Directory>
```

Alias-ok, átirányítások

Arra is van lehetőségünk, hogy a DocumentRoot-on kívüli fájlokat és könyvtárakat elérhetővé tegyünk. Erre szolgálnak az ún. Alias-ok.

```
Alias /icons /usr/share/apache2/icons
```

Ebben az esetben például a `http:/www.apache_host.hu/icons/test.png` hivatkozás estében a `test.png` fájlt a `/usr/share/apache2/icons` könyvtárból próbálja meg elérni.

```
AliasMatch ^/icons(.*) /usr/share/apache2/icons$1
```

Annyiban különbözik az Alias-tól, hogy ebben az esetben reguláris kifejezés segítségével adhatjuk meg, hogy milyen hivatkozások esetében akarjuk a fájlokat máshonnét venni.

```
Redirect /inf http://inf.duf.hu
```

Amennyiben az első paraméterként megadott hivatkozással kezdődik az URL-ben lévő PATH, akkor a kérést át kell irányítani a második paraméterként megadott helyre.

```
RedirectMatch (.*\.jpg)$ http://inf.duf.hu$1
```

A RedirectMatch esetében szabályos kifejezés segítségével adhatjuk meg, hogy milyen fájlok esetében történjen meg az átirányítás.

```
ScriptAlias /cgi-bin /usr/local/apache2/cgi-bin
```

Ugyanúgy viselkedik, mint az Alias, csak ebben az esetben az adott könyvtár CGI scripteket tartalmaz.

```
ScriptAliasMatch ^/cgi-bin(.*) /usr/local/apache2/cgi-bin$1
```

Reguláris kifejezés alapján is megtehetjük a CGI scriptek helyének a meghatározását.

CGI szkriptek végrehajtásához szükséges beállítások:

Ahhoz, hogy adott parancsfájlokat CGI-ként tudjunk végrehajtani több beállítás is szükséges. Erre mutat példát a következő konfigurációs fájl részlet:

```
AddHandler cgi-script .cgi .pl
ScriptAlias /cgi-bin /usr/local/apache2/cgi-bin
<Directory "/usr/local/apache2/cgi-bin">
        Options +ExecCGI
        Order allow,deny
        Allow from all
</Directory>
```

Az AddHandler utasítás segítségével a .cgi-re és a .pl-re végződő fájlok esetében beállítjuk, hogy CGI-ként legyenek kezelve. A ScriptAlias segítségével megadjuk, hogy hol keresse a scripteket. Nem célszerű a CGI-ket a DocumentRoot-on belülre elhelyezni. Azon könyvtár esetében, ahonnét szeretnénk CGI-ket futtatni, szükséges az ExecCGI opció engedélyezése.

Egy fájl, amit CGI-ként szeretnénk futtatni fontos, hogy rendelkezzen végrehajtási jogosultsággal, és megfelelő HTTP fejlécet generáljon. A webszerver

csakis ebben az esetben tudja lefuttatni, és elküldeni a böngészőnek a parancs kimenetét.

A PHP szkriptek futtatásához szükséges beállítások:
Manapság a dinamikus web tartalom kiszolgálásához gyakori a PHP használata. Amennyiben a PHP-t dinamikus modulként szeretnénk használni, abban az esetben telepítenünk kell vagy csomagból (libapache2-mod-php5), vagy forráskódból. Miután a telepítést elvégeztük, be kell állítanunk, hogy az Apache bizonyos fájlok esetében a PHP értelmezőt felhasználva adja vissza a kimenetet.

```
LoadModule php5_module modules/libphp5.so
AddType application/x-httpd-php .php .php5
AddType application/x-http-php-source .phps
```

A php5 modul betöltése mellett társítanunk kell az értelmezőhöz kiterjesztéseket. Ebben a példában azt láthatjuk, hogyha az Apache olyan hivatkozást kap, amely .php-re vagy .php5-re végződik, akkor nem egyszerűen visszaküldi a fájlt, hanem átadja a beépülő PHP modulnak, amely a benne lévő kódot lefuttatja és ennek a kimenetét fogja a Webszerver visszaadni.
A beépülő modul mellett egy PHP nyelven megírt script is futtatható CGI-ként.

Az Apache 2.2 fontosabb moduljai:

Az webszerver működéséhez kapcsolódó részfeladatokat modulok végzik el. Vagyis ezek együttes működése révén valósul meg a web kiszolgálás. Mindegyik konfigurációs direktívához tartozik egy modul, amelyben annak a direktívának az értelmezése történik. Amennyiben konfigurációs hibával találkozunk, akkor a következőket célszerű ellenőrizni:

- a direktíva nevét és/vagy paraméterét megfelelően adtuk-e meg
- a direktíva működéséhez szükséges modul be lett-e töltve
- a megfelelő helyen (kontextusban) akarjuk-e használni a direktívát

Néhány modul, a hozzá tartozó funkcióval:

- `mod_alias.so` Alias-ok és átirányítások kezelése
- `mod_auth[n]_*.so` hitelesítéshez kötődő modulok
- `mod_authz_*.so` „Feljogosításhoz” kötődő modulok
- `mod_autoindex.so` Könyvtár tartalmának listázása
- `mod_cache.so` Cache az URL alapján indexelve
- `mod_cgi.so` CGI végrehajtás
- `mod_dav.so` WebDAV protokoll kezelése
- `mod_headers.so` HTTP kérés és válasz fejlécek
- `mod_include.so` SSI kezelése
- `mod_ldap.so` LDAP kapcsolat
- `mod_mime.so` MIME típusok kezelése
- `mod_proxy.so` HTTP 1.1 proxy/átjáró
- `mod_rewrite.so` URL manipuláció
- `mod_so.so` Dinamikus modul betöltés
- `mod_ssl.so` HTTPS támogatás
- `mod_status.so` A szerver állapot figyelés
- `mod_suexec.so` „Setuid wrapper” a CGI scriptekhez

- `mod_userdir.so` Felhasználói saját oldalak
- `mod_vhost_alias.so` Virtual hostok kezelése

Az itt felsorolt modulok mellett számos extra modul is elérhető külső forrásból (third party modules). Ezekkel a webszerver extra működésre képes, más programozási nyelven írt kódokat is értelmezni képes. pl.: mod_python.so a Python scriptek értelmezéséhez, mod_mono.so ASP.NET oldalak, mod_security.so az Apache biztonságának megerősítése, mod_jk.so az Tomcat Java Servlet Container-rel való kapcsolat.

Az Apache indítása és leállítása (apachectl):

Az apachectl parancs fontosabb kapcsolói és paraméterei:
- `-f` a szerver konfigurációs fájl megadása
- `-t` a konfiguráció tesztelése
- `-k`
  - `start` indítás
  - `stop` leállítás
  - `restart` újraindítás
  - `graceful` újraindítása a meglévő kapcsolatok bezárás nélkül
  - `status`, `fullstatus` a webszerver állapotának lekérdezése
- `-D string` direktíva létrehozása, aminek a létezése a későbbiekben az IfDefine segítségével megvizsgálható

Példák apachectl használatára:
```
apachectl start
        #ha nincs egyéb opció, akkor a -k elhagyható
apachectl -DSSL start
        # indítás SSL támogatással
apachectl -k graceful
        # újraindítása konfigurációs fájlok újra olvasásával (reload)
```

Virtualhost-ok használata

Az Apache támogatja a név alapú és az IP alapú virtualhost-ok használatát. Az első esetben a gép egy IP címmel rendelkezik és a HTTP fejléc Host mezője alapján történik a kérés intézése a megfelelő virtuális géphez. Az IP alapú virtualhost-ok esetében az egyes virtuális gépek különböző IP címet kapnak.

Példa virtuális hosztok használatára külön naplózással:
```
NameVirtualHost 192.168.30.2:80
<VirtualHost tom.a106.duf.hu>
        DocumentRoot "/var/www/tom/htdocs"
        ServerName tom.a106.duf.hu
        ServerAdmin webmaster@a106.duf.hu
        ErrorLog /var/www/tom/logs/error_log
        TransferLog /var/www/tom/logs/access_log
        ScriptAlias /cgi-bin/ /var/www/tom/cgi-bin/
</VirtualHost>
```

