Sikos László:

Bevezetés a Linux használatába

BBS-INFO Kiadó – 2005.

A könyv nagyobb mennyiségben megrendelhető a kiadónál:
BBS-INFO Kft. 1630 Bp. Pf. 21. Tel.: 407-17-07

ISBN 963 9425 00 1

Kiadja a BBS-INFO Kft.
1630 Budapest, Pf. 21.
Felelős kiadó: a BBS-INFO Kft. ügyvezetője
Készült a Debreceni Kinizsi Nyomdában
Felelős vezető: Bördős János

TARTALOMJEGYZÉK

ELŐSZÓ

Manapság egyre több helyen találkozhatunk a Linux operációs rendszer mintegy százféle disztribúciója közül valamelyikkel. Láthatjuk az iskolában, egy hivatalban vagy valamelyik ismerősünknél. Az is előfordulhat, hogy laptop-ot szeretnénk vásárolni és mikor ki szeretnénk próbálni, kiderül, hogy a masinán még nincs operációs rendszer. A papírdobozában azonban van egy CD, melyen például egy SuSE Linux van. Ugyanez előfordulhat asztali PC vásárlásakor is egy bevásárlóközpontban.

Akik már évek óta foglalkoznak a személyi számítógépekkel, azok között is jócskán vannak olyanok, akiknek az operációs rendszer egyetlen alternatívája a Microsoft Windows. Nem hallottak sosem az IBM OS/2-jéről vagy a FreeBSD-ről, netán a Linux-ról. Vagy ha hallottak is, legfeljebb csak annyit tudnak róluk, hogy operációs rendszerek.

Talán a PC-k elmúlt két évtizedes pályafutása a magyarázata annak, hogy az akkori DOS-ból kiindulva a 16 bites Windows 3.x-en keresztül a Windows XP felé vettük az irányt. Nem ez az egyetlen lehetőség azonban egy személyi számítógép képességeinek maximális kihasználására. Sokan eleve elvetik az ötletet, hogy más rendszereket is megismerjenek. Napjaink programjai közül rengeteg elérhető Windows platformra (sok csak ez alá), így sokakat nem is foglalkoztat más operációs rendszerek használatának gondolata. Akik régebben kezdték az ismerkedést a PC-kkel és a DOS rendszer volt az első platform, amin dolgoztak, nem látják értelmét, miért is kellene újra visszatérni a karakteres felületre, amikor végre már eljutottunk oda, hogy egy csillogó-villogó grafikus felületen dolgozhatunk úgy, hogy a hardvereszközök kezelésével és más fontos feladatokkal egyáltalán nem vagy csak alig kell foglalkoznunk. Ha már úgyis ott van az óriási merevlemez, a hatalmas memória, a hihetetlen számolási kapacitással rendelkező processzor, miért forduljunk vissza a törtfehér színű karakterekhez, miért tanuljunk meg sok-sok parancsot, hogy kezelni tudjuk a számítógépet?

Az első kérdés, amit meg kell válaszolnunk, az az, hogy biztosan jelentősen megnehezíti-e a munkánkat, ha karakteres felületen dolgozunk? Másrész jöhetne még az a megjegyzés is, hogy bizony-bizony a Linux-on is van lehetőség

a grafikus felület használatára, de ha a rendszer mélyére szeretnénk látni, akkor ott van alapból 6 db karakteres terminál is...

A Windows rendszerek egyik nagy problémája hosszú ideig az volt, hogy a stabilitás kérdése nem volt megoldott. Másik probléma a hálózati adatvédelem, hiszen a Windows 9.x sorozat önálló PC-re készült, csak támogatta az internet használatát is. Az NT ág hálózati képességei jócskán túlmutattak a 9.x-es sorozatán, így a két ág összeért. A Windows XP napjaink egyik legnépszerűbb operációs rendszere hazánkban, de egyre többen látják meg a lehetőséget, hogy egy másik operációs rendszert is választhatnának.

Mi indokolja a Linux színrelépését és egyre szélesebb körű elterjedését? Mint majd látni fogjuk, a UNIX rendszerre épül, ami a nagygépes környezetek első operációs rendszerei között jelent meg és azóta töretlenül fejlődött. A robosztus felépítés, a szinte határtalan hálózatkezelési képességek, a stabilitás és végül, de nem utolsósorban az ingyenesség indokolja, hogy egyre többen próbálják ki a rendszert. Az informatikusoknak, a szakembereknek, a programozóknak további óriási előny a kereskedelmi operációs rendszerekkel szemben a nyílt forráskód. Míg a Windows több millió kódsorának működése meglehetősen rejtélyes, addig az érdeklődő megismerheti, a hozzáértő akár meg is változtathatja a Linux forráskódját. Ez persze a legtöbb Linux programra is igaz, így bárki elkészítheti a saját programverzióját a megadott (nem túl szigorú) feltételek mellett. Sokkal több tehát a lehetőségünk Linux alatt, de valóban igaz az is, hogy nagyobb számítástechnikai tudást igényel, mint társai. Cserébe azonban igazi urai lehetünk rendszerünknek és nem kell elszenvednünk az érthetetlenül lefagyó ablakokat.

Ha ki szeretnénk próbálni a Linux-ot, esetleg vannak olyan programjaink, melyek csak ezen a platformon elérhetők, nem kell feltétlenül lemondanunk a Windows-ról sem. A két rendszer egész jól megférhet egymás mellett, ha mindent megfelelően konfigurálunk. A Linux alatt elolvashatjuk a Windows alatt készült Word dokumentumot vagy Excel táblázatot is, bár egyelőre még kisebb-nagyobb kompatibilitási problémák felléphetnek.

Köszönet a lelkes felhasználóknak, teszterekneknek, programozóknak azért, hogy a Linux rendszercsalád képességei egyre jobban ösztönzik a komolyabb felhasználókat és az informatikusokat, hogy ezt a rendszert (is) használják.

A szerző

1. A LINUX OPERÁCIÓS RENDSZER

1.1. Történeti áttekintés

1.1.1. Unix gyökerek

A Linux operációs rendszer létjogosultságának megértéséhez mintegy három évtizedet vissza kell tekintenünk a múltba. Akkoriban a számítógépek ház nagyságúak voltak, megbízhatóságuk is hagyott némi kívánnivalót maga után. Az egyik legnagyobb probléma azonban az volt, hogy a különböző számítógépek szinte mindegyikén más és más operációs rendszer futott. A programok általában egy speciális probléma megoldására készültek és egy adott rendszerre írt szoftver nem működött más rendszereken. A fenti okokból kifolyólag további problémák is jelentkeztek. Ezek közül az egyik legfontosabb, hogy ha valaki kitanulta az egyik rendszer használatát, attól egy másik platformon még egyáltalán nem biztos, hogy boldogult. Ezért a komputerek világa akkoriban mind a rendszergazdák, mind a felhasználók számára egy bonyolult és nehezen kezelhető világ volt.

A számítógépek meglehetősen drágák voltak, a felhasználók pedig a vásárlás után sokszor magukra maradtak, mert nem igazán tudták, hogyan működik a készülékük.

Szerte a világon felismerték, hogy a kompatibilitási problémákat megoldva az ebből fakadó egyéb gondok is megszüntethetők. 1969-ben a Bell Laboratóriumban kifejlesztettek egy operációs rendszert, ami a kompatibilitási problémákat hivatott megoldani. A rendszer egyszerű és könnyen kezelhető volt. A korábbiaktól eltérően (Assembly helyett) C programozási nyelven írták és gondoltak a kód továbbfejleszthetőségére is. A Laboratórium fejlesztői a projektet Unix névre keresztelték.

Mivel régebben az operációs rendszerek egy adott gépre íródtak, a kód továbbfejleszthetősége kulcsfontosságú, hiszen a Unix-nak a fenti kódból csak egy igen kis részletre volt szüksége (ezt hívjuk kernelnek). A kernel az egyetlen olyan kódrészlet, melyet az egyes specifikus rendszerekre el kellett készíteni (ez a Unix rendszer alapja). Maga az operációs rendszer és annak minden funkciója erre a kernelre épült. A fejlesztés egy magasszintű programozási

nyelven, a C-n történt. A C nyelvet kifejezetten a Unix rendszer létrehozására készítették. Ezzel a megoldással lényegesen egyszerűbb volt olyan operációs rendszert fejleszteni, amely számos különböző hardveren futott.

A szoftverkereskedők is jól jártak, hiszen többletenergia nélkül is könnyedén a korábbi programok tízszeresét értékesítették. A különféle kereskedők számítógépei a Unix-nak köszönhetően ugyanazon a hálózaton kommunikálhattak, a felhasználók pedig különböző rendszereken dolgozhattak anélkül, hogy a másik rendszert ki kellett volna tanulniuk. A Unix tehát egy nagy előrelépés volt a számítógépes kompatibilitás történetében.

A fenti sikereken felbuzdulva a fejlesztés tovább folytatódott. Egyre több lehetőséggel ruházták fel a rendszert. Egyre nagyobb lett a rendszer támogatottsága is.

A Unix-ot kezdetben kizárólag nagygépes környezetben használták. Ahhoz, hogy valaki „testközelbe" kerülhessen a Unix-szal, egy nagy vállalatnál vagy egy egyetemen kellett dolgoznia.

A technika fejlődésével egyre több házi használatra készített személyi számítógép készült. Az 1980-as évek végére számos háztartásban megjelent a számítógép. Ekkorra már több Unix változat is létezett, de ezek mind drága rendszerek voltak.

1.1.2. A Linux atyja

1991-ben a Helsinki Egyetemen számítástechnikát tanuló Linus Benedict Torvalds gondolt először egy szabadon elérhető egyetemi Unix-változatra. A Unix fejlesztőivel felvette a kapcsolatot.

Linus kezdeti célja az eredeti Unix-hoz hasonló ingyenes rendszer elkészítése volt. A Unix szabványát, a POSIX-ot szintén szem előtt tartotta.

Akkoriban a Plug & Play technológia még nem létezett, de annyian szerettek volna egy saját Unix rendszert, hogy ez nem jelentett akadályt. Többen is bekapcsolódtak a fejlesztésbe. Az összes új hardvereszközhöz új meghajtóprogramokat írtak. Amint megjelent egy új hardver, aki megvásárolta, továbbadta azt a Linux tesztnek. (A rendszert fokozatosan kezdték így nevezni.) A programozók nem elégedtek meg saját hardvereszközeik driver-einek elkészítésével, sőt az sem zavarta őket, hogy a támogatott hardvereszközök listája egyre hosszabb lesz. Tulajdonképpen ezeknek az embereknek köszönhető, hogy a Linux nemcsak a legújabb számítógépeken fut, hanem olyan régi hardvereszközöket is támogat, melyek még akkor jelentek meg a piacon, amikor a Linux adott verziója még nem is létezett.

Két évvel Linus Torvalds levelei után már mintegy 12000 Linux-felhasználó volt. A hobbi-programozók között is népszerű projekt egyre nagyobbra nőtte magát, de mindvégig betartotta a POSIX előírásait.

Az utóbbi években a Unix összes képességét beépítették a Linux-ba, így válhatott mára érett operációs rendszerré. A Linux tulajdonképpen egy teljes Unix-klón, ami a munkaállomásokon és a csúcsteljesítményű szervereken egyaránt futtatható. Egyes változatai tenyérszámítógépeken, mobiltelefonokon, sőt komolyabb karórákon is használhatók. Mára a hardver- és szoftverpiac legfontosabb szereplői saját Linux-fejlesztői gárdával rendelkeznek, így az operációs rendszer támogatottsága tovább nő. Számos új személyi számítógéphez telepített Linux-ot kínálnak.

1.1.3. Alkalmazások Linux-ra

Ma már a személyi számítógépek körében is egyre elterjedtebb a Linux. A fejlesztők kezdetben inkább a hálózatmenedzsmentre és a különféle szolgáltatásokra összpontosítottak, az igényeknek megfelelően azonban idővel elkészültek az irodai programcsomagok is. A szövegszerkesztők, táblázatkezelők, prezentáció-készítők egyre jobb alternatívát kínálnak a Microsoft Office programcsomag termékei mellett.

A fejlesztéséből következik, hogy a Linux a szerver oldalon is kitesz magáért. Stabil és megbízható rendszer. Olyan vállalatnak, mint az online könyvkereskedéssel foglalkozó Amazon, megfelelő adatbázis- és kereskedelmi szolgáltatásokat nyújt. A német hadsereg ugyanúgy fel tudja használni a platformot, mint az amerikai postahivatal. Különösen az internet-szolgáltatók és az internetes szolgáltatást nyújtók tudják kihasználni a Linux lehetőségeit, hiszen itt szükség van tűzfalra, proxy- és webszerverre stb. Még a filmiparban is fel tudják használni az operációs rendszert. A Shrek című film készítésénél például több, Linux-ot futtató számítógépet is felhasználtak. Sok célhardverben is, pl. biztonsági kamera-megfigyelő-rendszerek, stb. tudtunk nélkül is linux operációs rendszer fut a háttérben. Összességében elmondható, hogy a Linux mára a mindennapos használatban is megállja a helyét.

1.2. A felhasználói interfész

1.2.1. Egy bonyolult rendszer?

A kérdésre adott válasz a válaszadótól függ. A tapasztalt UNIX-felhasználók tagadólag válaszolnak, hiszen a Linux a UNIX-ból nőtte ki magát, ideális környezetet nyújtva a haladó felhasználóknak és a programozóknak.

A fejlesztők számára azért ideális a platform, mert mindennel rendelkezik, amire a programozónak csak szüksége lehet, a fordítóktól kezdve a fejlesztési eszközökön át a hibakereső programokig. Még a legegyszerűbb Linux disztribúció is tartalmazza ezeket a szoftvereket. Az ingyenes C-fordító a rendszer része, a hozzá tartozó leírások és dokumentációk nemkülönben.