```
<VirtualHost jerry.a105.duf.hu>
        DocumentRoot "/var/www/jerry/htdocs"
        ServerName  jerry.a105.duf.hu
        ServerAdmin webmaster@a105.duf.hu
        ErrorLog /var/www/jerry/logs/error_log
        TransferLog /var/www/jerry/logs/access_log
</VirtualHost>
```

10.2.5. A Debian csomagban elérhető Apache 2.2 telepítése

Amennyiben nem szeretnénk forráskódból fordítani, akkor bináris csomagokból is telepíthetjük az Apache 2.2-es verzióját. Különböző MPM-ek támogatásával fordított csomagok érhetők el. Például:

- apache2-mpm-prefork hagyományos process alapú
- apache2-mpm-worker szál alapú modell

Az apache2.2-common csomagban találhatók azok a részei a webszervernek, amelyek mindegyik MPM esetében megegyeznek. A csomagok telepítésekor a adminisztrációs parancsok a /usr/sbin-be, a webszerver moduljai pedig a /usr/lib/apache2/modules könyvtárba kerülnek. A /etc/init.d/apache2 szkript, vagy az apache2ctl parancs segítségével megoldható a szolgáltatás indítása és leállítása.

Az Apache 2.2-höz tartozó konfigurációs fájlokat tartalmazó könyvtár a /etc/apache2. Itt a telepítést követően az alábbiakat találjuk:

- `conf.d` konfigurációs könyvtár, elsősorban Apache-ot használó alkalmazások konfigurációi számára
- `mods-available` az elérhető modulokhoz .conf és .load fájl, ezekkel lehetséges adott modult betölteni, és a hozzá kapcsolódó konfigurációt megadni
- `mods-enabled` szimbolikus linkeket tartalmaz, amelyek a `mods-available` könyvtárban lévő fájlokra mutatnak. Az itt megadott modulok kerülnek betöltésre a web szerver indításakor.
- `sites-available` az elérhető virtuális hosztok konfigurációs fájljai
- `sites-enabled` szimbolikus linkeket tartalmaz, amelyek a `sites-available` könyvtárban lévő fájlokra mutatnak. Az itt megadott virtuális hosztok lesznek engedélyezve a web szerver indításakor.
- `apache2.conf` A fő konfigurációs fájl, néhány általános beállítást tartalmaz, és ennek a segítségével történik a korábbiakban ismertetett könyvtárak „include”-ja.
- `ports.conf` Az Apache által használt portok beállítása

A Debian fejlesztők az előzőekben felvázolt struktúra kezelésének egyszerűsítésére fejlesztették ki az alábbi parancsokat:

- `a2ensite` virtuális hoszt engedélyezése
- `a2dissite` virtuális hoszt tiltása
- `a2enmod` modul betöltése
- `a2dismod` modul tiltása

A parancsok esetében egy site vagy modul nevet adhatunk meg paraméterként. Amennyiben paraméterek nélkül használjuk a parancsokat, akkor megmutatja a

rendelkezésre álló modulok vagy site-ok listáját, és a listát megtekintve beírhatjuk, hogy melyik esetében akarjuk a kért műveletet elvégezni. A parancsok kiadását követően újra kell tölteni az Apache konfigurációs fájljai és moduljait (/etc/init.d/apache2 force-reload).
Debian csomagban jónéhány third-party Apache modul is elérhető. Ezek a `libapache2-mod`-al kezdődő csomagokban vannak. Amennyiben PHP5-t is szeretnénk használni, akkor telepítenünk kell a `libapache2-mod-php5` csomagot, és a használni kívánt PHP bővítményeket (pl.: `php5-gd`, `php5-mysql`, ...). A csomagok úgy lettek elkészítve, hogy a telepítést követően használhatók az alapértelmezett beállításoknak megfelelően.

10.3. FTP szerver használata Debian GNU/Linux operációs rendszerben

10.3.1. Az FTP protokoll fontosabb jellemzői

A HTTP mellett elterjedt alkalmazás szintű protokoll az FTP (File Transfer Protokoll). Egy „nagyon régi" protokollról van szó, amelyet fájlok átvitelére találtak ki. A protokoll esetében fontos megemlíteni azt a biztonsági problémát, hogy a felhasználók jelszavai titkosítatlanul utaznak a hálózaton. Vagyis, hogyha valaki lehallgatja a hálózati forgalmat, és elkapja az egyes csomagokat, abban az esetben mindenfajta feltörés nélkül hozzájuthat adott felhasználó jelszavához. Amiért mégis elterjedt, azt valószínűleg a gyors adatátviteli lehetőségnek köszönheti. Nagyon sok alkalmazás létezik, amely képes FTP kliensként működni.

Azt FTP összetett protokoll. A működéséhez kettő TCP adatfolyamot használ. Ezek a portok tradicionálisan:
- 21-es TCP port: ftp command (ftp parancs csatorna)
- 20-as TCP port: ftp data (ftp adat csatorna)

A parancs csatornán keresztül történik a felhasználók hitelesítése és egyes parancsok küldése. Egy másik TCP kapcsolat pedig az adatok továbbítására való. Amellett, hogy az FTP kettő adatfolyamot használ, kétfajta működési módja is létezik, az aktív illetve a passzív mód.
Az aktív mód működése:
- A kliens kapcsolódik a szerver 21-es TCP portjához egy N > 1024 szabad TCP port felhasználásával
- A kliens megnyitja (listening) az N+1-es szabad TCP portot , és a PORT N+1 üzenetet elküldi a szervernek. Innét tudja a szerver, hogy neki melyik porthoz kell majd kapcsolódnia.
- A szerver ezután a 20-as TCP portjáról kapcsolódik a kliens által megnyitott portra

Ebben az esetben tehát egy a szerver felé irányuló, illetve egy a kliens felé irányuló TCP kapcsolat épül fel.

Passzív mód esetében a működés a következő:
- A kliens kapcsolódik a szerver 21-es TCP portjához egy N > 1024 szabad TCP port felhasználásával
- A kliens a PORT parancs helyett a PASV parancsot használja, amire a szerver megnyit egy random portot (P >1024), és ezt visszaküldi a kliensnek (PORT P)

- A kliens ezután N+1 -es szabad TCP porton keresztül kezdeményezi az adatkapcsolatot a szerver megnyitott távoli P TCP portjával

Ebben az esetben tehát kettő, a szerver felé irányuló TCP kapcsolat épül fel.

Amennyiben a kliens oldalt tűzfal védi, akkor az valószínűleg nem engedélyezi a bejövő TCP kapcsolatokat. Ebben az esetben az aktív mód nem fog működni, csak a passzív mód. Amikor azonban egy belső hálózatban vagyunk az FTP szerverrel, abban az esetben használhatjuk az aktív módot.

A tűzfalak esetében külön problémát jelent, hogy képesek-e felismerni, hogy FTP protokollról van szó, és ennek megfelelően eldönteni, hogy a kapcsolat jöhet-e befelé. A Linux kernel szintű tűzfala esetében létezik a Connection tracking támogatás FTP protokollra is. Ezzel megvizsgálható, hogy az adatkapcsolat valóban egy létező FTP session-höz tartozik-e.

Az FTP esetében létezik ún. Anonymous FTP, amely esetében bárki névtelenül tölthet le fájlokat a szerverről, illetve a felhasználónévhez és jelszóhoz kötött. Az Anonymous FTP esetében általában nincsen feltöltési lehetőség, míg a másik esetében általában van feltöltési lehetőség is. Az adott szerver konfigurálásakor beállítható, hogy milyen lehetőségei legyenek adott felhasználóknak.

A Debian GNU/Linux több olyan csomagot is tartalmaz, amelyek segítségével az FTP szerver funkcióval megvalósítható. Ilyenek például az ftpd, proftpd, pure-ftpd, vsftpd, wu-ftpd csomagok. Választásom azért eset legfőképp a ProFTPD-re, mert nagyon sok konfigurációs lehetősége van és konfigurálásában nagyon hasonlít az Apache-ra. A ProFTPD hivatalos weboldala a http://www.proftpd.org címen érhető el.

10.3.2. A ProFTPD (Professional File Transfer Protocol Daemon) jellemzői

- Sokoldalúan konfigurálható és biztonságos FTP kiszolgáló, jelentős felhasználói táborral.
- Egy fő konfigurációs fájl (/etc/proftpd/proftpd.conf) és egy modulok betöltéséhez (/etc/proftpd/proftpd.conf).
- A fő konfigurációs fájl mellett lehetséges helyi beállításokat megadni a .ftpaccess fájlok segítségével. (Apache esetében .htaccess volt.)
- Akár konfigurációs fájl nélkül is képes működni.
- Képes önállóan (standalone) módon és inetd/xinetd felhasználásával is működni.
- Standalone módban nem privilégizált felhasználóként fut.
- A ProFTPD nem hajt végre egyetlen külső alkalmazást sem.
- Megoldható a fájlok és könyvtárak rejtése felhasználónév vagy csoportnév vagy jogosultságok alapján.
- Moduláris felépítésű, könnyen kiterjeszthető az alapértelmezett működése (MySQL, LDAP, SSL, ...).
- Naplózása kompatibilis a wu-ftpd-vel, rendelkezik utmp/wtmp támogatással.
- Több virtuális szerver és anonymous FTP használatának lehetősége.
- Képes a felhasználókat „chroot”-olni, ezáltal minden felhasználó csak a saját könyvtára tartalmát látja, nem tud a könyvtár hierarchiában felfelé

lépkedni.
- IPv6 támogatással is rendelkezik.
- Nyílt forrású (GNU GPL licence), nemcsak Linux alatt létezik működő verziója.

Az előző felsorolásból láthatjuk, hogy a ProFTPD sok képességgel rendelkezik. A következő alfejezetben a konfigurációs lehetőségek közül fogok néhányak kiemelni.

10.3.3. A ProFTPD néhány konfigurációs direktívája

Az Apache-hoz hasonlóan itt is léteznek különböző kontextusok. Egy direktíva érvényességét tehát meghatározza az, hogy milyen helyen lett kiadva. A ProFTPD esetében a lehetőségek:
- /etc/proftpd/proftpd.conf fő konfigurációs fájl
- .ftpaccess helyi beállítások

A fő konfigurációs fájlon belül használhatunk különböző blokk direktívákat:
- külön megadás nélkül a fő szerverre vonatkozó beállítások
- `<Global> ...</Global>` az összes virtuális hosztra vonatkozó beállítások. Ezek igény szerint helyileg felüldefiniálhatók.
- `<Virtualhost>...</Virtualhost>` adott virtuális hosztra vonatkozó beállítások
- `<Anonymous rootdir>...</Anonymous>` anonymous ftp elérésre vonatkozó beállítások a gyökérkönyvtár megadásával
- `<Directory dir>...</Directory>` adott könyvtárra vonatkozó beállítások
- `<Limit cmds>...</Limit>` egyes FTP parancsok (cmds) korlátozása

Az egyes blokk direktívák egymásba ágyazhatók. Az előző felsorolás a beágyazást is bemutatja. Vagyis egy adott blokk direktíva a nála korábban szereplőkön belül használható.

Néhány általános beállítás a FTP szerverre vonatkozóan:

```
ServerName "ProFTPD – Debian"
```
A szerver nevének a megadása.

```
ServerType standalone
```
A szerver működési módjának a megadása. Ebben a példában önállóan működik.

```
Bind 192.168.30.2
```
Az IP cím, amelyen keresztül el tudjuk érni a szervert.

```
Port 21
```
A TCP port, amelyet a szerver figyel.

```
PassivePorts 49152 65534
```
Passzív mód esetében melyik TCP portokat nyissa meg a szerver.

```
User proftpd
Group nogroup
```
A felhasználó, felhasználói csoport akinek a nevében a szerver végrehajtódik.

```
DefaultRoot ~ users
```
Az users nevű csoport tagjai csak a saját könyvtárukat érhetik el. Célszerű beállítani, hogy ne lehessen tetszőleges rendszerbeállítást elérni.

```
DefaultServer on
```
Amennyiben több virtuális szerverünk is van, megadhatjuk, hogy melyik legyen az alapértelmezett.

```
DisplayLogin welcome.msg
DisplayQuit quit.msg
DisplayFirstChdir .message
```
Üzenetek megjelenítése bejelentkezéskor, kilépéskor és az első könyvtár váltáskor.

```
ShowSymlinks on
```
Mutassa-e a szimbolikus linkeket, mint önálló fájltípust?

```
ListOptions "-lt"
```
A könyvtár listázások alkalmával használandó listázási opciók.

```
MaxLoginAttempts 3
```
A bejelentkezési kísérletek száma.

```
TimeoutNoTransfer 600
TimeoutStalled 600
TimeoutIdle 1200
```
Az időtúllépés beállítása, hogyha nincs átvitel, megállt az adatátvitel, nem történik semmi.

```
MaxInstances 30
MaxClients 100
MaxClientsPerHost 4
MaxClientsPerUser 3
```
A gyermekfolyamatok és a kliens maximális száma egyidejűleg. Lehetséges hosztonként és felhasználóként is korlátozni a kapcsolatok maximális számát.

```
AuthUserFile /home/ftpusers
AuthGroupFile /home/ftpgroups
```
A ProFTPD képes arra, hogy a felhasználók adatait ne a rendszer felhasználói adatbázisából kérdezze le, hanem saját fájlokat használjon erre a célra.

```
Include /etc/proftpd/ldap.conf
```
Külső saját konfigurációs fájl betöltése.

Az FTP-n keresztüli jogosultságok beállítása

Az FTP protokoll sok parancsot definiál. Ezek közül választhatunk, hogy melyeket szeretnénk engedélyezni, illetve melyeket nem.
Néhány FTP protokollon belüli utasítás a teljesség igénye nélkül:

- CWD (Change Working Directory) könyvtár váltás
- MKD / XMKD (MaKe Directory) könyvtár létrehozás

- RNFR (ReName FRom) átnevezni valamiről
- RNTO (ReName TO) átnevezni valamire
- DELE (DELEte) fájl törlése
- RMD / XRMD (ReMove Directory) könyvtár törlése
- RETR (RETRieve) letöltés
- STOR (STORe) feltöltés

Azért, hogy a jogosultság megadáskor ne kelljen többet ezek közül felsorolni adott esetben, ezért definiáltak jogosultsági szinteket. Ezekhez egyszerre több „elemi utasítás” tartozik.

- READ fájl olvasáshoz kötődő utasítások
- WRITE létrehozáshoz, törléshez, átnevezéshez kötődő utasítások
- DIRS könyvtár váltáshoz, listázáshoz kötődő utasítások
- LOGIN a bejelentkezéshez kötődő utasítások

A <Limit> blokkon belül adhatjuk meg az adott parancsokra vagy jogosultságokra vonatkozó „korlátozásokat”:

```
Order allow,deny
```
A jogosultsági megadások sorrendjének megadása. Először az engedélyezések, majd a tiltások.

```
AllowAll
DenyAll
```
Bárhonnét engedélyezett, tiltott az elérés.

```
Allow from 192.168.31.1
Deny from 192.168.31
```
Engedélyezés, tiltás adott IP címekről vagy tartományokból.

```
AllowUser [name], AllowGroup [name]
DenyUser [name], DenyGroup [name]
```
Engedélyezés vagy tiltás felhasználó vagy csoportnév alapján.

```
AllowOverride on
```
A beállítások helyi felüldefiniálásának engedélyezése.

```
AllowOverwrite on
```
A fájlok felülírásának engedélyezése.

```
AllowRetrieveRestart on
AllowStoreRestart on
```
Letöltés, feltöltés folytatásának engedélyezése.

```
HideUser [name], HideGroup [name]
HideNoAccess
HideFiles [regexp]
IgnoreHidden on
```
Fájlok és könyvtárak rejtése tulajdonos, tulajdonos csoport, jogosultság, vagy reguláris kifejezés alapján. Ahhoz, hogy a rejtés működjön, az IgnoreHidden

opciót be kell kapcsolni.