A Linux első változatainál még viszonylag komoly előismeretekre volt szükség a rendszer használatához. Hiába voltak leírások minden rendszeren, nem mindenki volt olyan türelmes, hogy megkeresse, elolvassa és értelmezze a szakmai kifejezések garmadáját. Sokaknak ezért nem tetszett a Linux. A fejlesztők azonban felismerték, hogy ha jobb eredményeket szeretnének elérni az operációs rendszerek piacán, komoly változtatásokra lesz szükség.

1.2.2. A kevésbé tapasztaltak sem maradnak magukra

Az olyan vállalatok, mint a RedHat, a SuSE vagy a Mandrake, különféle disztribúciókat készítettek tömeges felhasználás céljára. A programok és szolgáltatások menedzselésének megkönnyítésére grafikus felhasználói felületet (GUI) intergáltak a rendszerbe. Persze napjainkban sem árt egy Linux-felhasználónak mélyebb ismeretekkel rendelkeznie, de nem feltétlenül szükséges nagy tudás ahhoz, hogy saját igényeinknek megfelelően állítsuk be a rendszert.

Ha valakinek ilyen igénye van, akár egyetlen karakter gépelése nélkül is bejelentkezhet, grafikus felületen. Ugyanilyen könnyedén futtathat alkalmazásokat is. Emellett azonban elérhetjük a rendszer magját is, ha erre mégis szükség lenne. Ebből a sajátos szerkezetből kifolyólag a kevésbé tapasztalt felhasználók mintegy belenőhetnek a rendszer használatába, hiszen minden elvégezhető karakteres felületen is. A kezdőknek nem kell bonyolult parancsokat megjegyezniük már az első alkalommal, a tapasztalt felhasználónak pedig nem kell ugyanolyan módon elvégeznie egy műveletet, mint ahogy akkor tette, amikor először találkozott a rendszerrel.

A szolgáltatások fejlesztésének folytatásával a rendszer nagy változásokon ment át. Mi döntjük el, hogy saját Linux-unk melyik gyönyörű felületet használja, sőt ha ez sem elég, akár korábbi Windows- vagy Macintosh-felületünk kinézetéhez hasonló felületet is választhatunk. Az utóbbi időben már a 3D-gyorsítás támogatása is beépült a rendszerbe, s napjainkban az oly népszerű USB-eszközök is használhatók Linux-on. A rendszer vagy egyes csomagok egyetlen kattintással frissíthetők. A Linux az összes elérhető szolgáltatást az egyszerű emberek számára is érthető módon logikus listába rendezi. Ha például képernyőkímélőre lenne szükségünk, nem kell mást tennünk, mint felkeresni egy erre szakosodott honlapot (pl. http://kde.org/screenshots/), majd kiválasztani és letölteni a nekünk megfelelő fájlt.

1.3. A rendszer tulajdonságai

1.3.1. Nyílt forráskód

A nyílt forráskód ötlete egyszerű: ha a forráskódokat programozók olvashatják, sőt akár módosíthatják is, a kód fejlődni fog. Felhasználhatjuk a forrás-

kódokat saját céljainkra, hibakeresést hajthatunk végre rajtuk, javíthatjuk az általunk felfedezett hibákat stb. Az így létrehozott szoftver rugalmas és jobb minőségű, mint a zárt keretek között fejlesztett kereskedelmi programok, hiszen sokkal többen tudták tesztelni, jóval változatosabb körülmények között, mint a hagyományos technológiával készült szoftvereket.

Sokáig az egyetlen akadálya a nyílt forráskódú szoftverek elterjedésének az volt, hogy sokan nem láttak benne üzleti hasznot. Mintegy két évtizednek kellett eltelnie ahhoz, hogy elegen ráébredjenek a valóságra – a haszon nem az eladásban, hanem a támogatásban van.

A Linux ma már nem csupán egy operációs rendszer. Egy operációs rendszer létrehozásához teljes infrastrukturális támogatottságot nyújt, a hozzá való programok készítéséhez és teszteléséhez éppúgy. A Linux képes felvenni a versenyt mai gyorsan változó világunk kihívásaival.

1.3.2. Előnyök

A Linux-nak számos előnye van, melyek nagy részét UNIX-gyökereinek köszönheti:

- **ingyenesség:** Ha nem akarunk (vagy nem tudunk) komoly összegeket költeni egy operációs rendszerre, még egy CD árát sem kell rászánnunk. A teljes rendszer ingyenesen letölthető az internetről. Nincs regisztrációs díj, a frissítések pedig szintén ingyenesek, mint ahogy a forráskód is.

  A leggyakoribb licensz a GNU Public License (GPL). Ez azt jelenti, hogy bárki módosíthatja a rendszert, sőt terjesztheti is saját módosított változatát, azzal a kitétellel, hogy a forráskódot is elérhetővé kell tenni.

- **platformfüggetlenség:** Egy olyan kereskedő, aki egy új típusú számítógépet szeretne értékesíteni és nem tudja, milyen operációs rendszer fog futni a gépén, egy Linux kernelt felhasználva el tudja készíteni az adott gépen működő változatot, hiszen az ehhez szükséges dokumentáció ingyenesen rendelkezésre áll.

- **folyamatos futás:** A UNIX-hoz hasonlóan a Linux-ot sem kell állandóan újraindítani. Ebből következik, hogy a rendszer alatt nyugodtan elindíthatunk egy hosszabb műveletet éjszakára vagy automatikusan ütemezhetünk feladatokat, hiszen így nemcsak az információ-feldolgozást, de a hardver kihasználtságát is sokkal kiegyensúlyozottabbá lehet tenni. A Linux tehát felhasználható minden olyan helyen is, ahol nincs állandó felügyelet.

- **biztonság, rugalmasság:** A Linux biztonságpolitikája a UNIX biztonságán alapul. A robosztusság azonban nem pusztán abban nyilvánul meg, hogy védve vagyunk az internetes betörésektől. A fejlesztői gépünk éppoly biztonságos, mint a tűzfalunk. (Ráadásul a kisebb elterjedtsége, az erősza-

kos üzletpolitikát és a gátlástalan haszonszerzést mellőző volta miatt kevesebb támadás is éri.)

- **skálázhatóság:** Teljesen mindegy, hogy egy 2 MB-os memóriával rendelkező tenyérszámítógépünk van vagy egy több exabájtos tárolókapacitással bíró, több száz gépes rendszerünk. A számunkra szükséges részeket telepíthetjük, a felhasználatlanokat eltávolíthatjuk. Nem szükséges nagyteljesítményű számítógép a hatékony munkához. Ha kisebb fejlesztésekkel foglalkozunk, a Linux tökéletesen alkalmazkodik a feladathoz.
- **igen rövid hibakeresési idő:** Mind az operációs rendszer, mind az arra készült alkalmazások hibáit igen gyorsan kijavítják. Több ezer ember dolgozik a fejlesztésen és a tesztelésen, így a hibákat nemcsak megtalálni fogják rövid időn belül, hanem ki is fogják azokat javítani. Sokszor csupán néhány óra szükséges egy hiba megtalálásához és javításához.

1.3.3. Hátrányok

Az előnyök mellett vannak természetesen hátrányok is:

- **túl sok disztribúció:** A rengeteg disztribúció első hallásra ilyesztő vagy képtelen is lehet, csupán nézőpont kérdése. (Persze nem rossz, hogy mindenki megtalálja a neki megfelelő változatot, de ne felejtsük el, ezek kezelése azért mutat némi eltérést.)
  Bármely Linux-felhasználót megkérdezünk, mindenki a saját maga által használt disztribúciót tartja a legjobbnak. Akkor melyiket válasszuk? Ezzel a kérdéssel nem kell túl sokat foglalkoznunk, hiszen az összes változatban az alapvető csomagok többé-kevésbé megegyeznek. Míg a TurboLinux inkább a nagyobb gépeken használatos, addig a RedHat inkább a szerverekhez, a SuSE pedig a munkaállomásokhoz való. A különbségek azonban inkább csak felszínesek. A legjobb megoldás az lenne, ha számos disztribúciót kipróbálnánk, erre azonban sokszor nincs lehetőségünk. Számos tanácsot olvashatunk az interneten vagy szaklapokban a nekünk megfelelő Linux kiválasztásához. Érdemes megnézni a hardver- és szoftver-támogatottságot is. A más-más disztribúciók, bár azonos alapra építkeznek, mégis bizonyos eltéréseket mutató felületet dolgoznak ki, amely egy kevés gyakorlattal rendelkező felhasználónak problémát okozhat.
- **a kezdők számára nem túl jó felhasználó-barátság:** A népszerűségnek köszönhetően egyre többen próbálják könnyebbé tenni a kezdők dolgát, így sokan készítenek és fordítanak dokumentációkat. Sajnos azonban számos dolgot csak a kezdők számára nehezen kezelhető karakteres felületen van lehetőségünk megvalósítani, ahol számos parancs mélyreható ismeretére van szükség. Sok funkció, beállítás nem, vagy csak nagyon bonyolultan végezhető el a grafikus felületen, ami szintén nagy hátrány. Ha

valami nem működik megfelően a linuxos parancsok részletes ismeretének hiányában nem biztos, hogy helyre tudjuk állítani a rendszert. A parancsok pedig még egy DOS-t ismerő felhasználónak is bonyolultnak tűnhetnek. Ráadásul a kezelés tekintetében (kattintás, több asztal, stb.) is vannak eltérések más operációs rendszerekhez képest.

- **lasabb indulás és leállítás:** Egy grafikus felülettel rendelkező rendszer feléledési és a gép kikapcsolhatóságáig való eljutási (leállási) ideje is viszonylag hosszú.
- **körülményesebb lemezhasználat:** Sajnos Linux alatt a CD-ket és egyéb cserélhető háttértárolókat nem lehet a behelyezésüket közvetően azonnal használni (mint a DOS-ban és Windows-ban), hanem először be kell csatolni a fájlrendszerbe. Ez különösen gyakori lemezcsere esetén igen körülményessé teheti a használatot.
- **megbízhatóság:** A nagy kérdés az, hogy hogyan lehet egy ingyenes dolog megbízható? Nem szabad elfelejteni, hogy mi döntjük el, hogy használjuk-e a Linux-ot vagy sem. Egyébként számos teszt bizonyítja, hogy a Linux sok esetben nemhogy ugyanolyan hatékony, mint kereskedelmi társai, hanem még jobb és gyorsabb is a hagyományos megoldásoknál. Ha a Linux nem lenne megbízható, miért lett olyan népszerű, miért használják több millióan világszerte? A felhasználók jelentéseinek köszönhetően a fejlesztők napról napra jobbá teszik a rendszert. A projekt soha nem ér véget. Az örökké változó környezetben a Linux egy olyan projekt, mely az örök tökéletesedésen munkálkodik.

2. AZ ELSŐ LÉPÉSEK

2.1. A bejenetkezéstől a kijelentkezésig

Ahhoz, hogy közvetlenül tudjunk dolgozni egy Linux rendszeren, szükségünk van egy felhasználói névre és egy jelszóra. Mindig be kell jelentkeznünk a rendszerbe.

A korábbiakban már említettük, hogy alapvetően kétféle üzemmódban használhatjuk a rendszert: karakteres módban (a DOS-hoz hasonlít, mintha az egérrel, multitaszkkal és többfelhasználós lehetőséggel lenne kibővítve), illetve grafikus módban (utóbbi szebb, de jóval nagyobb a rendszerigénye).

2.1.1. Karakteres mód

A szöveges üzemmódot egy teljesen fekete képernyőről és (a legtöbb esetben) fehér karakterekről lehet megismerni. A szöveges módban a bejelentkező képernyő általában némi információval szolgál arról a gépről, melyen éppen dolgozunk (például megtudjuk a gép nevét). Végül kapunk egy promptot, mely a bejelentkezésünkre vár, például:

```
RedHat Linux Release 8.0 (Psyche)
blast login: _
```

A karakteres mód bejelentkezése eltér a grafikus bejelentkezéstől (ahol a felhasználói név megadása után az Enter billentyűt kell leütnünk), mivel itt nincs semmilyen gomb a képernyőn, amire az egérrel kattinthatnánk. A jelszót kell megadni, majd még egy Entert kell ütni. A gépelésünkből semmi sem látszik (még a jelszó-bekéréseknél megszokott * sem), a prompt sem mutatja a gépelést. Ennek biztonsági okai vannak.

Ha a rendszer egy érvényes felhasználóként beenged minket, némi információhoz jutunk (ez a nap üzenete), mely bármi lehet. A UNIX rendszereken népszerű egy-egy bölcs gondolat megjelenítése is. Ezután kapunk egy shellt, melyet már ugyanaz a prompt jelez, mint a grafikus módban.