```
TransferLog /var/log/proftpd/xferlog
SystemLog   /var/log/proftpd/proftpd.log
```
A fájl átvitelek és az FTP szerver működésének naplózása.

A ProFTPD néhány modulja, a hozzá tartozó funkcióval:

- `mod_auth` Felhasználó hitelesítés
- `mod_cap` Modulok „képességeinek” (betöltés, kivétel, ...) megadása
- `mod_core` FTP Protokoll kezelése (RFC 959)
- `mod_ldap` LDAP alapú hitelesítés
- `mod_log` Naplózási beállítások
- `mod_ls` Listázás
- `mod_quotatab` Felhasználói kvóta
- `mod_radius` RADIUS (Remote Authentication Dial-In User Service) alapú hitelesítés
- `mod_ratio` Le és feltöltés aránya
- `mod_readme` Üzenetek megjelenítése
- `mod_rewrite` Kérések átszerkesztése menet közben
- `mod_sql` SQL adatbázis alapú hitelesítés és naplózás
- `mod_tls` TLS (Transport Layer Security) támogatása
- `mod_wrap` TCP wrapper használata
- `mod_xfer` Fájl átvitel

A ProFTPD-vel kapcsolatos parancsok:

Számos olyan parancs létezik, amely segítségével a ProFTPD működését vizsgálhatjuk és befolyásolhatjuk.

- `ftpdctl` A démon működését befolyásolhatjuk a segítségével. Például az egyes virtuális szerverekre vonatkozó állapot információk, szerver indítás/leállítás, adott kapcsolatok megszakítása, ...
- `ftpcount` Az ftp kapcsolatok számát mutatja meg.
- `ftpstat` Napokra bontva statisztika a fájl átvitelekről, ezeknek a sebességéről, a sikeres átvitel arányáról.
- `ftpasswd` A felhasználók hitelesítésére használható helyi fájlok létrehozása (`AuthUserFile, AuthGroupFile`)
- `ftpshut` Az ftp szerver időzített leállítása.
- `ftptop` Az ftp szerver monitorozása a top parancshoz hasonló felületen keresztül.
- `ftpwho` Megmutatja, hogy kik vannak aktuálisan bejelentkezve és mióta.

A ProFTPD képes a PAM használatára. Működésére tehát hatással van a /etc/pam.d/proftpd fájl tartalma. Az alapértelmezett hitelesítési módszerek mellett ez tartalmazza a következő sort a telepítést követően:

```
auth required pam_listfile.so item=user sense=deny \
file=/etc/ftpusers onerr=succeed
```

Ez a beállítás annyit jelent, hogy azon felhasználóktól, akik szerepelnek a `/etc/ftpusers` nevű fájlba, azoktól meg kell tagadni a bejelentkezést. Alapesetben a root-ként történő bejelentkezés pontosan emiatt kerül tiltásra.

Példa ftpasswd használatára:

```
echo 'jelszo' |ftpasswd --passwd --file=/home/ftpusers --name=xyz123
--stdin --uid=1500 --gid=200 --md5 --home=/home/xyz123 --shell=/bin/
false
```

Amennyiben egy felhasználó esetében az alapértelmezett shellt /bin/false-ra állítjuk, akkor ő normál módon nem tud bejelentkezni és parancsokat futtatni.

10.4. A Samba

10.4.1. A Samba legfontosabb jellemzői

A Samba projekt jelmondata: „Opening Windows to a Wider World”. Amennyiben rendelkezünk az „ablakozós” operációs rendszer valamelyik változatával, és szeretnénk elérni a Linux operációs rendszerben tárolt könyvtárainkat vagy fájljainkat, abban az esetben a Samba egy jó választás lehet. A http://www.samba.org a hivatalos web oldala.

A Samba legfontosabb jellemzői:

- A Samba szerver lehetővé teszi a fájl és nyomtató megosztást. Ebben az esetben a Samba lehet egy munkacsoport vagy tartomány tagja, vagy irányítója.
- A TCP/IP fölötti kommunikációt támogatja, mindenféle megosztás elérése csak ez alapján történhet.
- A Linux mellett más operációs rendszerekben (Unix, OpenVMS, Netware) is elérhető.
- Nyílt forrású, GNU GPL licenc alatt érhető el
- Samba kliens segítségével képesek vagyunk „ablakozós” operációs rendszerben megosztott mappák vagy nyomtatók elérésére illetve Samba szerver által megosztott erőforrások elérésére.
- Az SMB (Server Message Block) és a CIFS (Common Internet File System) protokollokon keresztül történhet a hálózati erőforrásokhoz való hozzáférés.
- A Linux kernel tartalmazza az Smbfs és CIFS támogatást. Ezek lehetővé teszik, hogy hálózaton keresztül csatlakoztassunk egy megosztott könyvtárat (smbmount).

A Samba tehát tartalmaz olyan eszközöket, amelyek segítségével a Linux-ot futtató számítógép akár kliensként, akár pedig kiszolgálóként is működhet azon az elven, amelyen az ablakozós operációs rendszerben a „Network Neighborhood (Hálózati helyek)” is eléri a hálózati erőforrásokat.

A Samba szerver legfontosabb jellemzői:

- fájl és nyomtató megosztás az SMB és/vagy CIFS felhasználásával
- lehetővé teszi, hogy „ablakozós” operációs rendszerből vagy bármilyen SMB vagy CIFS kompatibilis program segítségével elérjük a Linux alatt megosztott erőforrásainkat
- WINS (Windows Internetworking Name Server) szolgáltatás nyújtása
- Munkacsoportok vagy tartományok létrehozása és vezérlése
- Saját profile és logon script tárolása központi helyen
- Működhet központi hitelesítő szerverként is

A Samba szerver tehát több, egymástól függetleníthető funkciót is ellát. Mind az Apache, mind pedig a ProFTPD esetében egy fő démon folyamatról beszélhetünk, amelyek létrehoznak több szálat vagy gyermekfolyamatot. Ezzel szemben a Samba funkcióinak a megvalósítását több démon végzi.

- **nmbd**
  - névfeloldás
  - WINS, NetBIOS névkiszolgáló
  - hálózati erőforrások tallózása
- **smbd**
  - a megosztott erőforrások kezeléséért felelős
  - fájl és nyomtató megosztása
  - kommunikáció kezelése (SMB, CIFS)
  - hitelesítés, erőforrás zárolás (locking)
- **winbindd**
  - másik tartományban lévő felhasználók „leképezése” Linux UID-ra és GID-re

A démonok közül elsőként az nmbd-t majd pedig a smbd-t kell indítani. A winbindd-re csak speciális szituációban van szükség. Abban az esetben, hogyha csak WINS kiszolgálóként akarom használni a Samba-t, akkor csak az nmbd futtatására van szükségem.
A Debian 4.0 tartalmazza bináris csomagként a Samba 3.0-t. Amennyiben csomagból telepítjük (samba), akkor a /etc/init.d/samba script segítségével megoldhatjuk a Samba indítását,leállítását és újraindítását.

A Samba több TCP és UDP portot is használ.
Az smbd szolgáltatás a

- 139/tcp NetBIOS Session Service
- 445/tcp Microsoft Naked CIFS

portokat, a nmbd szolgáltatás pedig a

- 137/udp NetBIOS Name Service
- 138/udp NetBIOS Datagram Service

portokat használja.

10.4.2. A Samba néhány konfigurációs direktívája

A Samba esetében Debian GNU/Linux operációs rendszerben a fő konfigurációs fájl a /etc/samba/smb.conf. A Samba konfigurálása során nagyon sok direktíva közül választhatunk.
A konfigurációs fájl esetében szögletes zárójelek jelzik egy adott blokk elejét. A blokk hatóköre egészen addig tart, amíg egy következő blokk direktíva nem kezdődik.
A [global] direktíva után következő utasítások globális érvényűek lesznek. Itt olyan beállításokat adunk meg, amelyek a Samba szerver általános működését befolyásolják. Emellett van arra lehetőség, hogy saját blokkokat használjunk. Ebben az esetben a szögletes zárójelek között megadott név lesz automatikusan az adott megosztás neve, és utána állíthatjuk be a megosztáshoz tartozó paramétereket.

Egy minimális smb.conf fájl lehet a következő:

```
[global]
workgroup = A-106
[homes]
guest ok = no
read only = no
```

A [global] szekció esetében használható konfigurációs lehetőségek közül néhány:

`security =`

`user`	helyi felhasználónév és jelszó alapján
`share`	megosztásokhoz tartozó jelszóval, ami nem kötelező
`server`	másik Samba vagy NT szerver felhasználásával
`domain`	tartományvezérlő (domain controller) felhasználásával
`ads`	Active Directory használatával

A security értéke meghatározza, hogy a Samba milyen módon próbálja meg a bejövő kérések ellenőrzését és hitelesítését.

`server string = Linux_debian`
Egy leírás adható a szerver neve mellé.

```
interfaces = 172.31.0.1
bind interfaces only = yes
```
Amennyiben több hálózati eszközünk és IP címünk van, akkor megadható, hogy melyeken keresztül lehessen a Samba szervert elérni.

`workgroup = A-105`
A munkacsoport vagy tartomány neve

```
hosts allow = 172.31.
hosts deny = 192.168.31.
```
Honnét engedélyezett és honnét tiltott a Samba szerver elérése.

```
guest account = nobody
```
A Vendég felhasználóhoz milyen Linux felhasználó tartozzon jogosultság szempontjából.

```
max log size = 500
log file = /var/log/samba/log.%m
```
A maximális naplófájlok méretének és helyének megadása. A példában gépnevenként különböző fájlba kerülnek a naplóbejegyzések.

```
character set = utf-8
client code page = 852
```
A karakter készlet és a kliens oldali kódlap megadása.

```
domain logons = yes
logon home = \\%L\profiles\%U\
logon script = logon.bat
logon drive = H:
```
Tartományi bejelentkezés engedélyezése és a saját felhasználói profil, a bejelentkezéskor lefutó parancsfájl és a saját meghajtó megadása.

```
encrypt passwords = true
```
Titkosított jelszavak használata.

```
unix password sync = false
```
A Samba jelszavak módosításakor a rendszerbeli jelszót is szükséges-e módosítani?

```
os level = 65
```
Az operációs rendszer „fejlettsége” a „master browser” funkció elnyeréséhez. Amennyiben kellően nagyra állítjuk, úgy biztosan a Samba szerver látja el a különböző vezérlő funkciókat.

```
domain master = yes
```
A Samba szerver legyen a tartomány esetében a master browser.

```
local master = yes
```
A Samba szerver legyen a adott alhálózat esetében a master browser.

```
preferred master = yes
```
Amikor a Samba szerver indul, akkor automatikusan magához ragadja-e a master browser funkciót.

Egy megosztás esetében használható direktívák:

A globális direktívák mellett fontos dolog, hogy milyen lehetőségeink vannak a saját megosztások esetében. A [share1] direktíva annyit jelent, hogy az utána következő utasítások a share1 nevű megosztásra vonatkoznak.

```
path = /home/pub
comment=Publikus adatok
```
A megosztáshoz tartozó elérési útvonal és megjegyzés

`browseable = yes`
A megosztás megjelenjen-e a tallózási listában.

`read only = yes`
`writeable = no`
Megosztás csak olvasásra.

`printable = yes`
Nyomtató megosztásának definiálása.

`public = no`
`guest ok = no`
A megosztás nyilvános legyen-e, tehát vendég felhasználóval is hozzá lehessen-e férni?

`create mask =0700`
Fájlok esetében az alapértelmezett jogosultság amivel a fájl létrejön.

`directory mask = 0770`
Könyvtárak esetében az alapértelmezett jogosultság amivel a fájl létrejön.

`valid users=tux beastie`
`invalid users = root bin daemon adm sync shutdown`
Mely felhasználók számára engedélyezett/tiltott a megosztás elérése. Használhatók a direktívák a [global] szekcióban is.

`read list= beastie`
`write list= tux @users`
Azon felhasználók listája, akik csak olvashatják vagy írhatják és olvashatják az adott megosztás tartalmát. A @ jel után felhasználói csoportokat adhatunk meg.

`admin users = root Administrator`
Az adott megosztás esetében minden jogosultsága megvan.

`veto files = /.htaccess/.tmp*/`
Fájlok listázásának tiltása.

`hide files= /.mc/`
Fájlok „felruházása” a hidden DOS fájl attribútummal.

`hide unreadable`
Azon fájlok „felruházása” a hidden DOS fájl attribútummal, amelyekre vonatkozóan nincsen olvasási jogosultság.

Példák megosztásokra:

```
[homes]
   comment = Home Directories
   browseable = no
   writable = yes
   valid users = %S
```

Minden felhasználó a saját könyvtárát elérheti.

```
[printers]
comment = All Printers
path = /var/spool/samba
printer admin = pr_admin
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = No
```

A Samba esetében fontos megemlíteni, hogy amikor helyi hitelesítésről beszélünk, akkor nem a /etc/passwd, /etc/shadow alapján azonosít. A korábbi verziók esetében a /etc/samba/smbpasswd fájlban tárolódtak a felhasználók jellemzői. A 3.0-es verziótól kezdve már TDB típusú adatbázisban történik a tárolás. A /var/lib/samba/passdb.tdb fájlban tárolódnak a felhasználók adatai, illetve ugyanebben a könyvtárban találhatók egyéb konfigurációs adatbázisok is, amelyek például a megosztott nyomtatók adatait, vagy az account-okra vonatkozó egyéb adatokat tartalmazzák.

A Samba esetében a következő fontos jellemzők tárolódnak a felhasználók esetében:

- username
- UID
- Lanman Password Hash (jelszó tárolása a régebbi rendszerek számára)
- NT Password Hash (jelszó tárolása az NT és újabb rendszerek számára)
- Account flags (a jelszóval, azonosítóval kapcsolatos beállítások)

A Samba szerver működésével kapcsolatos parancsok:

- `net` Samba és távoli CIFS szerverek adminisztrációja
- `pdbedit` Samba felhasználók felvétele, jellemzőik módosítása
- `smbcontrol` Vezérlő üzenet küldése a démonoknak
- `smbpasswd` Samba felhasználók jelszavának módosítása
- `smbstatus` Samba állapotának megtekintése
- `testparm` smb.conf ellenőrzése

A Samba kliens oldalhoz kapcsolódó parancsok:

- `findsmb` Távoli SMB-t használó gép keresése
- `nmblookup` NetBIOS nevek feloldása
- `rpcclient` MS-RPC hívások végrehajtása
- `smbcacls` NT ACL-ek manipulációja SMB megosztásokon
- `smbclient` FTP kliensre hasonlító SAMBA kliens
- `smbget` `wget`-szerű program SAMBA-n keresztüli letöltésre
- `smbspool` Fájl küldése SAMBA nyomtatónak
- `smbtar` SAMBA megosztások mentése szalagra
- `smbtree` Karakteres felület SMB hálózat tallózó

Az smbmount (mount.smb) parancs segítségével megtehetjük egy SMB-n keresztül megosztott könyvtár csatlakoztatását.
Az smbmount parancs fontosabb opciói:

```
smbmount service mount-point [ -o options ]
```

Ahol:

- `service`: egy hálózati erőforrás megadása az UNC szintaxishoz hasonlóan. Ebben az esetben a `//host/share` jelölést használhatjuk, ahol a `host` jelenti a Samba szerver nevét vagy IP címét, a `share` pedig a megosztás nevét.
- `mount-point`: csatlakoztatási pont

Az opciók esetében a fontosabb lehetőségek:

- `username=`: a felhasználó, akivel csatlakozni kívánunk
- `password=`: a kapcsolódáskor használandó jelszó
- `guest`: vendégként való csatlakozás
- `uid=`: a helyi UID, akinek a „tulajdonába kerül” a csatlakoztatott könyvtár
- `gid=`: a helyi GID, akinek a „tulajdonába kerül” a csatlakoztatott könyvtár
- `workgroup=`: a munkacsoport vagy tartomány neve
- `ip=`: a szerver IP címe
- `ro`: csak olvasható módban
- `rw`: írható és olvasható módban
- `iocharset=`: adott karakterkészlet használata
- `codepage=`: adott kódlap használata

Az smbumount parancs segítségével leválaszthatunk egy csatlakoztatott SMB megosztást.

Példa smbmount használatára:

```
smbmount //srv/apps /mnt/prgs -o options username=tux,password=titok
```

A Debian tartalmazza a swat (Samba Web Administration Tools) nevű csomagot, amely segítségével böngészőn keresztül lesz lehetőségünk a Samba konfigurálására.

10.5. A MySQL

Manapság nagyon gyakran használnak Linuxot web-es feladatok ellátására. Ennek kapcsán gyakran LAMP-ként emlegetik a fejlesztői környezetet, ahol az egyes betűk jelentése: **L**inux **A**pache **M**ysql **P**HP. Az utóbbi években jelentős fejlődésnek lehettünk tanúi. A MySQL 5.0-as verziója már tartalmazza a legtöbb olyan képességet, amellyel az Oracle vagy a DB2 rendelkezik. A MySQL Enterprise Server kereskedelmi termék, ám a Community Server nyílt forrású. Manapság a legtöbb disztribúció tartalmazza csomag formájában. A MySQL-t nemrég vásárolta meg a Sun.

10.5.1. A MySQL szerver legfontosabb jellemzői

- több felhasználós, több szálú adatbázis szerver
- SQL alapú, az ANSI/ISO SQL nyelvet használja saját bővítésekkel
- sebességre optimalizált
- nagy felhasználói tábor
- a 4.1-es verziótól felfelé van beágyazott SELECT
- az 5.0-as verzióban megjelentek a nézetek (VIEW) és a triggerek és a tárolt eljárások támogatása
- különféle Storage Engine -eket támogat (MyISAM, InnoDB, ...)
- képes együttműködni a Linux HA (High Availability) alkalmazásaival
- megoldható több szerver szinkron vagy szinkron replikációja
- képesek vagyunk MySQL klaszterek építésére
- több programozási nyelvből elérhető különféle API-k, csatolók és program-könyvtárak (library) felhasználásával
- különböző nyelvek és karakterkészletek támogatása

A MySQL 5.0 elérhető csomagból is, de a gyártó különböző platformokra biztosít belőle előre fordított binárisokat. Amennyiben 32 bites rendszerünk van, úgy pl.: a mysql-5.0.51a-linux-i686-glibc23.tar.gz számunkra megfelelő lesz. A gcc mellett elérhető az Intel C fordítójával előállított változat is. A MySQL hivatalos weboldala a http://www.mysql.com, a Community Server a http://dev.mysql.com-ról tölthető le.

10.5.2. A MySQL szerver telepítése bináris archívumból

Ha szeretnénk tudni a telepítési lépéseket, legegyszerűbb, hogyha belenézünk az tömörített fájlba, és áttekintjük az ott lévő INSTALL-BINARY fájl tartalmát. Ebben megtalálhatjuk azokat a parancsokat, amelyek segítségével telepíthetjük a MySQL adatbázis szervert.
Tételezzük fel, hogy letöltöttünk a /root könyvtárba a mysql-5.0.51a-linux-i686-glibc23.tar.gz-t.
Ebben az esetben a telepítés egyes lépései:

- `groupadd mysql` # mysql csoport létrehozása
- `useradd -g mysql mysql` # mysql felhasználó létrehozása
- `cd /usr/local`
- `tar xvzf /root/mysql-5.0.51a-linux-i686-glibc23.tar.gz` # az archívum kicsomagolása
- `ln -s mysql-5.0.51a-linux-i686-glibc23 mysql` # szimbolikus link
- `cd mysql`
- `chown -R mysql .` # a tulajdonos a 'mysql' nevű felhasználó
- `chgrp -R mysql .` # tulajdonos csoport beállítása
- `scripts/mysql_install_db –user=mysql` # a kiindulási adatbázis létrehozása
- `chown -R root .` # a tulajdonos a 'root' nevű felhasználó
- `chown -R mysql data` # a 'data' könyvtár tulajdonosa a 'mysql' nevű felhasználó
- `bin/mysqld_safe --user=mysql &` # az adatbázis szerver indítása

Amennyiben már egy korábbi verziót telepítettünk a gépre, úgy a telepítése egyes lépéseit feleslegesek. Ebben az esetben csak a kicsomagolás és a

jogosultságok beállítása szükséges. A korábbi verzióból (feltéve, hogy az is 5.0-s ághoz tartozik) át lehet másolni az adatbázisokat

10.5.3. A MySQL könyvtárai és parancsai

A telepítést követően a következő könyvtárak és fájlok jönnek létre a /usr/local/mysql könyvtárban (a fájlok esetében csak a fontosabbakat sorolom fel):

- bin — a MySQL-hez tartozó binárisok
  - mysql — MySQL monitor (parancs soros kliens)
  - mysqladmin — az adatbázis szerver adminisztrálása
  - mysql_client_test — a kliens működésének tesztelése
  - mysql_config — a fordítás opciók lekérdezése
  - mysqld — a MySQL démon
  - mysqldump — mentés készítése adatbázisról vagy tábláról
  - mysqld_multi — több MySQL szerver egyidejű kezelése
  - mysqld_safe — az adatbázis szerver indítása
  - mysqlimport — adatok importálása adattáblába
  - mysqlshow — adattáblák felépítésének megmutatása
  - mysql_upgrade — adattáblák ellenőrzése frissítéskor
- data — adatbázisok: minden adatbázis külön könyvtárban az adatbázis nevének megfelelően
- docs
  - mysql.info — dokumentáció
- include — header fájlok
- lib — program-könyvtárak
- man — felhasználói kézikönyv oldalak
- mysql-test — adatbázis működési tesztje
- scripts
  - mysql_install_db — a kiindulási adatbázis létrehozása
- share — nyelvi fájlok, karakter készletek
- sql-bench sebesség tesztelése
- support-files — példa konfigurációs fájlok, scriptek
  - mysql.server — init.d script a szerver indítására, leállítására
- tests — Perl nyelven készített tesztelő scriptek

10.5.4. A MySQL néhány konfigurációs direktívája

A MySQL konfigurálása az alábbi fájlok segítségével lehetséges:

- /etc/my.cnf — globális beállítások
- $MYSQL_HOME/my.cnf — a szerverre vonatkozó beállítások (MYSQL_HOME vagy a telepítési könyvtár elérési útvonalát, vagy pedig az adatbázisokat tartalmazó könyvtár elérési útvonalát tartalmazza)
- --defaults-extra-file=”file” — a szerver paramétereként átadható tetszőleges konfigurációs fájl
- ~/.my.cnf — felhasználóként a saját beállításainkat megadhatjuk

A konfigurációs fájl jellege hasonlít a Samba esetében használt módszerhez. Itt is szekciók vannak, amelyek elejét szögletes zárójelekkel jelöljük
A konfiguráció fájlban lehetnek:

- #megjegyzések

- [group] #adott program vagy csoportnév amelyre az utána lévő beállítások vonatkoznak
- option # opció paraméterek nélkül
- option=value # opció paraméterrel
- set-variable=variable=value # szerver által használt „belső változók”

Példa MySQL konfigurációjára:

```
[client]
#password = my_password
port = 3306
socket = /var/run/mysqld/mysqld.sock

[mysqld]
user = mysql
basedir = /usr/local/mysql
datadir = /usr/local/mysql/data
language = /usr/local/mysql/mysql/share/hungarian
pid-file = /var/run/mysqld/mysqld.pid
socket = /var/run/mysqld/mysqld.sock
port = 3306
log = /var/log/mysql.log
tmpdir = /usr/local/mysql/tmp
default-character-set=utf8
```

Az előző példában szereplő opciók jelentése:

- `user` # a felhasználó, akinek a nevében kell futtatni a szervert
- `passwd` # a kliens kapcsolódáshoz használt jelszó
- `basedir` # a telepítési könyvtár
- `datadir` # az adatbázisokat tartalmazó könyvtár
- `port` # a használt TCP port, alapértelmezés a 3306
- `socket` # az 's' típusú speciális fájl, amin keresztül a kommunikáció történik
- `language` # a hibaüzenetek nyelve
- `default-character-set` # alapértelmezett karakter készlet
- `tmpdir` # könyvtár az ideiglenes fájlok számára
- `log` # a naplófájl helye
- `pid-file` # a démon PID-je ebben a fájlban lesz tárolva

A MySQL néhány további beállítása:

`bind-address=127.0.0.1`
Melyik IP címen keresztül legyen elérhető

`skip-networking`
Csak helyileg lesz elérhető a szerver egy Unix Domain Socket-en keresztül.

`max_connections=200`
A maximális egyidejű kapcsolatok száma

Amennyiben nem szükséges, hogy az adatbázis szerver hálózaton keresztül is elérhető legyen, úgy célszerű beállítani, hogy csak localhost-on hallgatózzon. Ekkor egy külső támadó közvetlenül nem tudja támadni az adatbázis szervert. A kapcsolatok számának korlátozása pedig megakadályozza, hogy a szervert túlzott terhelését.

10.5.5. A MySQL jogosultsági rendszere

Ahhoz, hogy adatbázis szervert tudjuk adminisztrálni, ismernünk kell, hogy az milyen módon azonosítja a felhasználókat. A MySQL esetében a felhasználói adatai a „mysql” nevű adatbázisban tárolódnak.

A MySQL szerver két lépésben ellenőriz egy adott klienst:

- Elsőként ellenőrzi, hogy adott helyről (IP címről) megengedett-e a kliens kapcsolódása.
- Amennyiben ez lehetséges, akkor a kapcsolatot követően minden parancs végrehajtásakor megvizsgálja, hogy az adott kliens rendlkezik-e a megfelelő hozzáféréssel.

Jogosultság szempontjából a következő adattáblákat használja a MySQL:

- `user` Ebben vannak tárolva a globális jogosultságok. Melyik host-ról jelentkezhet be adott felhasználó, a megfelelő jelszóval és mik az ő jogosultságai.
- `db`: Ebben van tárolva, hogy melyik host-ról érhet el egy adott felhasználó egy adatbázist és milyen módon.
- `host`: Ebben van tárolva, hogy melyik host-ról érhető el egy adatbázis és milyen módon.

Ezen három tábla alapján meghatározható, hogy egy felhasználónak van-e arra lehetősége, hogy kapcsolódjon. Illetve, hogyha tudott kapcsolódni, akkor adatbázis szinten milyen jogosultságai vannak.

- `tables_priv` Ebben vannak tárolva, hogy adott helyről adott felhasználónak milyen jogosultságai vannak egy bizonyos adattáblára vonatkozóan.
- `columns_priv` A tábla jogosultságai mellett itt még adott oszlophoz tartozó jogosultságok is tárolva vannak.
- `procs_priv` A szerver oldali tárolt eljárásokra és függvényekre vonatkozó jogosultságok.

A jogosultságok meghatározása az alábbi képlet alapján történhet:

Global level (`user`)
OR Database level (`db` **AND** `host`)
OR Table level (`tables_priv`)
OR Column level (`columns_priv`)
OR Routine level (`procs_priv`)

A MySQL által használt jogosultságok (privilégiumok):

A jogosultságokat csoportosíthatjuk olyan szempontból, hogy melyik szinten

adhatjuk meg. Az előző képletből megnézhetjük a különböző jogosultsági szinteket.

Adatbázis szinten a következő jogosultságok adhatók meg:

- SELECT jog SELECT parancs végrehajtására
- INSERT jog INSERT parancs végrehajtására
- UPDATE jog UPDATE parancs végrehajtására
- DELETE jog DELETE parancs végrehajtására
- CREATE jog adattábla létrehozásra (CREATE TABLE)
- DROP jog adattábla törlésére (DROP TABLE)
- GRANT jog jogosultságok adására
- REFERENCES jog hivatkozások használatára
- INDEX jog index létrehozására és törlésére
- ALTER jog adattábla szerkezetének változtatására (ALTER TABLE)
- CREATE TEMPORARY TABLES jog ideiglenes adattábla létrehozására
- LOCK TABLES jog adattábla zárolására
- CREATE VIEW jog nézetek létrehozására
- SHOW VIEW jog nézetek megtekintésére
- CREATE ROUTINE jog eljárások és függvények használatára
- ALTER ROUTINE jog eljárások és függvények változtatására
- EXECUTE jog eljárások és függvények végrehajtására
- TRIGGER jog trigger létrehozásra és törlésre

Az adatbázis szintű jogosultságok magukba foglalják a tábla, oszlop és rutin szintű jogosultságokat.

Globális szinten az adatbázis szintű jogosultságok mellett még a következők is elérhetők:

- RELOAD jog a szerver beállítások újraolvasásához
- SHUTDOWN jog a szerver leállításához
- PROCESS jog a szerver folyamatok megtekintésére
- FILE jog adatok fájlba mentésére (SELECT INTO) és betöltésére (LOAD DATA)
- SHOW DATABASES jog az adatbázisok megtekintésére
- SUPER jog a szerver megfigyeléséhez, kliensek kapcsolatok megszakításához, a szerver működésének megváltoztatásához
- REPLICATION CLIENT jog a master szerver eléréséhez (replikáció esetében), a frissítéshez feltétlenül szükséges
- REPLICATION SLAVE jog a master vagy a slave állapotának lekérdezéséhez
- CREATE USER felhasználó létrehozása és jellemzőinek módosítása

A rendszergazdai jelszó megadása:

Miután egy adatbázis szervert telepítettünk, elsőként a rendszergazdai jelszót kell beállítanunk. A jelszó beállítását többféle módon is megtehetjük:

A `SET PASSWORD FOR` utasítással:

```
mysql -u root mysql
mysql> SET PASSWORD FOR root@localhost=PASSWORD('new_passwd');
mysql> SET PASSWORD FOR root@"hostname"=PASSWORD('new_passwd');
```

Figyeljünk arra, hogy alapesetben kettő felhasználó esetében is be kell állítanunk a jelszót.

Mivel a jogosultságok is adattáblában vannak tárolva, ezért adatbázis művelettel is megoldható.

```
mysql -u root mysql
mysql> UPDATE user SET Password=PASSWORD('new_passwd')
    -> WHERE user='root';
mysql> FLUSH PRIVILEGES;
```

Amennyiben a későbbiekben kapcsolódni akarunk a szerverhez, akkor szükséges a `-p` opció megadása a parancs esetében. Ebben az esetben ugyanis nem próbál meg jelszó nélkül csatlakozni, hanem bekéri a jelszót.

Jogosultság adása és elvétele felhasználóktól:
Jogosultság adás a GRANT utasítás segítségével történhet. Ennek általános szintaxisa a következő:

GRANT [privileges]
ON [db or table]
TO [user@host] IDENTIFIED BY 'password';

Példa jogosultság adásra:

GRANT SELECT ON test.* TO db1@localhost IDENTIFIED BY 'db1_jelszo';
GRANT SELECT, INSERT ON test.log TO db2@a-621 IDENTIFIED BY 'db2_jelszo';

Az első parancs a test nevű adatbázis összes táblájára vonatkozóan ad SELECT jogosultságok a db1 nevű felhasználónak, aki localhost-ról bejelentkezhet az utasítás végén megadott jelszóval.
A második parancs esetében az adott felhasználó két privilégiumot kap, de csak a test adatbázis log nevű táblájára vonatkozóan.

Jogosultság visszavonása a REVOKE parancs megadásával történhet. A parancsot például az alábbi módon használhatjuk:

REVOKE [privileges]
ON [db or table]
FROM [user@host];

Példa jogosultság megvonásra:
REVOKE SELECT ON test.* FROM db1@localhost;
REVOKE INSERT ON test.log FROM db2@a-621;

10.5.6. Adatbázis mentése és visszaállítása MySQL-ben

Gyakran lehet arra szükség, hogy egy adatbázis vagy annak egy adattábláját mentenünk kell, majd egy másik gépre átvinni (importálni). A MySQL alapértelmezésként SQL fájlt generált, amelyet, hogyha egy másik megfelelő verziójú MySQL-ben futtatunk, akkor automatikusan a mentett adatbázis létrejön.

A mentés elvégezhető a mysqldump parancs segítségével.

```
mysqldump -u root -p db1 table1 > db1_dump.sql
```

Amennyiben sikeresen beírjuk a rendszergazdai jelszót, úgy a `db1` adatbázis `table1` táblájának mentése belekerül a `db1_dump.sql` fájlba. Alapértelmezésként a parancs elhelyezi a mentés elején a tábla létrehozásához szükséges CREATE TABLE utasítást is. A `-t` kapcsolóval megoldható, hogy csak az adatok kerüljenek bele a fájlba, a `-d` kapcsolóval pedig csak a tábla létrehozását menti a fájlba, a benne lévő adatokat figyelmen kívül hagyja.

Visszaállításnál használhatjuk a mysql parancsot. Amennyiben az adattábla már létezik, akkor törölni kell a tartalmát vagy esetleg a táblát is. Amennyiben a mentés elején ott van a tábla létrehozó utasítás, akkor a következő paranccsal meg tudunk tenni egy visszaállítást:

```
mysql -u root -p db1 table1 < db1_dump.sql
```

10.6. Levelező szerver használata Linuxon

10.6.1. Az e-mail küldéssel kapcsolatos fogalmak és problémák

Az elektronikus levelezés már jó néhány éve a kommunikáció egyik fő eszköze lett. Az e-mail kezelésével kapcsolatban definiálnunk kell fogalmakat:

- MTA (Mail Transfer Agent): Az az alkalmazás, amely az elektronikus levelek fogadását és kézbesítését végzi (E)SMTP ((Extended) Simple Mail Transfer Protocol) vagy vele „rokon” protokoll segítségével. Ilyen alkalmazás például: exim, sendmail, qmail, smail, postfix, ...
- MDA (Mail Delivery Agent): Olyan alkalmazás, vagy démon, amely a levelek kézbesítését végzi. Ilyen alkalmazás például a procmail. A postfix esetében például a local vagy a virtual démonok látják el ezt a feladatot.
- MUA (Mail User Agent): Olyan alkalmazás, amely segítségével az elektronikus levelesládánkat kezelhetjük és a vele kapcsolatos alapvető műveleteket (olvasás, írás, törlés, nyomtatás, ...) elvégezhetjük. Manapság a legtöbb ilyen alkalmazás biztosít lehetőségeket a levelek szűrésére és különböző mappákba való csoportosítására. Példa MUA alkalmazásokra: mutt, pine, kmail, Mozilla Thunderbird (Idedove), SquirrelMail, ...

A levelezéshez számos alkalmazás szintű protokoll kapcsolódik:

- SMTP: Az MTA-k közti kommunikációra.
- POP3 (Post Office Protocol): Levelek letöltése és törlése
- IMAP (Internet Message Access Protocol): Levelek letöltése, mozgatása, törlése a szerveren.

Az utóbbi kettő protokollnak elérhetők olyan változatai is, amelyek esetében a kapcsolat titkosított (pl.: SSL felett). Ajánlott ezeknek a használata a nagyobb biztonság érdekében.
Az SMTP protokoll egy régi protokoll. Amikor kitalálták, akkor elég volt 7bites adatok átvitele. Ez az örökség a mai napig létezik. Amikor valamilyen bináris fájlt, vagy ékezetes betűt szeretnénk átvinni, abban az esetben szükséges az adataink 8bitesről 7 bitesre alakítása. Ami természetesen méret növekedéssel jár. Ahhoz, hogy mellékleteket lehessen csatolni, fontos kiterjesztés a MIME (Multipurpose Internet Mail Extension). Ezzel a levél tartalmát több részre bonthatjuk, és ahol szükséges a 8bites adatokat például Base64 kódolással 7 bitessé alakíthatjuk.
A levelezéshez kapcsolódik a levelek digitális aláírása. Ahhoz, hogy biztosak legyünk abban, hogy ki küldte nekünk a levelet ellenőriznünk kell a digitális aláírását. Sok levelező programban elérhető gpg (OpenPGP) bővítmény, amellyel megvizsgálhatjuk az aláírást. A program emellett, hogy aláírásra használható, akár egy levél titkosított küldésére is képes. Az SMTP protokoll megengedi az anonymous levél küldését.
Az elektronikus levelezés biztonsági szempontból egyáltalán nem tekinthető biztonságosnak. A szolgáltatás manapság természetes, mindenki használja éppen ezért a különféle vírusoknak, férgeknek kedvenc módszere a terjeszkedéshez. A másik fontos és egyre fokozottabban jelentkező probléma a levélszemét (spam, UCE). Ezekre a problémákra szerver oldalon reagálhatunk. A túl szigorú szűrés éppen úgy problémákat jelent, mint ha nem megfelelően történik a vírusok vagy a spam-ek detektálása és különválasztása. A következőkben egy MTA-t fogok bemutatni, illetve néhány beállítási lehetőségét.

10.6.2. Az Postfix

A Postfix hivatalos web oldala a http://www.postfix.com. A jegyzet írásakor a 2.5.1-es a legutolsó stabil kiadása. A Postfix nyílt forrású projekt, IBM fejlesztésből ered és a legtöbb Unix, BSD, Linux operációs rendszerben elérhető. A legfontosabb jellemzői:

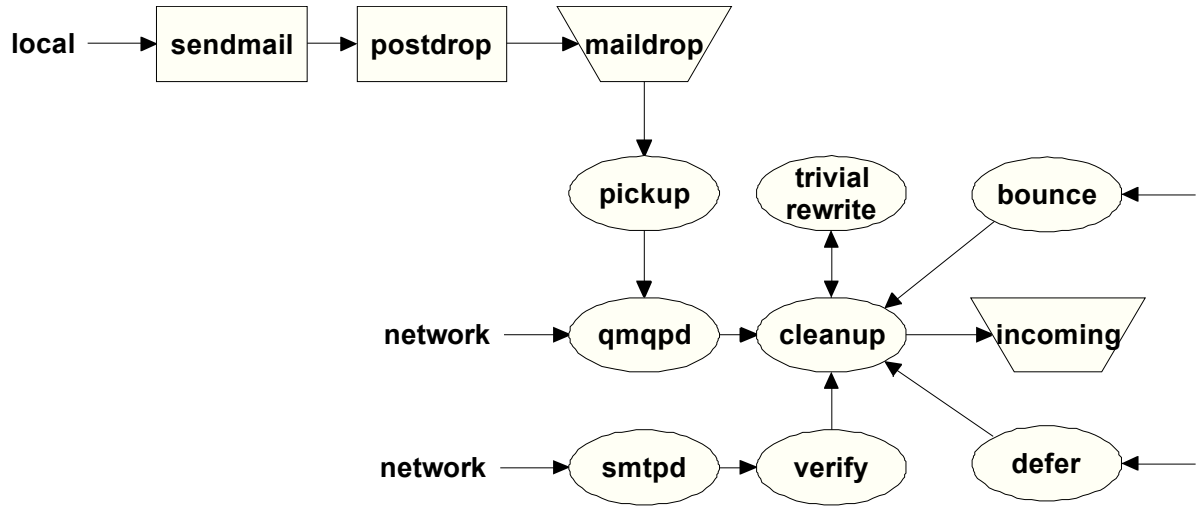
- A fejlesztésnél különösen hangsúlyosak a biztonsági kérdések.
- Moduláris felépítésű, könnyen konfigurálható és kibővíthető.
- Igyekszik kompatibilis maradni az sendmail-el.
- Képes szűrők használatára, amelyek segítségével spam és vírus szűrés megvalósítható (pl.: spamassassin, amavis, clamav).
- Megoldható a hozzáférés vezérlés szabályozása, belső levelezési listák védelme.
- Képes bizonyos beállításokat, felhasználói adatokat külső adatbázisokból venni (pl.: DBM, LDAP, MySQL, PostgreSQL, ...)
- Megoldható a felhasználók hitelesítése SASL (Simple Authentication and Security Layer) vagy TLS (Transport Layer Security) segítségével. A Cyrus vagy a Dovecot SASL implementációkkal együttesen használható. Így csak azok lesznek képesek elektronikus levél küldésére, akik erre jogosultságot kaptak.

- Virtuális felhasználói fiókok és tartományok hozhatók létre.
- Együttműködés a POP3 és IMAP kiszolgálókkal. (pl.: Dovecot)

Amikor a Postfix levelező szervert csomagból telepítjük, akkor az alábbi lehetőségek közül választhatunk:

- `No configuration` Ebben az esetben nem akarunk kezdeti konfigurációt.
- `Internet Site` Mint az internethez kapcsolódó gép, képes legyen a levél fogadásra és kézbesítésre.
- `Internet with smarthost` Ebben az esetben ez a gép fogadja az internet felől érkező leveleket és továbbadja egy másiknak (smarthost), amely helyezi el a felhasználók leveleit a megfelelő helyre.
- `Satellite system` Átjátszóként működik két szerver között.
- `Local only` Csak helyi kézbesítés.

A levelező szerver működését két részre bonthatjuk. Kell tudnia levelet fogadni és küldeni. Természetesen lehet olyan konfiguráció, amikor nem akarjuk mindkettőt használni. Az szerver működésére jellemző, hogy több folyamat együttes eredményeként lesznek az elektronikus levelek fogadva és elküldve. Az egyes folyamatok sorokat (queue) használnak az adattovábbítás folyamán. A fogadás addig tart, amíg az incoming queue-ba megérkezik az elektronikus levél. A következő ábrán a levél fogadásának folyamatát látjuk:



A Postfix e-mail fogadási folyamata: 3. sz. ábra

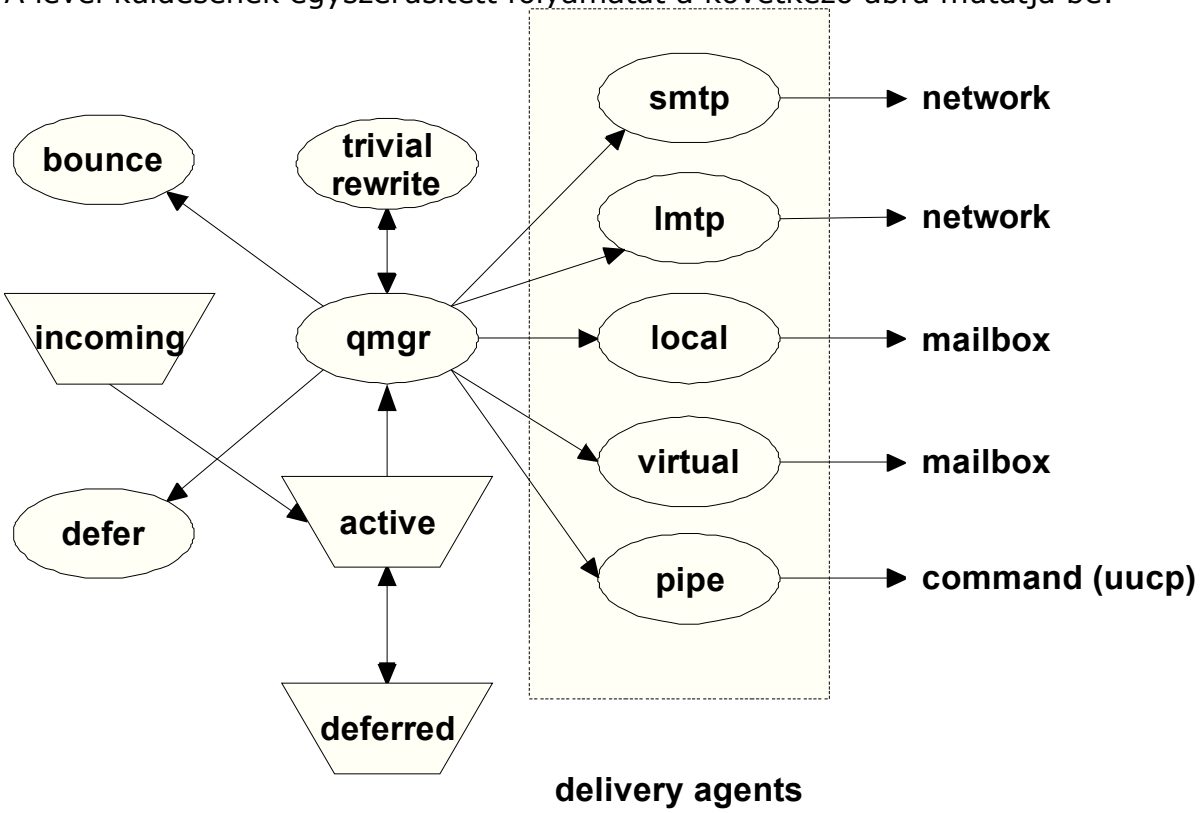
A Postfix esetében a master nevű démon irányítja az elektronikus levelezést. Ő indítja el az egyes démonokat és avatkozik be a rendszer működésébe a kapott visszacsatolások révén.
A fogadás esetében a fontosabb módszerek, lehetőségek:

- Egy elektronikus levél származhat egy helyi felhasználótól. Ebben az esetben a `sendmail` parancs a (kompatibilitás miatt) fogadja e levelet, majd átadja a `postdrop` parancsnak, amely elhelyezi azt a `maildrop` nevű sorban. Innét a `pickup` nevű démon szedi ki és továbbítja a `cleanup` felé.
- A hálózat felől érkező levelek esetében az SMTP-n vagy QMTP-n (Quick Mail Queuing Protocol (az SMTP gyorsítása a nagyobb teljesítmény érdekében))

keresztüli leveleket tudja fogadni. Ennek a megvalósítását az `smtpd` és a `qmqpd` végzi. A fogadásokról az `anvil` gyűjti a statisztikákat és figyeli, hogy kapcsolódások számát és értesíti erről a `master`-t.

Ezután a `cleanup` szerver végzi el a leveleken az utolsó módosításokat, például a From: és egyéb fejrész elemek hozzáadását. Fontos a szerepe abból a szempontból is, hogy fogadja a visszapattanó leveleket és a levelező szerver üzeneteit is. Ezután a levelek az `incoming queue`-ba kerülnek.

A levél küldésének egyszerűsített folyamatát a következő ábra mutatja be:



A Postfix e-mail küldési folyamata: 4. sz. ábra

A küldési folyamat „lelke” a `qmgr` (queue manager). Azért, hogy ne lehessen túlterhelni egy kis méretű sor (`active`) alapján történik az üzenetek feldolgozása és továbbítása az egyes kézbesítő ügynökök (delivery agent) számára. A `trivial-rewrite` démont cím átírásra, cím feloldásra és cím ellenőrzésre használhatja fel. A fontosabb üzenet kézbesítő démonok:

- `smtp` küldés hálózaton keresztül SMTP-vel
- `lmtp` küldés hálózaton keresztül LMTP (Local Mail Transfer Protocol)-al
- `local` helyi felhasználók postafiókjába
- `virtual` helyi „virtuális” felhasználók postafiókjába
- `pipe` levél továbbadása pipe felhasználásával (pl.: UUCP (Unix to Unix Copy) használatakor)
- `bounce` visszapattanó levelek (cleanup)
- `defer` késleltetett levelek (cleanup)

10.6.3. A Postfix néhány konfigurációs direktívája

A Postfix fő konfigurációs fájlokat a /etc/postfix könyvtárban találjuk. A master.cf fájlban találhatjuk a master démonnak szóló beállításokat. Egy adott szolgáltatáskor milyen programot kell indítani, és milyen paraméterekkel. A Postfix legtöbb démonja chroot környezetben fut a /var/spool/postfix könyvtáron belül.
A Postfix esetében is rengeteg opció közül választhatunk a konfiguráció során. A /etc/postfix/main.cf néhány beállítása:

```
command_directory = /usr/sbin
```
A Postfix parancsainak elérési útvonala.

```
daemon_directory = /usr/lib/postfix
```
A démonok elérési útvonala.

```
queue_directory = /var/spool/postfix
```
Az üzenetsorok elérési útvonala.

```
inet_interfaces = eth0
inet_protocols = ipv4
```
A hálózati interfész és a protokoll megadása, amit figyelni kell.

```
mail_owner = postfix
setgid_group = postdrop
```
A felhasználó, felhasználói csoport akinek a nevében a levelezőszerver működik.

```
default_privs = nobody
```
Az alapértelmezett felhasználó a helyi kézbesítéshez.

```
message_size_limit = 10240000
```
Maximális levél méret.

```
mail_spool_directory = /var/mail
```
A felhasználók leveleit tároló könyvtár.

```
syslog_facility = mail
syslog_name = postfix
```
A naplózási beállítások.

```
myhostname = mail.domain.hu
```
A levelezőszerver neve (FQDN).

```
mydomain = domain.hu
```
A tartománynév megadása.

```
myorigin = /etc/mailname
```
A helyi kimenő levelek esetében milyen nevet fűzzön a küldő neve után.

```
mynetworks = 127.0.0.0/8 192.168.31.0/24
```
Azon alhálózatok, ahonnét engedélyezett a levél küldése (mail relay).

```
mydestination = $myhostname, $myorigin, $mydomain, localhost.
$mydomain, , localhost
```
Azon címek listája, amelyek esetében a levelezőszerver úgy veszi, hogy ő a célállomás.

```
alias_database = hash:/etc/aliases
alias_maps = hash:/etc/aliases
```
A fájl, ahol az álnevek találhatók. A levelek kézbesítése más felhasználó levelesládájába.

```
relayhost = mail.szolgaltato.hu
```
Adott hoszt neve vagy IP címe, aminek egy belső hálózat felől továbbítani kell a levelet, és az fogja kézbesíteni a célhelyre.