Két jótanács kezdőknek:

- ne jelentkezzünk be rendszergazdaként
  Szöveges módban is igaz, hogy csak akkor érdemes rendszergazdaként bejelentkezni, ha erre feltétlenül szükség van (felhasználók létrehozása,

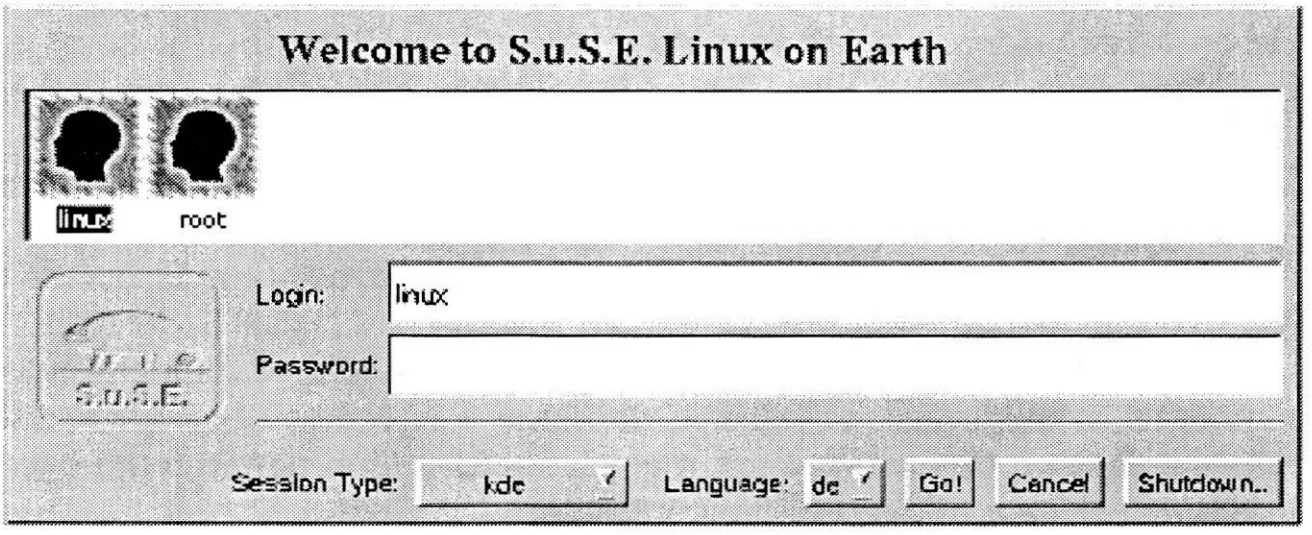
szoftvercsomagok telepítése, rendszerkonfigurálás stb.). Ha végeztünk ilyen jellegű tevékenységünkkel, hagyjuk el ezt a különleges bejelentkezést és nem privilegizált felhasználóként folytassuk a munkát.
A kijelentkezés a logout paranccsal történik. Akkor jelentkeztünk ki sikeresen, ha újra látjuk a bejelentkező ablakot.

- ne kapcsoljuk ki a számítógépet
A kijelentkezés után ne kapcsoljuk ki a számítógépet! Ezt csak a leállítási folyamat után tehetjük meg!

2.1.2. Grafikus mód

Napjainkban rengetegen használnak grafikus felületet. Itt a kapcsolódáshoz először megadjuk a felhasználói nevet, majd egy új ablakban a jelszavunkat. A bejelentkezéshez az egérrel a bejelentkező ablakban OK-t kell nyomni vagy le kell ütni az Enter billentyűt. (Egy nyomógomb egérrel történő megnyomása általában egyenértékű az Enter billentyű hatásával. Grafikus felületen a bal egérgomb többnyire az egérkurzor alatti nyomógombok lenyomására, míg a jobb oldal gomb a helyi menük lekérésére szolgál.)

Sokaknak eleve furcsa lehet, miért is kell minden rendszerindításnál bejelentkezni akkor is, ha csak egyedül használjuk a számítógépet. Az ok pedig egyszerű: az operációs rendszer a felhasználónév–jelszó párossal azonosítja az előtte ülő személyt, hiszen a Linux többfelhasználós rendszer. További magyarázat a bejelentkezés szükségességére, hogy a rendszer így védettebb is. Az általános feladatokhoz egyébként nem szükséges root felhasználóként belépni, ezért célszerű a mindennapi használatra egy egyszerű felhasználót létrehozni és azzal bejelentkezni. Így még véletlenül sem árthatunk a rendszernek.



Általában nem túl jó ötlet grafikusan csatlakozni a root felhasználói névvel (rendszergazdaként), mert a grafika használata számos más programot is futtat, a rendszergazda esetében kiemelt felhasználói jogokkal. Ahhoz, hogy a

lehető legkisebb kockázatot vállaljuk, a grafikus bejelentkezéshez egyszerű felhasználóként jelentkezzünk be.

Miután megadtuk a felhasználónév-jelszó párost, eltelik egy kis idő (számítógépünk sebességétől, az adott szoftvertől és személyes beállításainktól függően), mire a grafikus felület elindul.

A folytatáshoz használjuk a grafikus felületet, vagy nyissunk meg egy terminálablakot (xterm-nek is nevezik). A program az Applications > Utilities, System Tools vagy Internet menüből indítható, az alkalmazott ablakozótól függően. Lehet, hogy valamely ikon is az xterm ablakra hivatkozik. Emellett a háttéren jobb egérgombbal kattintva előhozott helyi menüből is legtöbbször elérhető az alkalmazás.

A menük tallózásakor rengeteg dolgot véghezvihetünk anélkül, hogy egyetlen parancsot is begépelnénk. Ahhoz azonban, hogy igazán ki tudjuk használni a rendszer lehetőségeit, feltétlenül meg kell ismerkednünk a rendszer „szívével". Az egérnél erősebb eszközre van szükség a taszkok kezeléséhez. Ez az eszköz a shell. A grafikus módban a shell egy terminálablak nyitásával aktiválható.

A terminálablak rendszerünk vezérlőpanelének is felfogható. Egy egyszerű, de hatékony karakteres eszköz. Alapból egy promptot mutat, mely megjeleníti a felhasználó bejelentkezési nevét és az aktuális munkakönyvtárat:

```
felhasznalo:~>_
```

A prompt nemcsak így nézhet ki. A másik gyakori forma az alábbi szintaktikával írható le:

```
[felhasznalo@hoszt konyvtar]
```

Ebben a megadásban a felhasznalo a bejelentkezési nevünk, a hoszt az a gép, melyen dolgozunk, a konyvtar pedig aktuális helyünket jelzi a fájlrendszerben.

Bárhogy is néz ki egy prompt, a lényeg, hogy információkat nyújt a rendszerről és nem része az általunk kiadott parancsoknak.

A grafikus módban történő kijelentkezéshez be kell zárnunk az összes terminálablakot és egyéb alkalmazást. Ezután válasszuk a logout ikont vagy keressük meg a Log Out menüpontot. Nem feltétlenül szükséges mindent bezárni és a rendszer is megteheti helyettünk, de ekkor a következő bejelentkezéskor a most aktuálisan futó alkalmazások megjelenhetnek, ami nem mindig egyezik kívánságainkkal (ez a működés konfigurálható).

Ha újra megjelenik a bejelentkező ablak, a kijelentkezés sikeres volt.

Miután megismerkedtünk a rendszerbe történő bejelentkezéssel és az onnan történő kijelentkezéssel, készek vagyunk a legfontosabb parancsok megtanulására.

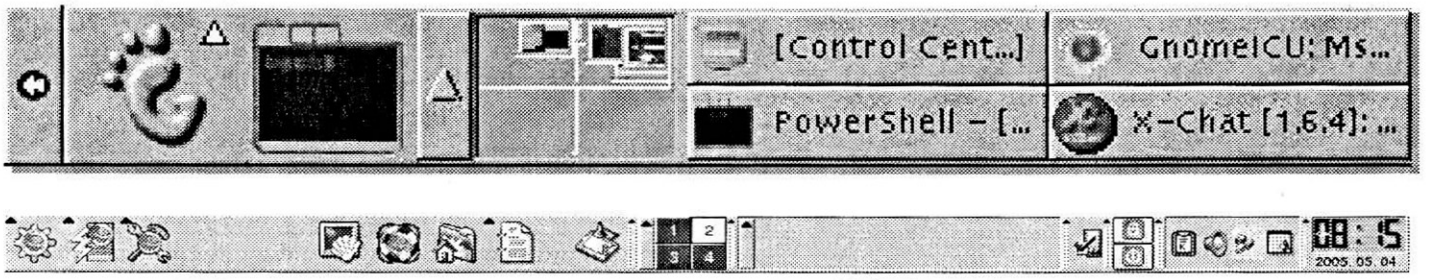
2.1.2.1. A munkaasztal

Linux alá számos ablakozó létezik, ezért az, hogy milyen lehetőségeink vannak, a választott típus is meghatározza (pl. Gnome, Sawfish stb.). A teljes képernyőt lefedő grafikus felületet mint munkaasztalt kezeljük (a Windows operációs rendszerhez hasonlóan). Itt helyezkednek el a leggyakrabban használt rendszerprogramok indító ikonjai, valamint saját hivatkozásokat is elhelyezhetünk igényeinknek megfelelően. Az asztalnak háttérképet is beállíthatunk. Egyes ablakkezelők (pl. Gnome) lehetővé teszik témaválasztók használatát is. Ezek segítségével az asztal megjelenését teljesen meg lehet változtatni.

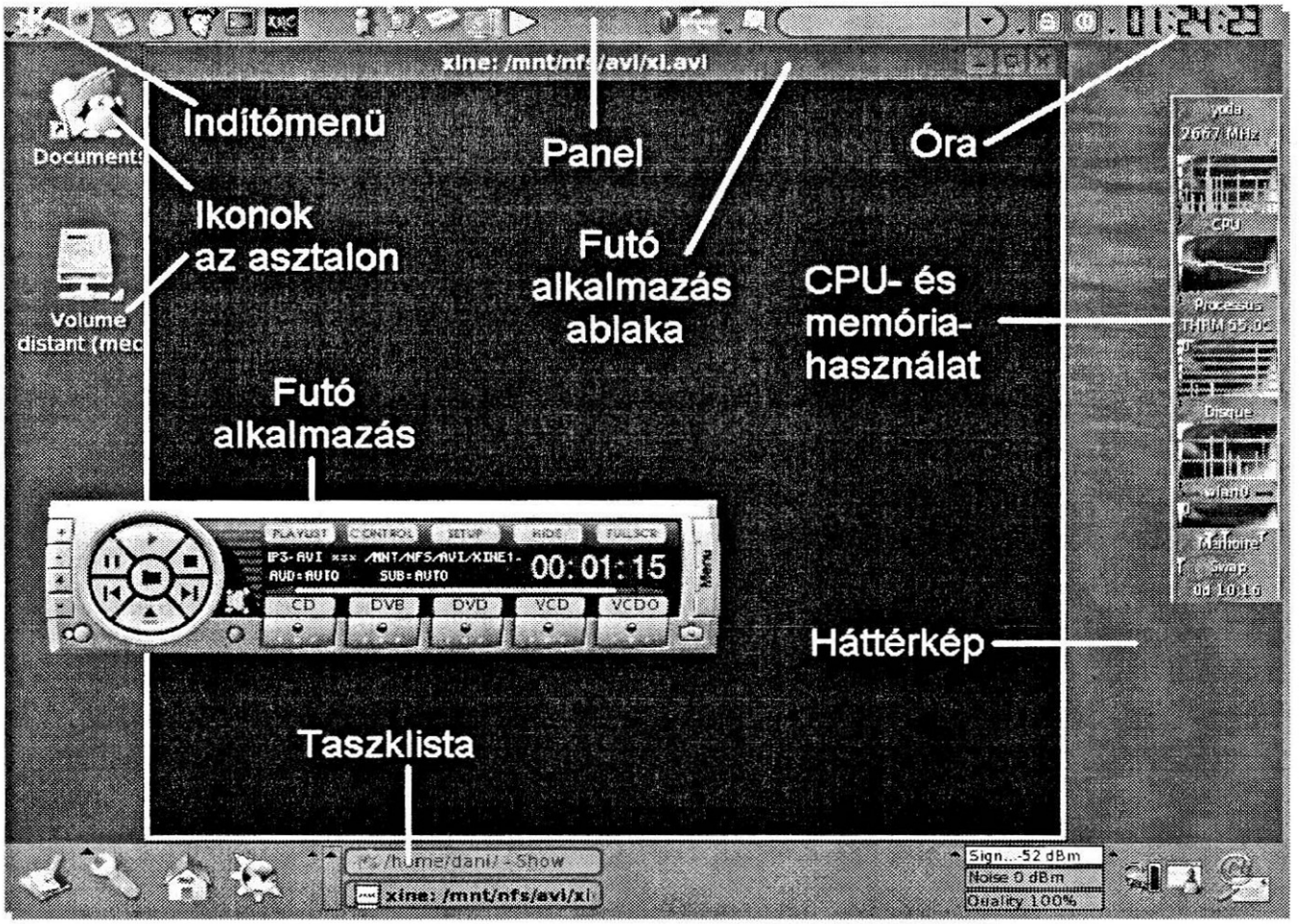
Alapból a képernyő alsó részén foglal helyet az úgynevezett panel (a Windows-os tálca Linux-os megfelelője), de helyzete ettől eltérő is lehet. A képen látható KDE felületnek például két panelje is van: egy a képernyő tetején, egy az alján található. A felső panel bal szélén látható egy K betűs ikon, mely mellett balról egy lefelé mutató háromszög is van. Ezzel nyitható meg a menü (a Windows Start gombjának felel meg). Jobbra további színes ikonok láthatók, melyekkel a felhasználó saját programjait indíthatja. A panel jobb szélén egy digitális óra kapott helyett. Alul látható a futó alkalmazások, taszkok listája. A panelek részeként sokszor futhatnak úgynevezett beépülő kisalkalmazások is, melyek mindig jelen vannak, mégsem foglalnak nagy helyet.