```
smtpd_recipient_restrictions =
        permit_mynetworks,
        reject_invalid_hostname,
        reject_non_fqdn_hostname,
        reject_non_fqdn_sender,
        reject_non_fqdn_recipient,
        reject_unknown_sender_domain,
        reject_unknown_recipient_domain,
        reject_unauth_destination,
        reject_rbl_client cbl.abuseat.org,
        permit
```

Fontos annak a beállítása, hogy a szerverünk milyen feltételeknek megfelelően fogadja és kézbesítse az elektronikus leveleket. Nem célszerű az open relay, mert órákon belül „fekete listára” kerülhetünk a spam-elés miatt. Csak olyan helyről célszerű leveleket fogadni, amely hosztok rendelkeznek regisztrált tartománnyal és FQDN-el.

A levelező szerver csak azon tartományokra címzett leveleket fogadja el, amelyek megegyeznek a saját doménnévvel, vagy pedig a virtuális domének között szerepelnek.

A `postconf` parancs használatával megtekinthetjük az összes konfigurációs paramétert, illetve azoknak beállított és alapértelmezett értékeit.

10.6.4. A Postfix adminisztrációs parancsai

A `/etc/init.d/postfix` script segítségével a levelezőszerver indítható, leállítható, újraindítható.

- `mailq` levélküldő sor tartalmának listázása
- `postalias` alias adatbázis karbantartása
- `postcat` sorban lévő levél tartalmának megtekintése
- `postconf` konfiguráció megtekintése, beállítása
- `postdrop` levél elhelyezése a `maildrop` sorba
- `postfix` indítás, leállítás
- `postkick` üzenetküldés Postfix szolgáltatásnak

- `postlock` a leveleket tartalmazó könyvtár zárolása és parancs futtatása
- `postlog` naplózás
- `postmap` „lookup táblák” közti átalakítás
- `postqueue` sorok vezérlése
- `postsuper` sorok vezérlése amikor a levelezőszerver nem fut

10.7. Tűzfal használata Linuxon

10.7.1. A tűzfalak általános jellemzői, típusaik, elvek

A tűzfal a hálózati védekezés fontos eszköze. A segítségével meghatározhatjuk, hogy milyen forgalom juthat be a hálózatba illetve milyen helyekre lehessen kifelé adatokat küldeni. A tűzfal segítségével tehát védjük a hálózatunkon lévő számítógépeinket, ezen keresztül az adatainkat. Közvetetten mondhatjuk azt, hogy az adott cég vagy szervezet jó hírét is. Fontos, hogy a tűzfalat olyan helyre kell elhelyeznünk, hogy mindenfajta hálózati forgalom csakis rajta keresztül történhessen. A gyakorlatban ez annyit jelent, hogy több hálózati interfésszel rendelkező eszközt vagy számítógépet iktatunk a hálózati eszközök közé. Ezután definiálnunk kell azt, hogy melyik eszközről melyik irányokba történhet kommunikáció és milyen típusú.

A tűzfal esetében kérdés, hogy milyen eszközt használjunk a megvalósításához:
- célhardvert (hardveres tűzfal)
- szoftveres megvalósítást (pl. Linux, OpenBSD, ...)

A szoftveres megoldás általában olcsóbb és valamivel kisebb teljesítményt nyújt mint a hardveres megvalósítások. Az, hogy melyiket célszerű választanunk több tényezőtől is függ.

Amikor a védekezésről beszélünk megkülönböztethetünk különböző működési modelleket:
- helyi védekezés: egy szerver esetében fontos, hogy védekezzünk a szerver túlterhelése ellen illetve korlátozzuk az egyes szolgáltatások elérhetőségét
- tűzfallal védett LAN: ebben az esetben a tűzfal két hálózatot választ el egymástól. Az egyik a „külső” hálózat, ahonnét várhatók a támadások, míg a másik a „belső” hálózat amit védeni szeretnénk. Ebben az esetben kettő hálózati interfészre van szükségünk.
- tűzfallal védett LAN DMZ (De-militarized Zone)-vel kiegészítve. Ez annyiban különbözik az előzőtől, hogy a LAN-ban lévő gépek egy csoportját elkülönítjük. Azok a gépek, amelyek valamilyen szolgáltatást biztosítanak kerülhetnek bele ebbe a csoportba. A tűzfal ebben az esetben három interfésszel kell, hogy rendelkezzen. A módszer előnye az, hogy szabályozni lehet, hogy milyen módon lehessen a DMZ-ben lévő szerverekhez hozzáférni a külső hálózat felől, illetve a belső LAN-beli gépekről. Ebben az esetben általában a LAN-beli gépek külső elérése tiltott.

A tűzfalakat működési módjuk szerint különböző csoportokba sorolhatjuk:
- **csomagszűrő tűzfalak**: ebben az esetben a tűzfal képes az összes hozzá

érkező csomagon ellenőrzéseket végrehajtani. Az IP csomag fejrészének minden mezője alapján lehetséges a szabályok megfogalmazása. Természetesen az IP-ben utazó szállítási protokollokra vonatkozóan is lehetséges szűréseket tenni. A tűzfal ebben az esetben csak portokkal foglalkozik, nem képes annak a vizsgálatára, hogy ott milyen kommunikáció folyik.

- **dinamikus csomagszűrő (csomagvizsgáló) tűzfalak**: annyival több, mint az előző, hogy képes az egyes protokollok állapotainak a figyelembe vételére. Ennélfogva képes megkülönböztetni, hogy egy adott adatfolyam egy meglévő kapcsolathoz tartozik-e, vagy pedig egy teljesen új.
- **Alkalmazás szintű tűzfalak**: képesek adatfolyamba való beletekintésre, ami alapján el tudják dönteni a kommunikáció típusát és szükségességét.
- **Proxy tűzfalak**: Szintén az alkalmazási szinthez köthetők. Egy proxy esetében megvalósítható, hogy felhasználói névhez és jelszóhoz kössük a kommunikáció lehetőségét.
- **NAT (Network Address Translation) útválasztók**: Ebben az esetben egy gép végzi a címfordítást és rajta keresztül történhet a belső hálózatból történő kommunikáció. A módszer előnye, hogy elrejti a belső hálózatot, hátránya viszont az, hogy nagyobb erőforrást igényel a működéséhez. A belső hálózat összes gépe ebben az esetben egy IP cím mögül fog látszani a külső hálózat gépei számára. Amennyiben egy szolgáltatás esetében IP cím alapján van adott számú kapcsolódási lehetőség, előfordulhat, hogy azért nem érjük el a szolgáltatást, mert valaki más a belső hálózatunkból épp használja azt.
- **Személyes tűzfalak**: Ebben az esetben a cél egy önálló rendszer védelme. A külső támadások mellett előfordulhatnak belső támadások is. Ezek ellen csak úgy védekezhetünk, hogy korlátozzuk a szolgáltatások hozzáférhetőségét.

Egy tűzfal csak akkor ad biztonságot, hogyha jól be van állítva. Amennyiben „túl megengedő”, akkor olyan adatforgalom is áthaladhat rajta, amely nem kívánatos lenne. Ahhoz, hogy ilyen probléma ne jelentkezzen ajánlott néhány elvet, szabályt betartani.

- **Mindent tilos, kivéve amit szabad elv!** Elsőként tiltsunk le mindenféle forgalmat, majd a szükségeseket engedélyezzük.
- Tiltsuk le a „source route” IP opciót tartalmazó csomagok átengedését. Egy csomag ne rendelkezhessen a saját útválasztására vonatkozóan.
- Védekezzünk az IP hamisítás (IP spoofing) ellen. Mindig ellenőrizzük, hogy egy IP címről érkezett csomag a megfelelő interfészen keresztül érkezett-e. Ne engedjük, hogy egy külső helyről azt próbálják meg elhitetni, hogy az adatforgalom egy belső IP címről érkezett.
- Védekezzünk az elárasztás (flooding) és a DoS típusú támadások ellen, úgy, hogy korlátozzuk egy adott időintervallumra vonatkozó kapcsolatok számát.
- Figyeljünk oda a naplózásra, a naplófájlok méretére.

10.7.2. A kernel szintű tűzfal

A Linux esetében kernel szintű tűzfalról beszélhetünk. A tűzfal megvalósítása

kernel szintjén történik. Amennyiben a kernelünk rendelkezik a megfelelő támogatással, akkor definiálhatjuk a tűzfalunk szabályait. A szabályok meghatározása felhasználói térben futó alkalmazások segítségével történhet.
A következő alkalmazások segítségével definiálható a tűzfal az egyes kernel verziók esetében:

- 1.1 ipfw
- 2.0 ipfwadmin
- 2.2 ipchains
- 2.4, 2.6 iptables

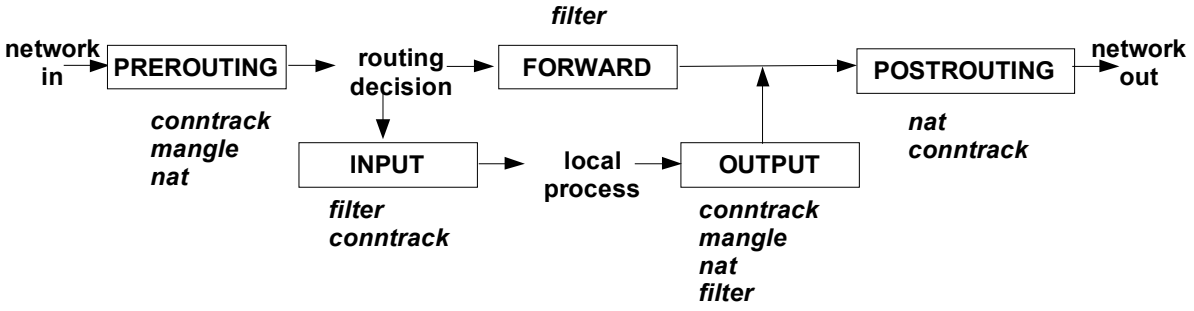
A kernel szintű tűzfalra jellemző, hogy a netfilter struktúrára (http://www.netfilter.org) épül. Számunkra külön örvendetes hír, hogy netfilter core-team tagjai között magyar fejlesztőt is találunk Kadlecsik József (KFKI) személyében. A netfilter struktúra úgynevezett „kampókat" (hook) definiál a protokoll stack megfelelő pontjain. Amikor egy adott csomag eljut az adott pontra, akkor átadódik a netfilter keretrendszernek. A kernel modulok az egyes hook-okhoz regisztrálhatják magukat. A kernel modul regisztrációja során egy prioritást rendel a saját funkciójához. Amikor az adott hook aktiválódik, akkor a prioritás dönti el, hogy mely modul kerül meghívásra. A tűzfal a kernel szempontjából egy speciális eszköznek tekinthető. Modulok segítségével a működése kiterjeszthető.
A keretrendszerben lévő szoftverek segítségével megoldható:

- a csomagszűrés (packet filtering) – filter alrendszer
- a hálózati cím és port fordítás (network address [and port] translation NA[P]T) – NAT alrendszer
- csomag megváltoztatás (packet mangle) – mangle alrendszer
- kapcsolat követés (connection tracking) – conntrack alrendszer

Az iptables egy általános táblázatokat tartalmazó struktúra, amelyben a szabályokat definiálhatjuk. Egy szabály két részből áll:

- iptables matches: ez jelenti az egyezési feltételeket
- iptables target: ez jelenti a teendőt, amelyet egy feltétel esetében el kell végezni

Az újabb kernelek esetében a szabályokat az `iptables` parancs segítségével definiálhatjuk. A szabályokat minden rendszerinduláskor be kell tölteni. Van arra lehetőség, hogy a szabályokat csoportosítsuk és saját láncokat hozzunk létre.



Egy csomag útja a csomagszűrőben: 5. sz. ábra

Az előző ábra bemutatja, hogy miként halad keresztül egy csomag a

csomagszűrőben:

Az ábra bal oldalán érkezik meg egy csomag a hálózat felől.
A PREROUTING alapértelmezett lánc csak abban az esetben létezik, hogyha a NAT bővítménnyel rendelkezünk.
Itt a következő netfilter alrendszereket használhatjuk:

- conntrack
- DNAT (Destination NAT)
- mangle

Azokat a feltételeket adhatjuk meg itt, amelyeket még az útválasztás (routing) előtt kell elvégezni. Például:

- amennyiben NAT-ot használunk és egy belső hálózatbeli gép adott portját szeretnénk kívülről elérhetővé tenni
- ha transzparens proxy-t szeretnénk használni

A PREROUTING után érkezik meg a csomag az útválasztáshoz. Itt dől el, hogy a csomag a helyi gépnek szól vagy pedig továbbítani kell a csomagot egy másik interfészen keresztül.
Az INPUT alapértelmezett láncban adhatjuk meg a szabályokat a bejövő kapcsolatokra vonatkozóan. Itt a következő netfilter alrendszereket használhatjuk:

- conntrack
- filter

Ahhoz, hogy egy felhasználói folyamat (local process) megkaphasson egy csomagot, azt egy INPUT láncbeli szabálynak engedélyeznie kell.

A FORWARD lánc esetében adhatjuk meg a szabályokat az IP továbbításra vonatkozóan. Ahhoz, hogy a kernel képes legyen az egyes interfészek között IP továbbításra, ahhoz külön rendszerbeállítás szükséges a /proc fájlrendszeren keresztül:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

A FORWARD lánc esetében csak csomagszűrési szabályokat használhatunk.

Amennyiben egy felhasználói folyamat kommunikálni szeretne hálózaton keresztül, akkor rá az OUTPUT láncban lévő szabályok vonatkoznak. A kifelé való kommunikációhoz feltétlenül szükséges az, hogy legyen egy olyan szabály, amely megengedi a kommunikációt. Az OUTPUT esetében bármelyik netfilter alrendszert használhatjuk.

Az OUTPUT és a FORWARD láncon túljutó csomagok ezután a POSTROUTING lánchoz kerülnek. Ez a PREROUTING lánchoz hasonlóan csak akkor létezik, hogy a NAT támogatás elérhető a kernelben. Itt a következő netfilter alrendszereket használhatjuk:

- conntrack
- SNAT (Source NAT)

Az alapértelmezett láncok rendelkeznek ún. default policy-val. Itt azt lehet megfogalmazni, hogy mit kell tenni azzal a csomaggal, amelyik egyik korábbi szabályra sem illeszkedik.

Az iptables esetében többféle target (teendő) is megadható. Alapértelmezésként kettő biztosan elérhető:

- ACCEPT: a csomag elfogadásra kerül
- DROP: a csomag eldobásra kerül.

10.7.3. Az iptables parancs használata, paraméterei

Egy iptables parancs több részből állhat:

```
iptables [-t table] command chain match [target/jump]
```

Ahol:

- `table`: egy táblázat neve, például: `nat`
- `command`: egy parancssori kapcsoló
- `chain`: a lánc neve
- `match`: az egyezési feltétel megadása
- `target`: teendő
- `jump`: egy létező lánc neve, amire ugrani kell

Láncokkal kapcsolatos fontosabb műveletek:

- `-N chain` új lánc létrehozása
- `-X chain` létező lánc törlése
- `-P chain target` alapértelmezett teendő beállítása beépített lánc esetén
- `-L` lánc tartalmának listázása
- `-F chain` láncban lévő szabályok törlése
- `-Z chain` lánchoz tartozó számlázók nullázása

A `chain` egy adott lánc nevét jelenti, a `target` pedig a teendőt.

Példák iptables parancsokra:

- `iptables -N allowin`
- `iptables -P FORWARD DROP`
- `iptables -L`
- `iptables -L -t nat`

Az alapértelmezett láncokból van arra lehetőségünk, hogy egy korábban létrehozott saját láncra ugorjunk. Ebben az esetben addig történik a saját láncokban lévő szabályok vizsgálata, amíg egyértelműen el nem dől, hogy el kell fogadni vagy el kell dobni a csomagot. Amennyiben a láncon belül ez nem következik be, akkor az ugrás helye utáni szabály kerül feldolgozásra.

Láncban lévő szabályokkal kapcsolatos műveletek

- `-A rule` új szabály hozzáfűzése
- `-I [num] rule` szabály beszúrása adott pozícióba
- `-R num rule` szabály helyettesítése
- `-D rule | num` szabály törlése

Általános egyezési feltételek megadása:

- `-s [!] address/mask` forrás IP cím/ alhálózati maszk
- `-d [!] address/mask` cél IP cím/ alhálózati maszk
- `-p [!] protocol` szállítási protokoll neve
- `-i [!] interface` bejövő hálózati interfész neve
- `-o [!] interface` kimenő hálózati interfész neve
- `[!] -f` töredékek kezelése

Protokoll függő egyezési feltételek megadása:

TCP esetében:

- `--tcp-flags ...` TCP flag-ek (SYN, RST, ACK, ...) megadása
- `--sport [!] port` forrásport
- `--dport [!] port` célport
- `--tcp-option [!] id` TCP opciók
- `[!] --syn` ahol a SYN flag be van állítva (RST,ACK,FIN pedig nincsen, új TCP kapcsolat kezdeményezésekor)
- `--mss value[: value]` maximális csomagméret

UDP esetében:

- `--sport port` forrásport
- `--dport port` célport

ICMP esetében

- `--icmp-type` ICMP típusának megadása (echo-request, ...)

Kibővített egyezési feltételek (kiterjesztések, bővítmények) használata:

`-m extension` bővítmény nevének a megadása

state: csomag állapotának vizsgálata

```
-m state --state state-list
```

A state-list esetében a NEW (új), ESTABLISHED (felépült), RELATED (valamihez tartozó), INVALID (nem megfelelő) szavak használhatók.

Példák:

```
iptables -A FORWARD -o eth0 -j ACCEPT
iptables -A FORWARD -o eth1 -m state \
--state ESTABLISHED,RELATED -j ACCEPT
```

limit: kapcsolatok számának vizsgálata

```
-m limit
```

`--limit M/second` másodpercenként M-nél kisebb számok esetén legyen egyezés

`--limit-burst N` az N-nél kisebb csomagszám esetén legyen egyezés

Példa:

```
iptables -A FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT
```

mac: egyezés vizsgálat MAC cím alapján
`-m mac`

`--mac-source MAC`	adott MAC cím alapján

```
iptables -A INPUT -m mac --mac-source 00:50:56:00:00:02 -j ACCEPT
```

recent: egyezés „korábbi” IP címek alapján
`-m recent`

`--name str`	lista azonosító karakterlánc
`--set`	IP cím felvétele
`--update`	utolsó időpont frissítése
`--seconds N`	időkorlát megadása
`--hitcount M`	egyezések száma

```
iptables -A ssh -m recent --set --name ssh_counter
iptables -A ssh -m recent --update --name ssh_counter --seconds 60
--hitcount 2 -j blacklist
```

owner: egyezés ID-k alapján
`-m owner`

`--uid-owner userid`	UID alapján
`--gid-owner groupid`	GID alapján
`--pid-owner processid`	PID alapján

```
iptables -A OUTPUT -m owner --uid-owner 1000 -j ACCEPT
```

multiport: több port felsorolása vesszővel elválasztva
`-m multiport`

`--sport port1,port2,...`	forrás port
`--dport port1,port2,...`	cél port

```
iptables -A INPUT -p tcp -m multiport --sport 80,443 -j ACCEPT
```

IP range: több IP cím megadásának lehetősége
`-m iprange`

`--src-range startIP-endIP`	forrás IP címek
`--dst-range startIP-endIP`	cél IP címek

```
iptables -A INPUT -p tcp -m iprange \
--src-range 192.168.30.101-192.168.31.121
```

mark: korábban megjelölt csomagokra vonatkozó egyezés
`-m mark`

`--mark ID` jelölés azonosító számának megadása

```
iptables -t mangle -A INPUT -m mark --mark 1 -j DROP
```

Példa egy otthoni internetkapcsolat védelmére:

```
iptables -N block
iptables -A block -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A block -m state --state NEW -i ! ppp0 -j ACCEPT
iptables -A block -j DROP
iptables -A INPUT -j block
iptables -A FORWARD -j block
```

Ebben az esetben saját magunk kezdeményezhetünk kifelé menő kapcsolatokat, viszont kívülről nem építhetnek fel a gépünkhöz irányuló kapcsolatot.

Néhány lehetőség a teendő (target) megadására:

LOG: naplózás
`--log-level 7-0 (debug, info, notice, warning, error, crit, alert, emerg)` syslog naplózási „szintek”
`--log-prefix "string"` azonosító sztring

```
iptables -A INPUT  -m limit --limit 1/second -j LOG  --log-level
info --log-prefix "PORTSCAN"
```

REJECT: eldobás ICMP unreachable, prohibited vagy TCP RST csomag küldésével

```
iptables -A FORWARD -p tcp --dport 113 -j REJECT --reject-with tcp-
reset
```

RETURN: saját lánc esetében visszatérés a szülő láncba és a következő szabály alkalmazása (egyébként pedig a default policy)

```
iptables -A allowin -p tcp --dport 80 -j RETURN
```

SNAT: forrás IP cím átírása a kimenő interfész IP címére
`--to-source startIP-endIP:port1-port2`

```
iptables   -A   POSTROUTING   -s   172.16.0.0/16   -o   ppp0   \
 -j SNAT --to-source 192.168.2.2
```

Amennyiben a ppp0 interfész fix IP címmel rendelkezik, úgy a belső hálózatbeli gépek forgalmát a saját IP címre módosítva megoldható az internetkapcsolat megosztása.

MASQUERADE: az SNAT speciális esete, amikor például a kimenő interfész dinamikusan kapja az IP címet.

```
--to-ports port1-port2

iptables -t nat -A POSTROUTING -p tcp -j MASQUERADE \
--to-ports 2000-3000
```

DNAT: cél IP cím (port) átírása valamilyen belső IP címre (portra)
`--to-destination startIP-endIP:port1-port2`

Használható port-forward-ra, amely esetben a külső interfész felől érkező adott portra érkező csomagokat egy belső hálózaton lévő géphez továbbíthatunk.

```
iptables -t nat -A PREROUTING -i ppp0 -p tcp --dport 80 -j DNAT \
--to-destination 172.16.0.3
```

REDIRECT: a DNAT egy speciális esete, amikor saját port(ok)ra történik a csomagok átirányítása
`--to-ports port1-port2`

A segítségével készíthetünk transzparens proxy-t:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-
ports 3128
```

A működéshez szükség van a például a Squid-re, ami a 3128-as TCP porton figyel és be van állítva, hogy képes legyen transzparens módban is működni.

MARK: adott csomagok megjelölése a **mark** egyezési feltétel számára
`--set-mark N` adott szám hozzárendelése a csomagokhoz

```
iptables -t mangle -A PREROUTING -p tcp --dport 22 \
-j MARK --set-mark 2
```

Szabályok mentése és visszaállítása:

Ahhoz, hogy a szabályaink megmaradjanak ajánlott, hogy mentsük őket, és amikor a gépünk indul automatikusan beállításra kerüljenek.

Az iptables-save parancs a standard kimenetére menti a szabályokat. Az egyes sorokat ugyanazzal a megadási móddal adja vissza, mint amit az iptables parancs esetében használhatunk. Annyi a különbség, hogy az iptables parancs nem kerül bele a kimenetbe.

```
# Generated by iptables-save v1.3.6 on Fri Apr 11 22:00:00 2008
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A PREROUTING -d 192.168.2.2 -p tcp -m tcp --dport 80 -j DNAT \
--to-destination 172.16.0.3
-A POSTROUTING -s 172.16.0.0/255.255.255.0 -o eth0 -j SNAT \
--to-source 192.168.2.2
COMMIT
# Completed on Fri Apr 11 22:00:00 2008
# Generated by iptables-save v1.3.6 on Fri Apr 11 22:00:00 2008
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p udp -m udp --dport 53 -j ACCEPT
COMMIT
# Completed on Fri Apr 11 22:00:00 2008
```

A * után adjuk meg az alrendszer nevét, a : után pedig a lánc nevét, amely az adott alrendszeren belül található és a hozzá tartozó alapértelmezett teendőt. A többi sor ugyanolyan, mintha iptables parancsot használtunk volna. Az iptables-restore parancs képes arra, hogy ilyen formátumú fájlt olvasson és a benne megadott szabályokat beállítsa.

Amikor tűzfalat konfigurálunk, akkor az egyik lehetőség az, hogy scriptet írunk, amelybe az egyes parancsokat megadjuk. A másik lehetőség az, amikor a tűzfal konfigurálását úgy oldjuk meg, hogy fájlt szerkesztünk vagy generálunk, és azt állítjuk be az adott szituációban.

10.7.4. A tűzfal működéséhez szükséges kernel opciók

Ahhoz, hogy a kernel szintű tűzfal esetében az egyes netfilter alrendszereket és a megfelelő targeteket használjuk, annak feltétele, hogy a kernelbe fordítsuk be a megfelelő opciókat.

A Networking/Networking options menüpont alatt találhatjuk meg a jegyzetben bemutatott példákhoz szükséges opciókat:

Packet Socket - CONFIG_PACKET
TCP/IP networking - CONFIG_INET
Network packet filtering framework (Netfilter) - CONFIG_NETFILTER
  Core Netfilter Configuration
    Netfilter connection tracking support - CONFIG_NF_CONNTRACK_ENABLED
    FTP protocol support - CONFIG_NF_CONNTRACK_FTP
    Netfilter Xtables support (required for ip_tables) -
            CONFIG_NETFILTER_XTABLES
    "connlimit" match support" - CONFIG_NETFILTER_XT_MATCH_CONNLIMIT
    "limit" match support – CONFIG_NETFILTER_XT_MATCH_LIMIT
    "mac" address match support – CONFIG_NETFILTER_XT_MATCH_MAC
    "mark" match support – CONFIG_NETFILTER_XT_MATCH_MARK
    Multiple port match support – CONFIG_NETFILTER_XT_MATCH_MULTIPORT
    "state" match support – CONFIG_NETFILTER_XT_MATCH_STATE
  IP: Netfilter Configuration
    IPv4 connection tracking support (required for NAT) -
            CONFIG_NF_CONNTRACK_IPV4
    IP tables support (required for filtering/masq/NAT) -
            CONFIG_IP_NF_IPTABLES
    IP range match support - CONFIG_IP_NF_MATCH_IPRANGE
    recent match support – CONFIG_IP_NF_MATCH_RECENT
    Owner match support – CONFIG_IP_NF_MATCH_OWNER
    Packet filtering – CONFIG_IP_NF_FILTER
      REJECT target support – CONFIG_IP_NF_TARGET_REJECT
    TTL match s. - CONFIG_IP_NF_MATCH_TTL
    LOG target support - CONFIG_IP_NF_TARGET_LOG
    Full NAT – CONFIG_NF_NAT
      MASQUERADE target support – CONFIG_IP_NF_TARGET_MASQUERADE
      REDIRECT target support – CONFIG_IP_NF_TARGET_REDIRECT
    Packet mangling – CONFIG_IP_NF_MANGLE

11. Irodalomjegyzék

11.1. Könyvek, kiadványok

[1] Fred Butzen, Christopher Hilton: Linux hálózatok, Kiskapu Kft., 1999
[2] Marcel Gagné: Linux rendszerfelügyelet, Kiskapu Kft., 2002
[3] Rob Flickenger: Linux bevetés közben, Kiskapu Kft., 2003
[4] Pere László: GNU/Linux rendszerek üzemeltetése I.-II., Kiskapu Kft., 2005
[5] Tony Bautts, Terry Dawson, Gregor N. Purdy: Linux hálózati adminisztrátorok kézikönyve, Kossuth Kiadó ZRt, 2005
[6] Gerrit Huizenga, Badari Pulavart, Sandra K. Johnson: Linux kiszolgálók teljesítményének fokozása, Kiskapu Kft., 2006
[7] Himanshu Dwivedi: SSH a gyakorlatban, Kiskapu Kft., 2004
[8] Daniel Lopez: Apache zsebkönyv, Kiskapu Kft., 2007
[9] David Collier-Brown, Robert Eckstein, Peter Kelly: Samba, Kossuth Kiadó ZRt., 2001
[10] Julie C. Meloni: Tanuljuk meg a MySQL használatát 24 óra alatt, Kiskapu Kft., 2003
[11] Gregor N. Purdy: Linux iptables zsebkönyv, Kiskapu Kft., 2006
[12] Michael D. Bauer: Szerverek védelme Linuxszal, Kossuth Kiadó ZRt, 2003
[13] Lars Wirzenius: Linux rendszeradminisztrátorok kézikönyve, (elektronikus, FDL), 2003
[14] Daniel P. Bovet - Marco Cesati: Understanding the Linux kernel, O'Reilly, 2000

11.2. Internetes hivatkozások

[1] http://www.debian.org – Debian/GNU Linux
[2] http://www.debian.hu – Debian/GNU Linux magyar
[3] http://debian.lap.hu – Debian link gyűjtemény
[4] http://tldp.fsf.hu – Magyar Linux Dokumentációs Projekt
[5] http://www.hup.hu – Hungarian Unix Portal
[6] http://wiki.hup.hu – HUP wiki
[7] http://www.linuxvilag.hu – Linuxvilág magazin
[8] http://www.szabilinux.hu – Linux Dokumentációk Magyarul
[9] http://www.kernel.org – Linux kernel
[10] http://www.openssh.com – OpenSSH
[11] http://www.apache.org – Apache Software Foundation
[12] http://www.proftpd.org – The ProFTPD projekt
[13] http://www.samba.org – Samba
[14] http://www.mysql.com – MySQL
[15] http://www.postfix.org – Postfix
[16] http://www.netfilter.org – The netfilter.org project