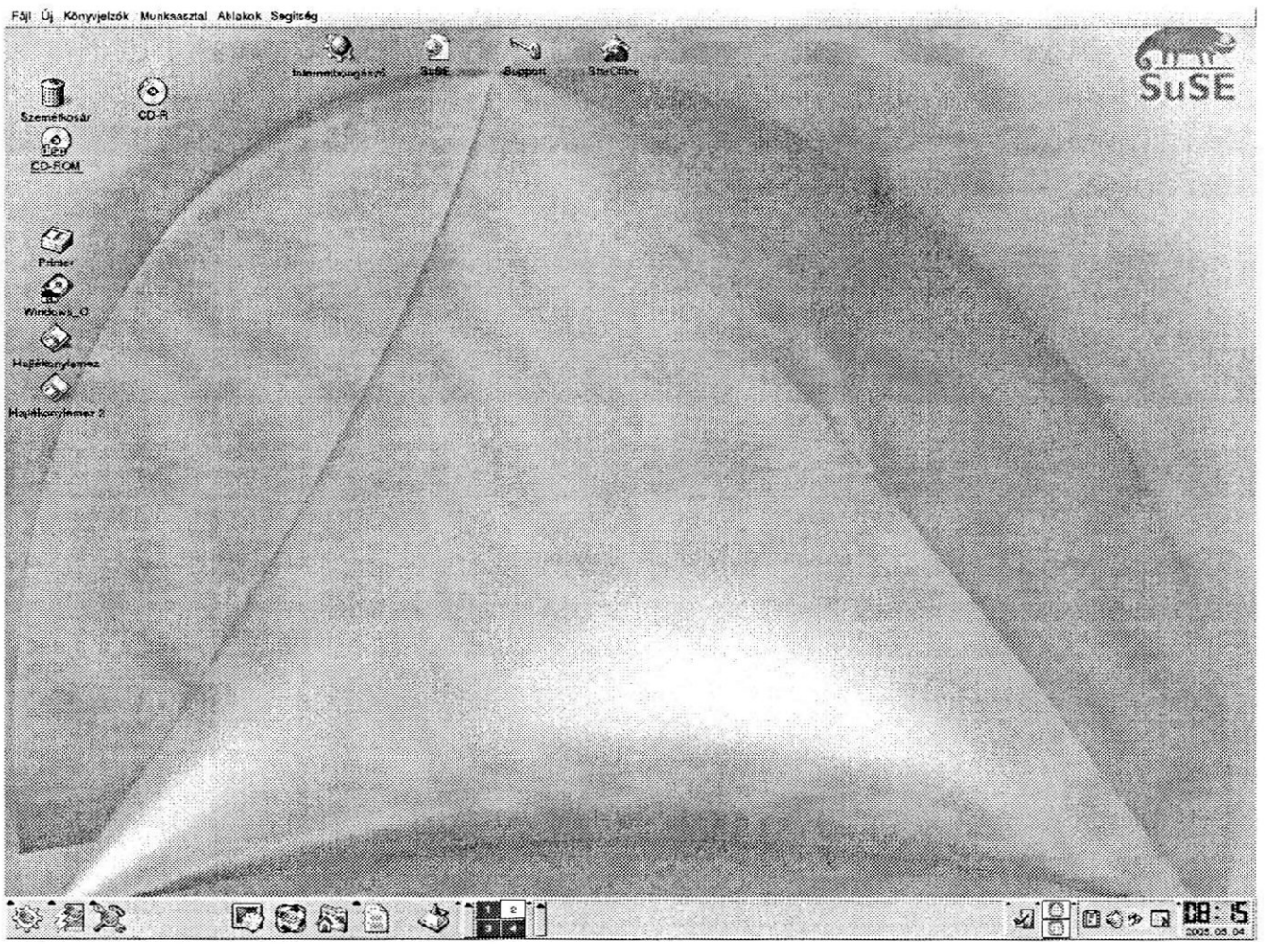
A grafikus felületek sokszor lehetőséget kínálnak a processzor- és memóriahasználat megjelenítésére is, erre látható egy példa a kép jobb szélén. A munkaasztalhoz beállíthatunk képernyővédőt is, ami akkor indul el, ha egy adott ideig nem használjuk a felületet.

Aki egy tappancsos ikont lát az indítómenünél, annak egy GNOME panellel van dolga, de vannak más panelek is:



A munkaasztal Linux alatt tehát maximálisan testreszabható, így nem is nagyon találkozhatunk két teljesen egyforma grafikus felülettel.

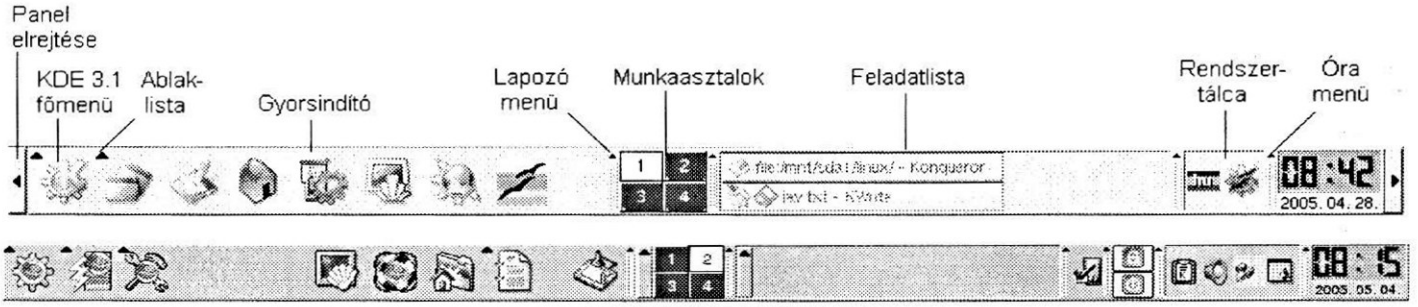

Indítómenü
Panel
Óra
Ikonok az asztalon
Futó alkalmazás ablaka
CPU- és memória-használat
Futó alkalmazás
Háttérkép
Taszklista
xine: /mnt/nfs/avi/xi.avi
Documents
Volume distant (mec
PLAYLIST
CONTROL
SETUP
HIDE
FULLSCR
IP3-AVI
/MNT/NFS/AVI/XINE1-
AUD=AUTO
SUB=AUTO
00:01:15
CD
DVB
DVD
VCD
VCDO
Menu
Disque
Mémoire
Swap
/home/dani/ - Show
xine: /mnt/nfs/avi/xi
Sign...-52 dBm
Noise 0 dBm
Quality 100%


Fájl Új Könyvjelzők Munkaasztal Ablakok Segítség
Szemétkosár
CD-R
CD-ROM
Printer
SuSE

2.2. Az alapok

2.2.1. A grafikus felület kezelése

Számos grafikus felület létezik Linux alá, de bejelentkezés után általában egy két részből álló felülettel állunk szemben. Az egyik rész – ami területileg nagyobb – a munkaasztalunk, a másik a képernyő alján (vagy szélén, esetleg tetején, beállítástól függően) elhelyezkedő panel („tálca"). Az asztalon a legfontosabb és leggyakrabban használatos rendszerikonok és saját ikonok helyezkednek el. A panel Linux alatt számos részből áll, a következőkben a KDE 3.1 felület paneljét vesszük alaposabban szemügyre:

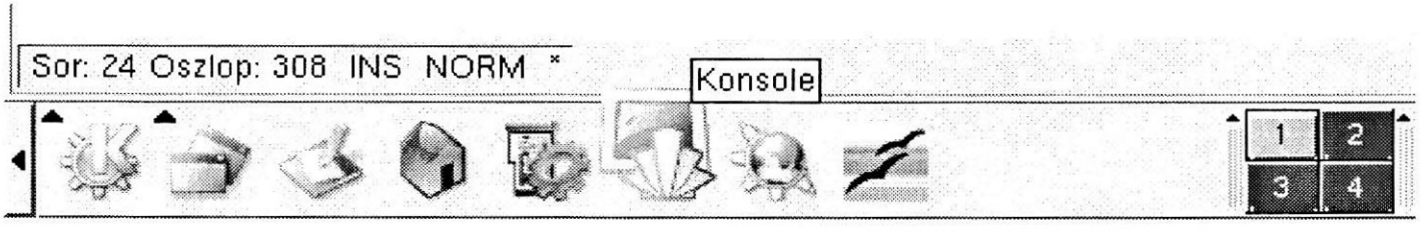


A tálca bal szélén található balra mutató nyílra kattintva a tálca a képernyő bal szélére kicsúszik, így láthatóvá válik az alatta levő terület is. A nyíl ekkor jobbra mutatóvá válik, erre kattintva visszakapjuk az eredeti tálcát. Ezt hívjuk a panel elrejtésének.

A következő nyíl az alkalmazások indítását könnyíti meg, ami felfelé mutat. Ez felel meg a Windows alatt megszokott Start menünek. A KDE főmenüjét nevezik K menünek.

Ettől jobbra a következő felfelé mutató nyíl az ablaklistát jeleníti meg. Az ablakokat egyenletesen és lépcsőzetesen rendezhetjük és villámgyorsan válthatunk is közöttük.

A fenti három menütől jobbra néhány ikon található, ami a Windows gyorsindítójának felel meg (A munkaasztal megjelenítése, Saját könyvtár, Vezérlőközpont, Konsole, Konqueror webböngésző, OpenOffice.org). Az az ikon, ami felett aktuálisan tartózkodik az egérmutató, megnagyobbodik és megjelenik a súgócímkéje:

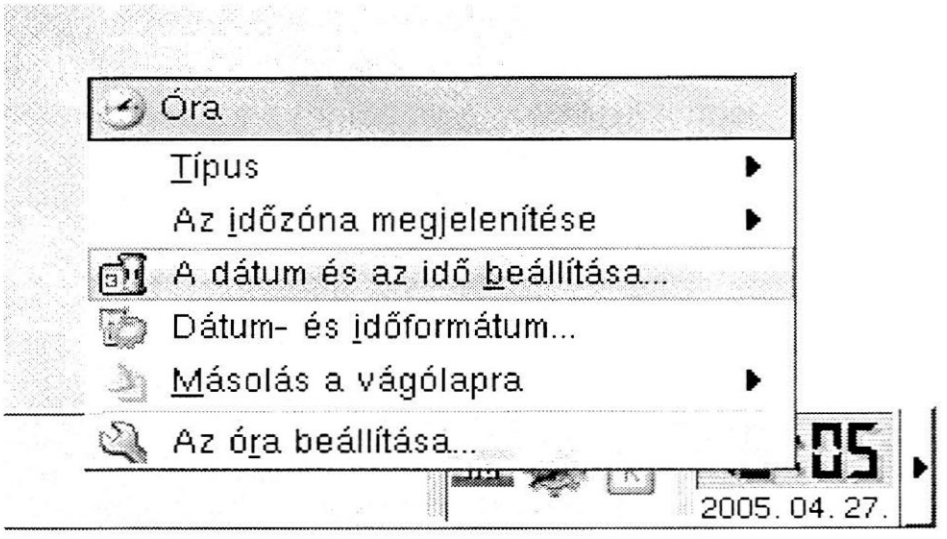


A következő felfelé mutató nyíl az úgynevezett Lapozó menüt jeleníti meg. Itt mozgatható, illetve eltávolítható a Lapozó.

Emellett található 2x2-es elrendezésben a négy munkaasztal (Desktop 1-4) szimbóluma. A munkaasztalok közt ezeken a szimbólumokon történő kattintással tudunk váltani.

A panel következő része a Feladatlista menü, melyben a Feladatlista mozgatható, eltávolítható és jellemzői állíthatók.

E rész mellett vannak a futó alkalmazások reprezentációi. A Windows operációs rendszerben is megszokott, szokásos módon egyszer kattintva egy alkalmazás címére az előtérbe ugrik, még egyszer rákattintva kis méretűvé válik. A Windows-tól eltérően azonban az aktív alkalmazás reprezentációja domború, s a többi a homorú.



A következő menü a Rendszertálca menü, ahol a billentyűzetkiosztás és hasonló fontos dolgok kapnak helyet (pl. az OpenOffice.org QuickStarter).

A panel utolsó menüje az Óra menü a képernyő jobb szélén. A dátum és idő beállítása mellett többféle óratípusból is választhatunk. Beállíthatjuk a dátum- és időkijelzés formátumát, az időzónát, sőt akár vágólapra is másolhatjuk az aktuális dátumot és időt, különféle formátumokban.

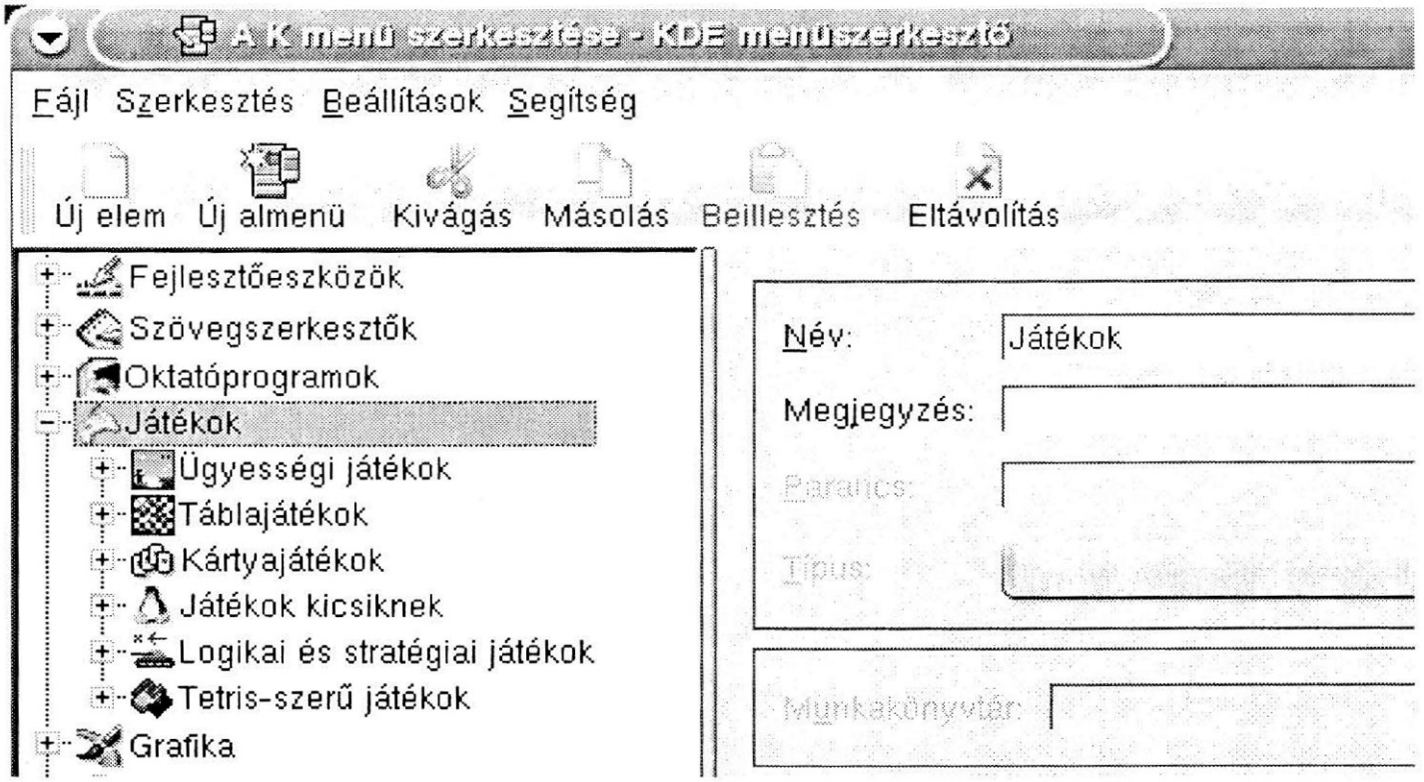
2.2.1.1. A menü testreszabása

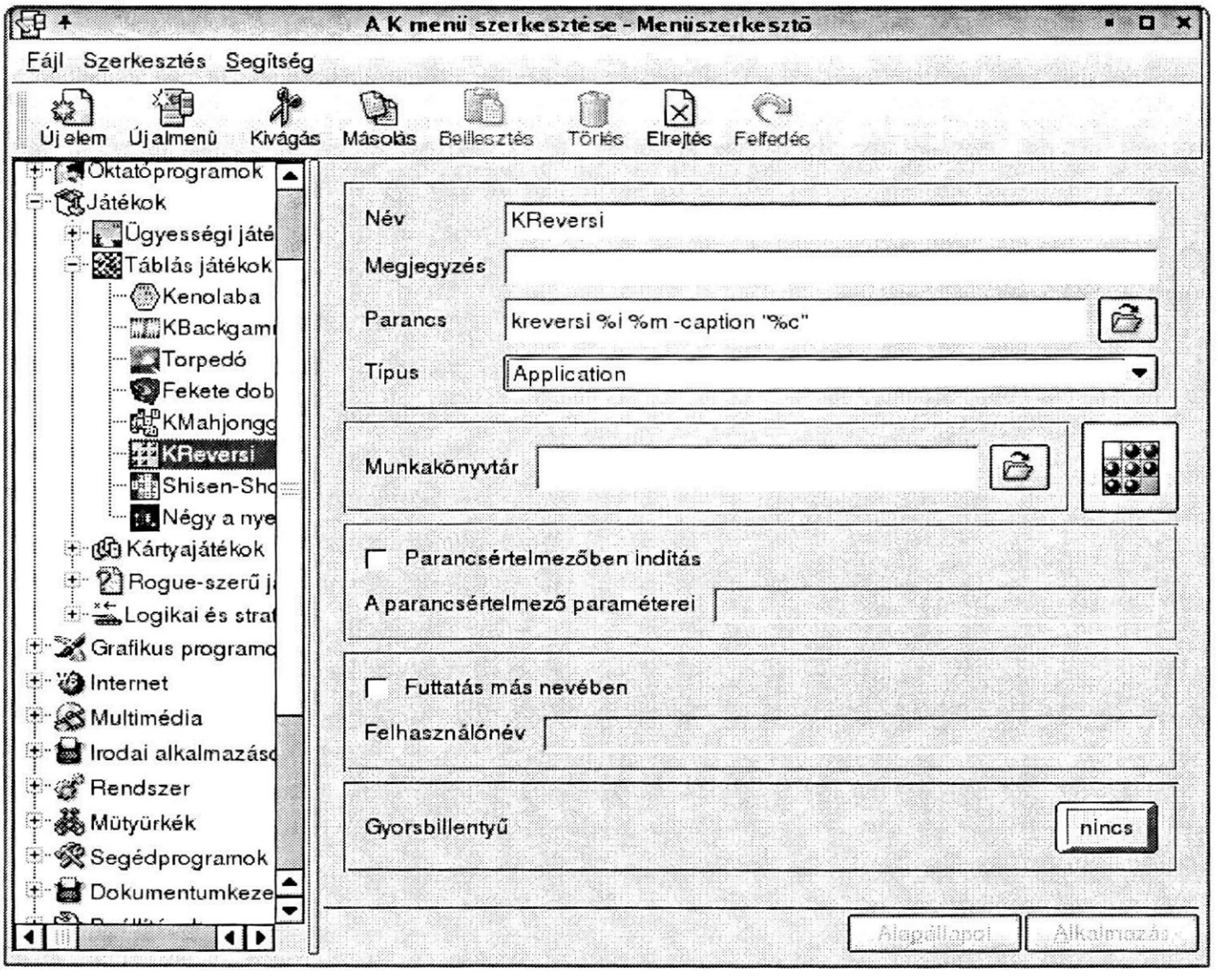
A KDE felülethez létezik külön menüszerkesztő, mellyel a K menüt maximálisan testreszabhatjuk. A program indítása a főmenü **Rendszer** menüjének **Beállítások** menüjében található **Menüszerkesztő** kiválasztásával történik.

A szerkesztővel új menüket, almenüket szúrhatunk be, meglévőket törölhetünk, áthelyezhetünk, átnevezhetünk stb.

A szerkesztést a fastruktúrájú megjelenítés teszi még könnyebbé.

A Windows faszerkezeteihez hasonlóan itt is a + jelekre történő kattintással bonthatunk ki egy-egy ágat, ekkor a jel - jellé változik. A faszerkezet egyes részeinek megnevezése előtt még az adott alkalmazás ikonja is megjelenik, tovább segítve menüszerkesztési munkálatainkat.

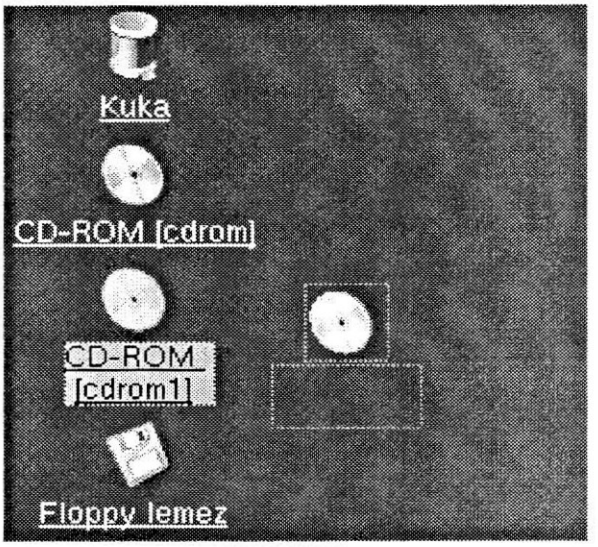

A K menü szerkesztése - KDE menüszerkesztő
Fájl Szerkesztés Beállítások Segítség
Új elem
Új almenü
Kivágás
Másolás
Beillesztés
Eltávolítás
Fejlesztőeszközök
Szövegszerkesztők
Oktatóprogramok
Játékok
Ügyességi játékok
Táblajátékok
Kártyajátékok
Játékok kicsiknek
Logikai és stratégiai játékok
Tetris-szerű játékok
Grafika
Név:
Játékok
Megjegyzés:
Parancs:
Típus:
Munkakönyvtár:


A K menü szerkesztése - Menüszerkesztő
Fájl Szerkesztés Segítség
Új elem
Új almenü
Kivágás
Másolás
Beillesztés
Törlés
Elrejtés
Felfedés
Oktatóprogramok
Játékok
Ügyességi játé
Táblás játékok
Kenolaba
KBackgam
Torpedó
Fekete dob
KMahjongg
KReversi
Shisen-Sho
Négy a nye
Kártyajátékok
Rogue-szerű j
Logikai és strat
Grafikus programo
Internet
Multimédia
Irodai alkalmazás
Rendszer
Mütyürkék
Segédprogramok
Dokumentumkeze
Név
KReversi
Megjegyzés
Parancs
kreversi %i %m -caption "%c"
Típus
Application
Munkakönyvtár
Parancsértelmezőben indítás
A parancsértelmező paraméterei
Futtatás más nevében
Felhasználónév
Gyorsbillentyű
nincs
Alapállapot
Alkalmazás

2.2.1.2. Műveletek az asztalon

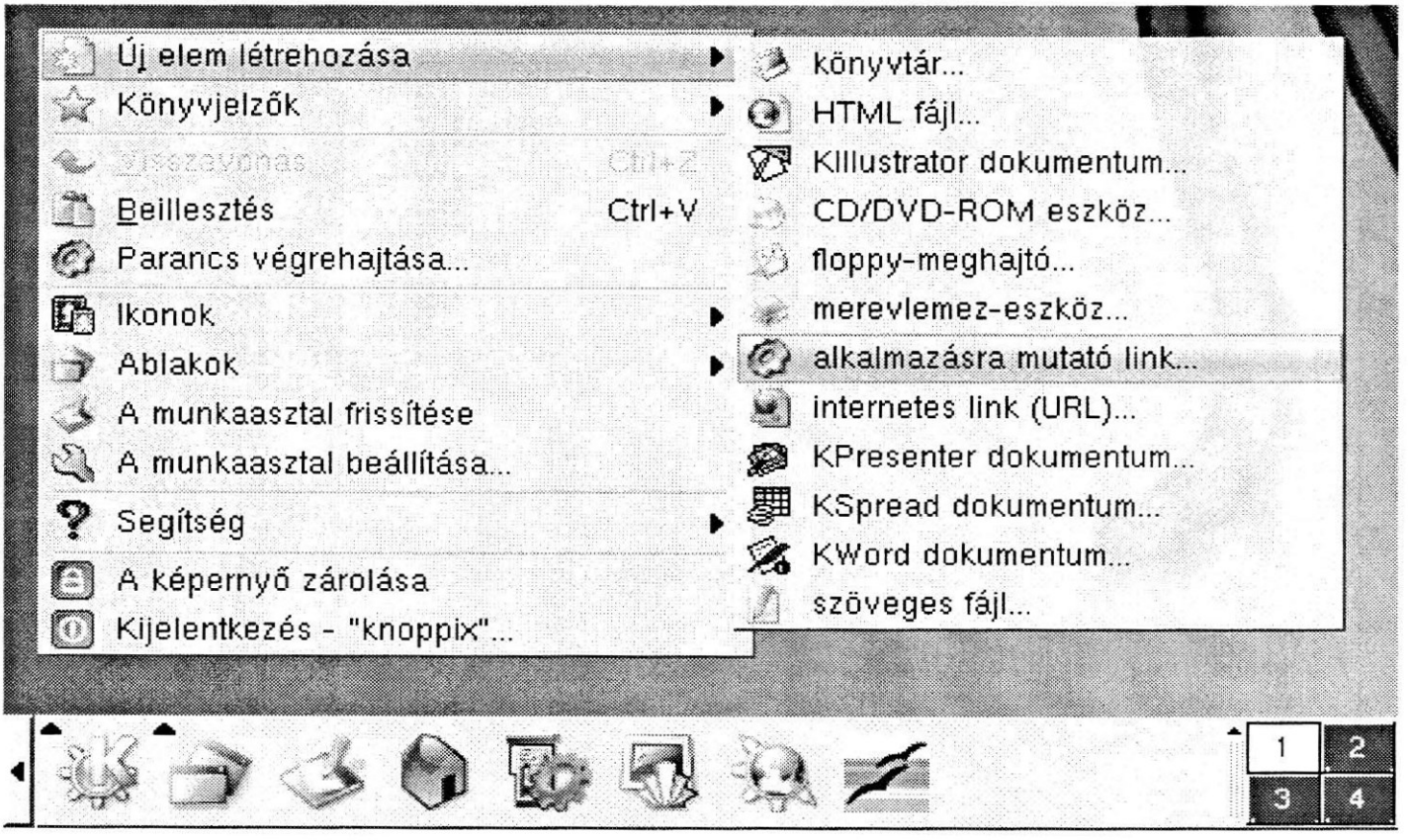
Az asztalikonok egyetlen kattintással választhatók ki, mint internetes böngészéseink során a hivatkozások. Az ikonok mozgatása úgy történik, hogy a bal egérgombot lenyomva tartva az egeret mozgatjuk, majd a kívánt helyen a gombot felengedjük.

A mozgatási művelet alatt az ikon és aláírása az eredeti helyen kijelölten látható, a mozgatás aktuális helyén pedig csak az ikon látható, vékony szaggatott vonalas kijelöléssel.



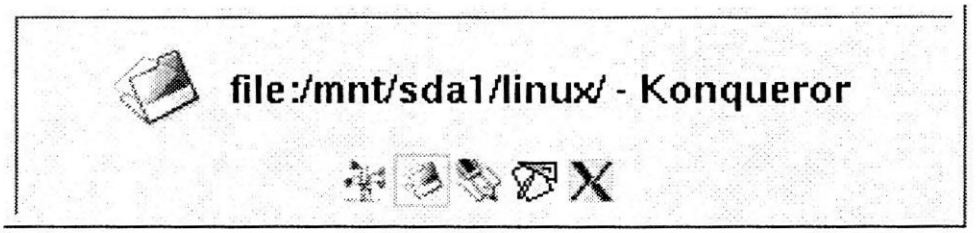
Ha egy futtatni kívánt alkalmazás ikonja el van helyezve az Asztalon, onnan egyetlen kattintással indítható. A könyvtárak megnyitása hasonlóan történik.

Ha új hivatkozást szeretnénk elhelyezni az asztalon, a jobb egérgombbal hívjuk elő a helyi menüt, majd válasszuk ki az Új elem létrehozása menü alkalmazásra mutató link... menüpontját:



2.2.1.3. Váltás az ablakok között

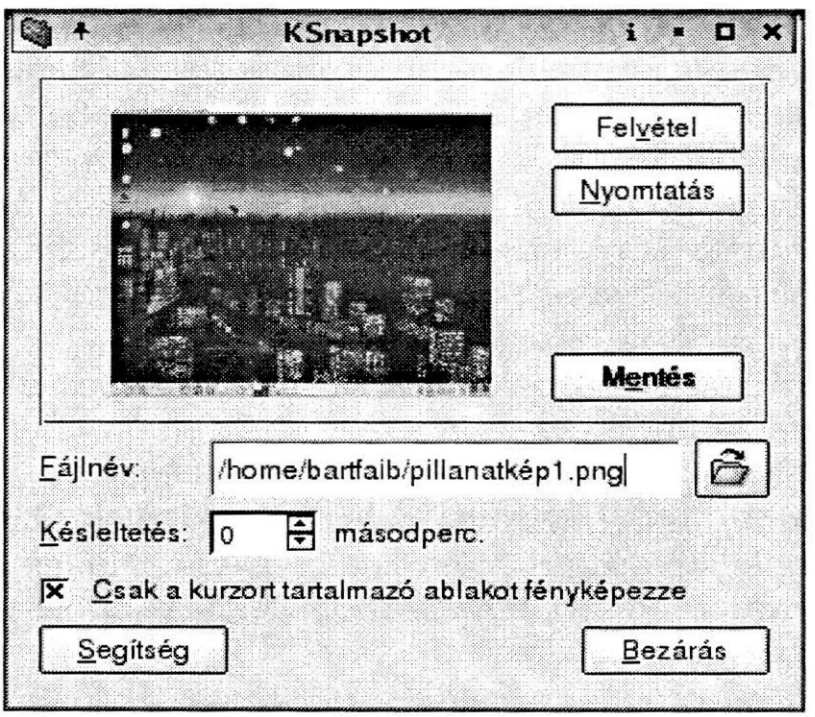
A Windows-hoz hasonlóan több lehetőségünk van a megnyitott alkalmazások ablakai közötti váltásra. Az egyik lehetőség a panel Feladatlista menüjének használata, ahol a kívánt alkalmazás címére kell egyet kattintani, s máris előtérbe kerül annak ablaka. Alkalmazhatjuk az úgynevezett gyors Alt+Tab billentyűkombinációt is, ahol az Alt gombot lenyomtva tartva megjelenik egy menü a futó alkalmazások ikonjaival és az aktuális alkalmazás megnevezésével. Emellett a Tab billentyűt nyomkodva lépegethetünk az alkalmazások között. Amelyik ikonja felett elengedjük a billentyűket, az kerül előtérbe.



2.2.1.4. Képernyőképek mentése

Míg Windows alatt a jól ismert Print Screen billentyű megnyomásával készíthetünk képernyőképeket, addig Linux alatt külön programot kell használni erre a célra. Ilyen úgynevezett képlopó szoftver például a KSnapshot, amit a K menü Grafika menüjéből indíthatunk.

A program ablak-képek és teljes képernyőképek készítésére egyaránt alkalmas. Helyi menüket, gyorsmenüket, mozgásokat stb. időzítés segítségével rögzíthetünk a képen. Azt, hogy milyen képformátumba (PNG, JPG, Windows Bitmap stb.) történjen a mentés, mi döntjük el.



2.2.2. A parancsok

Az alábbiakban megismerkedünk a legfontosabb Linux parancsokkal:

Parancs	Jelentés
ls	az aktuális munkakönyvtár fájljainak és könyvtárainak kilistázása (olyan, mint a DOS-ban a dir)
cd *konyvtar*	könyvtárváltás
passwd	az aktuális felhasználó jelszavának módosítása
file *fajlnev*	a *fajlnev* nevű fájl típusával megegyező típusú fájlok megjelenítése
cat *szovegfajl*	a *szovegfajl* tartalmának megtekintése
pwd	aktuális munkakönyvtár kiírása
exit vagy logout	kijelentkezés
man *parancs*	a *parancs* felhasználói dokumentumának megjelenítése
info *parancs*	a *parancs* információs oldalainak megjelenítése

2.2.3. Általános tudnivalók

A parancsokat a prompt után kell begépelni karakteres módban vagy a grafikus üzemmód terminálablakában, majd az Enter billentyűt kell leütni.

A parancsok önmagukban is kiadhatók (pl. ls), de opciók meghatározásával módosíthatjuk is a parancsok jelentését. Az opciókat általában kötőjellel (-) adjuk a parancshoz (pl. ls -a). Egy opcióként megadott karakter jelentése parancsonként eltérhet. A GNU programok hosszú opciókkal egészíthetők ki. Ezeket két kötőjel (--) előzi meg (pl. ls --all). Nem minden parancsnak vannak opciói.

Egy parancs argumentuma(i) azon objektumo(ka)t határozza/határozzák meg, mely(ek)re a parancs hatását érvényesíteni szeretnénk. Így például az ls /etc parancsban az /etc könyvtár az ls parancs argumentuma. Ez jelzi, hogy ennek a könyvtárnak a tartalmát szeretnénk megtekinteni az alapértelmezett helyett (ami az aktuális könyvtár), melynek megtekintéséhez az ls parancsot önmagában elég kiadni. Egyes parancsok argumentumai kötelezőek, mások argumentumai opcionálisak, elhagyhatók.

Linux-ban (a UNIX-hoz hasonlóan) a könyvtárakat per jelekkel (/) határoljuk (mint az URL-ekben is).

A . és .. szimbólumoknak különleges jelentésük van.

Ha nem szükséges, ne rendszergazdaként dolgozzunk. Általános munkánk során nem kellenek különleges felhasználói jogok. Ha egy parancs végrehajtásához vagy egy program futtatásához kiemelt jogokra van szükség, a rendszer úgyis figyelmeztet minket.

Különösen az első próbálkozásaink során fontos a dokumentuációk tanulmányozása, mert így lehet igazán elsajátítani a rendszer működését.

2.2.4. Gyorsbillentyűk a Bash használatakor

Számos billentyűkombináció áll a rendelkezésünkre a dolgunk megkönnyítéséhez és meggyorsításához a GNU shell-lel, a Bash-sel (majdnem minden Linux rendszeren ez az alapértelmezett shell).

Billentyű / kombináció	Funkció
Ctrl A	Kurzormozgatás a parancssor elejére.
Ctrl C	A futó program befejezése és a prompt visszaadása.
Ctrl D	Kijelentkezés az aktuális shell-ből, egyenértékű az exit vagy a logout begépelésével.
Ctrl E	Kurzormozgatás a parancssor végére.
Ctrl H	Backspace karakter generálása.
Ctrl L	A terminál törlése.
Ctrl R	Keresés a már kiadott parancsok között.
Ctrl Z	Egy program felfüggesztése.
Ctrl → Ctrl ←	Karakterenként lépkedhetünk a parancssorban balra vagy jobbra.
↑ ↓	A kiadott parancsok tallózása, ami különösen akkor hasznos, ha egy hosszabb parancsot kisebb módosításokkal újra ki szeretnénk adni.
Shift Page Up Shift Page Down	A terminál-puffer böngészése (azon szövegek megtekintése, melyek már felfelé kigördültek a képernyőről).
Tab	Parancs vagy fájlnév kiegészítése. Ha több lehetőség is van, a rendszer jelzi ezt, ha túl sok van, felkínálja, hogy megnézzük az összeset. Ha például van egy nagyon_hosszu_nevu_konyvtar nevű könyvtárunk, nem kell a parancsoknál a teljes nevet begépelnünk. A cd nagy parancs begépelése után ütünk egy Tab billentyűt, így a shell kiegészíti a könyvtárnevet, ha nincs több, nagy kezdetű könyvtárnevünk. Ha még csak n betűvel kezdődő könyvtárunk sincs, elég a cd n és a Tab alkalmazása.
Tab Tab	Megmutatja a parancs- vagy fájlnév-kiegészítés lehetőségeit.

2.3. Segítség

2.3.1. A kézikönyv – a man oldalak

Sok kezdő felhasználó azért vonakodik a man (a manual, azaz kézikönyv rövidítéséből) oldalak használatától, mert azok megjelenése kissé bizarr. Valójában egy nagyon jól szervezett segítségnyújtási formáról van szó. Ehhez nézzük meg a man parancs saját online-kézikönyvét!

man *man*

Az Enter leütése után megkapjuk a man parancs dokumentuációját:

```
man(1)                                                    man(1)

NAME
  man - format and display the on-line manual pages
  manpath - determine user's search path for man pages

SYNOPSIS
  man [-acdfFhkKtwW] [--path] [-m system] [-p string] [-C config_file]
  [-M pathlist] [-P pager] [-S section_list] [section] name ...

DESCRIPTION
  man formats and displays the on-line manual pages. If you specify
  section, man only looks in that section of the manual.
  name is normally the name of the manual page, which is typically the
  name of a command, function, or file. However, if name contains a
  slash (/) then man interprets it as a file specification, so that you
  can do man ./foo.5 or even man /cd/foo/bar.1.gz.
  See below for a description of where man looks for the manual
  page files.

OPTIONS
  -C config_file
lines 1-27
```

A következő oldal megtekintéséhez a szóköz billentyűt kell leütnünk. Az előző oldalra a b billentyűvel tudunk visszalépni. Ha végignéztük a dokumentumot, a man kilép és visszakapjuk a promptot. Ha korábban (a dokumentum közepén) szeretnénk kilépni, a q billentyűt kell leütnünk.
Egy man oldal – a fenti példából is láthatóan – több, jól elhatárolt részből áll:

- Az első sor tartalmazza annak a parancsnak a nevét, melyről éppen olvasunk és annak a résznek az azonosítóját, ahol ez az oldal van. A man oldalak fejezetekbe vannak rendezve. A parancsoknak sokszor többszörös man oldalai vannak, így például lehet felhasználói man oldal, rendszergazdai man oldal, fejlesztői man oldal.

- A parancs neve és rövid leírása olvasható, ezekből építkeznek a man oldalak indexei.
- A parancs szintaktikájából megtudhatjuk, hogyan lehet megadni az opciókat és/vagy argumentumokat. Az opciókat úgy foghatjuk fel, mint a parancs futtatásának egy módját. Az argumentum pedig azt határozza meg, mire vonatkozik a parancs. Egyes parancsoknak nincsenek opciói és argumentumai. Az elhagyható opciókat és argumentumokat a [és] jelek jelzik.
- A fenti részeket egy hosszabb leírás követi.
- Az opciók listája és azok leírása is olvasható. Az opciók gyakran kombinálhatók is. Ha nem, arról is értesülünk ebből a részből.
- Megtudhatjuk, mely shell változók befolyásolják a parancs működését (nem minden parancsnál van ilyen).
- Más man oldalakra való hivatkozásokat is találhatunk. Zárójelben van annak a man oldal-résznek a száma, ahol megtalálható ez a parancs. A gyakorlottabbak a SEE ALSO (lásd még) részre a SEE keresőstring utáni / paranccsal és az Enter lenyomásával ugranak.
- Néha találkozhatunk az ismert hibákkal is. Ekkor meg szokták adni azt is, hol lehet értesíteni a fejlesztőket, ha mi magunk találunk új hibákat.
- A fentieken túl szerzői és jogi információk is lehetnek a man oldalakon.

Már volt róla szó, hogy egyes parancsoknak többszörös man oldaluk van. A passwd parancsnak például az 1-es és az 5-ös szekcióban is van egy man oldala. Alapesetben a kisebb számú oldal jelenik meg. Ha másik oldalra van szükségünk, adjuk meg annak számát a man parancs után:

```
man 5 passwd
```

Ha egy parancs összes man oldalát látni szeretnénk egymás után, az -a opciót használjuk:

```
man -a passwd
```

Ekkor az első man oldal végére érve és a szóközt leütve a következő szekció man oldala jelenik meg.

2.3.2. Az info oldalak

A man oldalak mellett egy parancsnak vannak info oldalai is, melyeket az info paranccsal jeleníthetünk meg. Ezek általában kicsit több információt tartalmaznak és valamivel könnyebb a használatuk. Egyes parancsok man oldalai az info oldalakra hivatkoznak.

Nézzük meg az info parancs info oldalát:

File: info.info, Node: Top, Next: Getting Started, Up: (dir)

Info: An Introduction
****

Info is a program, which you are using now, for reading documentation of computer programs. The GNU Project distributes most of its on-line manuals in the Info format, so you need a program called "Info reader" to read the manuals. One of such programs you are using now.

If you are new to Info and want to learn how to use it, type the command `h' now. It brings you to a programmed instruction sequence.

To learn advanced Info commands, type `n' twice. This brings you to `Info for Experts', skipping over the `Getting Started' chapter.

* Menu:

* Getting Started:: Getting started using an Info reader.
* Advanced Info:: Advanced commands within Info.
* Creating an Info File:: How to make your own Info file.
--zz-Info: (info.info.gz)Top, 24 lines --Top----------
Welcome to Info version 4.2. Type C-h for help, m for menu item.

A nyílbillentyűkkel böngészhetünk a szövegben és mozgassuk a kurzort egy * karakterrel kezdődő sorra, mely azt a kulcsszót tartalmazza, melyről információt szeretnénk, majd üssünk Enter-t. A P, illetve az N billentyűvel léphetünk az előző, illetve a következő tárgyra. A szóközzel lapozhatunk egy oldalt, függetlenül attól, hogy ott egy új téma kezdődik vagy egy másik parancs info-oldala. A Q billentyűvel léphetünk ki.

2.3.3. A whatis és az apropos parancsok

A whatis (miez) paranccsal egy parancs rövid leírását kérhetjük le. Például:

[promptunk] whatis *ls*
ls (l) - list directory contents

A rövid leírás mellett a megfelelő oldalt tartalmazó man oldalak gyűjteményében az első részt láthatjuk.

Ha nem tudjuk eldönteni, hol kezdjük és melyik man oldalt olvassuk el, az apropos parancs több információval szolgál. Így ha nem tudjuk, hogyan kell elindítani egy böngészőt (browser-t), az alábbi parancsot adjuk ki:

promptunk> apropos *browser*

```
Galeon [galeon](1) - gecko-based GNOME web browser
lynx           (1) - a general purpose distributed information browser
                     for the World Wide Web
ncftp          (1) - Browser program for the File Transfer Protocol
opera          (1) - a graphical web browser
pilot          (1) - simple file system browser in the style of the
                     Pine Composer
pinfo          (1) - curses based lynx-style info browser
pinfo [pman]   (1) - curses based lynx-style info browser
viewres        (1x) - graphical class browser for Xt
```

Az Enter leütése után kiderül, hogy gépünkön számos böngésző van. Ezek nemcsak webböngészők, hanem vannak közöttük fájl-, FTP- és dokumentációböngészők is. Ha fejlesztői csomagokat telepítettünk, a böngészőkkel kapcsolatos programírásokra vonatkozó man oldalakkal is rendelkezhetünk.

Általánosságban elmondható, hogy az 1. szekció man oldalai – melyek az (1) jelöléssel vannak ellátva – egyszerű felhasználóként is kipróbálhatók. A felhasználó az apropos parancs kiadása után következetesen a galeon, a lynx vagy az opera programot fogja elindítani, hiszen ezek egyértelműen webböngészők (az angolul nem tudók helyzete persze nem ilyen rózsás).

2.3.4. A --help opció

A legtöbb GNU parancs támogatja a --help opciót, ami egy rövid leírást nyújt a parancs használatáról, illetve a használható opciókról. Példaként nézzük meg a cat parancsot a --help opcióval:

```
promptunk@hoszt: cat --help
Usage: cat [OPTION] [FILE]...
Concatenate FILE(s), or standard input, to standard output.

  -A, --show-all           equivalent to -vET
  -b, --number-nonblank    number nonblank output lines
  -e                       equivalent to -vE
  -E, --show-ends          display $ at end of each line
  -n, --number             number all output lines
  -s, --squeeze-blank      never more than one single blank line
  -t                       equivalent to -vT
  -T, --show-tabs          display TAB characters as ^I
  -u                       (ignored)
  -v, --show-nonprinting   use ^ and M- notation,
                           except for LFD and TAB
      --help               display this help and exit
      --version            output version information and exit

With no FILE, or when FILE is -, read standard input.

Report bugs to <bug-textutils@gnu.org>.
```

2.3.5. Grafikus segítség

Grafikus felületet választva sem kell magunkra maradnunk. Az alapértelmezett KDE fájlmenedzser, a Konqueror a man és info oldalakat egyszerűen és színesen jeleníti meg. A Location címsorban próbáljuk ki az info:info hatását! Meglepő módon az info parancs böngészhető info oldalát kapjuk. Ennek analógiájára a man:ls az ls parancs man oldalát hozza be.

A parancsneveket ki is egészíttethetjük. A fenti esetben az összes ls-sel kezdődő parancs man oldalát egy legördülő menüben láthatjuk. Az info:/dir címsorba történő begépelésével az összes info oldalt megtekinthetjük, típuskategóriákba csoportosítva.

A Gnome Help Browser (súgó böngésző) nagyon felhasználóbarát. A Gnome menüből az Applications > Help kiválasztásával, az asztal mentőöv-ikonjával vagy a gnome-help parancs terminálablakba történő begépelésével egyaránt indítható. A rendszer dokumentációja és a man oldalak egyszerűen böngészhetők.

A nautilus fájlmenedzserben a man és info oldalakat keresőlinkekkel is elérhetjük. Az egyszerű böngészhetőség mellett az oldalakat hivatkozások kapcsolják össze. A nautilus a parancssorból, a home könyvtár ikonjára történő kattintással vagy a Gnome menüből indítható.

2.3.6. Kivételek

Egyes parancsoknak nincs külön dokumentációja, mivel azok egy másik parancs részei. Ilyenek a cd, az exit, a logout, a pwd. Ezek a shell részei, ezért nevezzük őket beépített parancsoknak is. Ha ezekről a parancsokról szeretnénk információkat lekérdezni, a shell-ünk man vagy info oldalaira van szükség.

Ha eredeti rendszerkonfigurációnkat módosítottuk, előfordulhat, hogy a man oldalak ugyan léteznek, de nem láthatóak, mivel a shell környezet megváltozott. Ebben az esetben érdemes megvizsgálni a MANPATH változót.

Egyes programoknak vagy csomagoknak csak egy utasítás-halmazuk van, esetleg az usr/share/doc könyvtárra hivatkoznak.

A legrosszabb esetben véletlenszerűen eltávolítottuk a dokumentációt a rendszerből. Ekkor használjunk valamilyen keresőprogramot, ha pedig így sincs eredmény, akkor újra kell telepítenünk azt a csomagot, mely tartalmazza a dokumentáció által igényelt parancsot.

3. FÁJLRENDSZEREK, FÁJLKEZELÉS

Miután megismertük a Linux legalapvetőbb parancsait, célszerű részletesen foglalkozni a fájl- és könyvtárkezeléssel. Sok kezdő felhasználónak azért vannak problémái a Linux operációs rendszerrel, mert nem szokták meg ezt a filozófiát. Így aztán nem értik, mi hol található a fájlrendszerben.

A fejezetben foglalkozunk a rendszer legfontosabb fájljaival és könyvtáraival. Megnézzük, milyen lehetőségeink vannak, ha a fájlok tartalmára vagyunk kíváncsiak.

3.1. A Linux fájlrendszerei

3.1.1. Fájlok

A UNIX-ban (és így a Linux-ban is) igaz megállapítás, hogy minden egy fájl, ha pedig nem fájl, akkor egy folyamat.

A Linux-ban nincs különbség a fájl és a könyvtár között, mivel a könyvtár egy olyan fájl, ami más fájlok neveit tartalmazza. A programok, szolgáltatások, képek, szövegek stb. mind fájlok. A kimeneti és bementi eszközök (és általánosságban minden eszköz) fájlként fogható fel. Aki korábban csak DOS-ban, illetve Windows operációs rendszerben dolgozott, annak ez a filozófia merőben új. A Linux-ban a fentiek miatt nem lehet például egy CD-t csupán fizikailag eltávolítani (az Eject gomb megnyomásával), hiszen a lemez fájlrendszere a használathoz be volt csatolva a rendszer fájlrendszerébe. A fájlrendszert más operációs rendszerekben faszerkezettel rajzoltuk le. Bár Linux-ban is gondolhatunk erre a fastruktúrára, itt mégsem pontos ez a jelölés.

3.1.1.1. Fájltípusok

A legtöbb fájl Linux-ban egyszerűen fájl (szokásos fájl), mely közönséges adatokat tartalmaz (futtatható fájlok, programok, szövegfájlok, egy program bemenete vagy kimenete stb.).

Bár nagyon pontos elképzelés, hogy a Linux rendszerben minden fájl, van néhány kivétel is:

- könyvtárak: olyan fájlok, melyek más fájlok listáit tartalmazzák
- különleges fájlok: a kimenethez és bemenethez használt mechanizmus. A legtöbb speciális fájl a /dev könyvtárban van (a device, vagyis eszköz rövidítése).
- hivatkozások, linkek: olyan technika, mely lehetővé teszi, hogy egy fájl vagy könyvtár a fájlrendszer több részén is látható legyen.
- domain socket: különleges fájl, ami olyan hálózatkezelést kínál, melyet a fájlrendszer hozzáférés-vezérlése véd.
- named pipe: hálózati szemantikák nélküli kommunikációt megvalósító speciális memóriarész

Az ls parancs -l opcióval történő kiegészítésére a fájltípusok is láthatóvá válnak (minden sor első karaktere mutatja a típust):

```
sikosl:~/Documents> ls -l
total 55
-rw-rw-r-- 1 sikosl sikosl 28534 Jan 21 10:33 bevezet Linux.doc
-rw-rw-r-- 1 sikosl sikosl 65742 Jan 21 10:33 Linux.doc
drwxrwxr-x 2 sikosl sikosl 4096 Jan 25 11:35 tanfolyam
```

Az egyes sorok legelső karaktere az alábbi lehet:

Szimbólum	Jelentés
-	szokásos fájl
d	könyvtár
l	hivatkozás
c	különleges fájl
s	socket
p	named pipe

Sok esetben a fájltípusok megjelenítéséhez nem kell hosszú listát kérnünk, mert számos rendszerben az ls alapértelmezett jelentése helyett az ls -F parancs hajtódik végre, így a fájlnevek után a /=*|@ karakterek egyike jelenik meg, jelezve a fájltípust. A kezdők dolgát tovább könnyítheti az -F és a --color opciók együttes használata, mely a könnyebb olvashatóságot szolgálja.
Egyszerű felhasználóként kizárólag könyvtárakkal és szokásos fájlokkal, futtatható fájlokkal és hivatkozásokkal dolgozunk. A speciális fájltípusokkal a rendszergazdák és a programozók foglalkoznak.

3.1.2. Partícionálás

Nagyon sok embernek pontatlan fogalma van a partíciókról, pedig minden operációs rendszeren létre lehet hozni és el lehet távolítani partíciókat a lemezterület egységekbe szervezéséhez. Más rendszerekkel ellentétben a Linuxnál még a legegyszerűbb telepítés esetén is több partícióra van szükség. Mi ennek az oka?

A lemezterület darabolása és a különböző partíciók használatának oka katasztrófa esetén a nagyobb adatbiztonság megvalósítása. A merevlemez partíciókra osztásával az adatok csoportosíthatók és elkülöníthetők egymástól. Ha valamilyen okból kifolyólag az egyik partíció adatai megsérülnének, a másik partíciókon valószínűleg nem lenne adatvesztés. A partícionálás oka tehát a biztonságra és robosztusságra vezethető vissza.

Ha olyan fájlrendszerünk lenne, ami csak az áramkimaradás vagy a tároló-eszközök hirtelen eltávolításából adódó adatbiztonságot garantálja, adataink nem lennének védve a hibás blokkoktól, a fájlrendszer logikai hibáitól. Ekkor használhatjuk a RAID (Redundant Array of Inexpensive Disks) technológiát.

3.1.2.1. Partíciótípusok

Egy Linux rendszerben két fő partíciótípus létezik:

- *adatpartíció:* a Linux rendszeradatai, beleértve a root partíciót, ami a rendszer indulásához és leállásához szükséges összes adatot tartalmazza
- *swap partíció:* a számítógép fizikai memóriájának kiterjesztése, memóriaként működő merevlemez-rész (swap = csere)

A legtöbb rendszer tartalmaz egy root partíciót, egy vagy több adatpartíciót és egy vagy több swap partíciót. Kevert környezetekben más rendszeradatoknak is lehet külön partíciója (például FAT vagy VFAT fájlrendszer DOS és Windows adatokhoz).

A legtöbb Linux operációs rendszeren a telepítés során az fdisk használatával állítjuk be a partíció típusát. A típust a rendszer egyes esetekben automatikusan állítja be. Ha nem így van, akkor manuálisan kell beállítani a partíciótípust. A standard Linux adatpartíció száma 83, a swap partícióé 82, melyek fájlrendszere lehet normál (ext2) vagy naplózott (ext3). Ha a fenti értékeket nem jegyeztük meg, az fdisk program beépített súgója segít nekünk.

Az ext2 és az ext3 mellett a Linux számos más fájlrendszer-típust is támogat. Ilyen a viszonylag új Reiser fájlrendszer, a JFS, az NFS, a FATxx és egyéb fájlrendszerek, többek között más operációs rendszerek saját fájlrendszerei is.

A standard root partíció (melyet a / karakter jelez) mérete körülbelül 100-500 MB. Ez tartalmazza a rendszerkonfigurációs fájlokat, az alapvető parancsokat és szerver-programokat, rendszerkönyvtárakat, néhány ideiglenes

tárolóhelyet, valamint a rendszergazda home könyvtárát. Egy hétköznapi telepítés mintegy 250 MB-ot igényel a root partíció részére.

A swap tárolót csak maga a rendszer éri el, a normál működés alatt rejtve is van. A swap biztonsítani tudja a működést és a folyamatos munka feltételeit, bármi is történik. Linux alatt elvileg soha nem láthatunk olyan idegesítő üzeneteket, hogy elfogyott a memória, zárjunk be néhány alkalmazást, majd próbáljuk újra. Mára már rengeteg operációs rendszer átvette a UNIX swap vagy virtuális memória-technológiáját. Bár a merevlemezen közel sem lehet olyan sebességgel dolgozni, mint a fizikai memóriában, azért sokszor igencsak jól jön ez a plusz memória.

Általában úgy a leghatékonyabb a rendszer, ha a swap hely mérete a merevlemezen kétszerese a gép fizikai memóriájának. Már a rendszer telepítésekor érdemes erre gondolni. Ha például 512 MB memóriánk van, az alábbi lehetőségek közül választhatunk:

- 1 db 1 GB-os swap partíció
- 2 db 512 MB-os swap partíció
- két merevlemez használata esetén mindegyiken egy-egy 512 MB-os swap partíció

Ha nagyon sok I/O műveletet hajtunk végre, a legutolsó lehetőség adja a legjobb eredményt.

Az egyes szoftverek specifikációit telepítés előtt el kell olvasni. Egyes alkalmazások (például adatbázisok) több swap helyet igényelnek. A tenyérszámítógépekben pedig nem feltétlenül áll rendelkezésre a szükséges tárhely. A swap méretét az alkalmazott kernel verziója is befolyásolja.

Számos disztribúcióban a kernel külön partíción van, mivel ez a rendszerünk legfontosabb fájlja. Ebben az esetben bizonyára van egy /boot partíciónk is, melyen a kernelünk mellett a hozzá tartozó adatfájlok is megtalálhatók.

A maradék merevlemez-tárhelyek adatpartíciókra oszthatók, bár az is előfordulhat, hogy a nem rendszer-kritikus adatok egyetlen partíción kapnak helyet (például egy munkaállomás telepítésénél). Ha a nem rendszer-kritikus adatok külön partíciókon vannak, az alábbi partíciókra szokás osztani a rendszert:

- egy partíció a felhasználói programokhoz (/usr)
- egy partíció a felhasználó személyes adatainak (/home)
- egy partíció az ideiglenes adatokhoz (/var)
- egy partíció egyéb célokra és különleges szoftverekhez (/opt)

Ha egyszer létrehoztuk a partíciókat, már csak többet készíthetünk. Ellentétben a DOS és Windows operációs rendszerekkel a meglévő partíciók mérete ugyan változtatható, de nem tanácsos.

A merevlemezek partíciókra osztását a rendszergazda végzi el. Nagyobb rendszereken egyetlen partíciót akár több merevlemezre is kiterjeszthet a megfelelő szoftver használatával. A legtöbb disztribúció standard telepítése átlagos

felhasználóknak (munkaállomásoknak) készül, valamint általános célú szerverekhez, de támogatják saját partíciók létrehozását is. A telepítési folyamat alatt meghatározhatjuk saját partíciónk kinézetét, annak jellemzőit egy kifejezetten erre a célra készült programmal grafikus felületen vagy a karakteres fdisk alkalmazással.

Egy munkaállomás vagy kliens telepítés általában egyetlen személy használatára készül. A kijelölt telepítendő programok, közös felhasználói csomagok, például asztal-témák, kliens programok elektronikus levelezéshez, multimédiás programok, webes és egyéb eszközök, fejlesztőeszközök egyetlen nagy partíción is elhelyezhetők. A swap partíciót a fizikai memória kétszeresére méretezve egy általános munkaállomást készíthetünk. Ekkor a lehető legnagyobb a személyes használatra elkülönített lemezterület, de az adatintegritás megsérülésének valószínűsége nagyobb, mint több kisebb partíció alkalmazása esetén.

A szerveren a rendszeradatokat feltétlenül el kell különíteni a felhasználói adatoktól. A szolgáltatásokat nyújtó programok optimális esetben más helyen vannak, mint az ugyanezen szolgáltatás által kezelt adatok. Az ilyen rendszereken az alábbi partíciókat szokás létrehozni:

- egy partíció az összes rendszerindításhoz szükséges adat számára
- egy partíció a konfigurációs adatok és szerverprogramok részére
- egy vagy több partíció a szerver adatok (adatbázis-táblák, felhasználói levelek, ftp-archívum stb.) részére
- egy partíció a felhasználói programoknak
- egy vagy több partíció a felhasználó-specifikus fájloknak (home könyvtárak)
- egy vagy több swap partíció (virtuális memória)

A szerverek általában nagyobb memóriával és így nagyobb swap hellyel rendelkeznek, mint a munkaállomások. Egyes szerver-folyamatok a szokásosnál több swap helyet is igényelhetnek. A jobb teljesítmény érdekében a swap-et gyakran több partícióra bontják.

3.1.2.2. A df parancs

Egy futó rendszeren a partíciók információit a df paranccsal kérhetjük le (a disk full, azaz lemez tele vagy a disk free, azaz lemez szabad rövidítéséből). A Linux-ban a df GNU verziós és támogatja a -h opciót, mely nagyban segíti az olvashatóságot (a h a human readable, azaz ember által olvasható kifejezés rövidítése). A kereskedelmi UNIX-ok saját verziójú df-fel rendelkeznek. Ezek hatása általában hasonló, bár a GNU verziós eszközök gyakran túlszárnyalják más változatok képességeit.

A df parancs csak aktív nem-swap partíciókat jeleníti meg. Ezek tartalmazhatják más hálózati rendszerek partícióit is (egyesített környezetekben nem túl ritka).

```
sikosl:~> df –h
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda7             496M  183M  288M  39% /
/dev/hda1             124M  8.4M  109M   8% /boot
/dev/hda4              19G   15G  2.7G  85% /opt
/dev/hda5              14G  10.8G  2.1G  81% /usr
/dev/hda6             3.7G  2.7G  867M  77% /var
sl:/home              8.9G  3.7G  4.7G  44% /.automount/sl/root/home
```

A partíciók egy csatlakozási ponton vannak csatlakoztatva, ami rendszerünk szinte bármely könyvtára lehet (a későbbiekben még lesz róla szó).

3.1.2.3. A fájlrendszer vizuális megjelenése

Már volt róla szó, hogy a Linux fájlrendszert rendszerint elképzelhetjük egy faszerkezetként. A következőkben megnézzük egy Red Hat Linux fájlrendszerének sémáját.

A rendszergazdától, az operációs rendszertől, a gép felhasználásától függően a szerkezet természetesen változhat. Egyes könyvtárak nem léteznek, szükség szerint pedig egyéb könyvtárakat is létrehozhatunk. Még a könyvtárneveknek sem kell feltétlenül megegyezniük az itt leírtakkal, de a konvenciót többé-kevésbé szokás tartani.

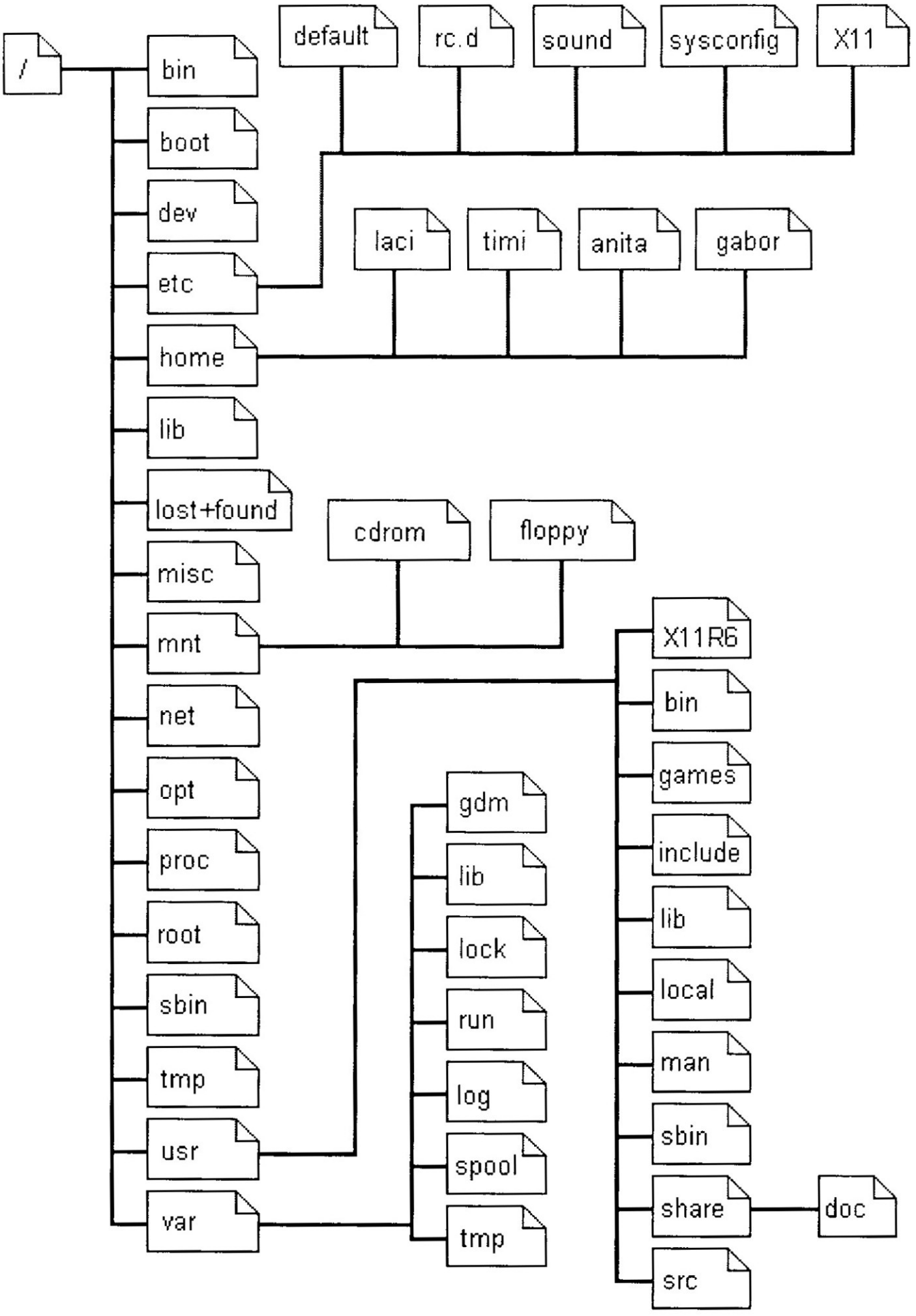
A fájlrendszer fastruktúrája a / jellel kezdődik. Mivel ebből a könyvtárból indul ki az összes többi könyvtár, szokás gyökérkönyvtárnak vagy a rendszer gyökerének is nevezni (root = gyökér).

Közvetlenül a gyökérkönyvtárból kiinduló könyvtárakat (a gyökérkönyvtár első szintű alkönyvtárait) a / jel előzi meg, ez jelzi helyüket a fájlrendszerben. Értelemszerűen emiatt nem lehet összekeverni ezeket a könyvtárakat más, alsóbb szintű, ugyanilyen nevű könyvtárakkal (ha vannak ilyenek).

Ha egy új rendszert indítunk el, célszerű megismerkedni a gyökérkönyvtárral. Ehhez a már megismert ls parancsra van szükség. Ha nem a gyökérkönyvtárban vagyunk, a cd / paranccsal léphetünk oda:

```
sikosl:~> cd /

sikosl:/> ls
bin/    dev/    home/   lib/          misc/   opt/    root/   tmp/   var/
boot/   etc/    initrd/ lost+found/   mnt/    proc/   sbin/   usr/
```


/
bin
boot
dev
etc
home
lib
lost+found
misc
mnt
net
opt
proc
root
sbin
tmp
usr
var
default
rc.d
sound
sysconfig
X11
laci
timi
anita
gabor
cdrom
floppy
gdm
lib
lock
run
log
spool
tmp
X11R6
bin
games
include
lib
local
man
sbin
share
doc
src

A gyökérkönyvtár jellemző alkönyvtárai és azok tartalma:

/bin	A rendszer, a rendszergazda és a felhasználók közös programjai.
/boot	Az indulási fájlok és a kernel, a vmlinuz. Egyes újabb disztribúciókban vannak itt grub adatok is. (A grub a GRand Unified Boot loader, vagyis fő egyesített indulási betöltő kifejezésből képzett név.)
/dev	Hivatkozásokat tartalmaz az összes CPU-n kívüli hardverre, melyek különleges tulajdonságú fájlokként kezelhetők.
/etc	A legfontosabb rendszerkonfigurációs fájlok helye (a Windows Vezérlőpultjához hasonló).
/home	A felhasználók home könyvtárai.
/initrd	Információ a betöltési folyamathoz (nem minden disztribúcióban). Ne távolítsuk el!
/lib	Könyvtárfájlok, beleértve a rendszer és a felhasználók számára szükséges programok összes fájltípusát.
/lost+found	Minden egyes partíció legfelső könyvtárában van egy lost+found könyvtár. Azok a fájlok vannak itt, melyeket esetleges hibás működés közben mentettünk.
/misc	Vegyes célokra.
/mnt	A standard becsatolási pont külső fájlrendszerekhez (pl. CD-ROM, digitális fényképezőgép stb.).
/net	Teljes távoli fájlrendszerek standard becsatolási pontja.
/opt	Különleges és kiegészítő szoftverekhez.
/proc	Rendszer-erőforrásokról információkat tartalmazó virtuális fájlrendszer. (Részletesebb információk a man proc paranccsal kérhetők, emellett a proc.txt részletesen kifejti a virtuális fájlrendszert.)
/root	A rendszergazda home könyvtára. Ne keverjük össze a / karakterrel jelölt gyökérkönyvtárat a /root könyvtárral, ami a rendszergazda (a root felhasználó) home könyvtára! (A root szó jelentheti a gyökeret és a rendszergazdát is.)
/sbin	A rendszer és a rendszergazda által használt programok.
/tmp	A rendszer által használt ideiglenes tárhely, melynek tartalma újraindítás után törlődik, így munkáink tárolására ne használjuk!
/usr	A felhasználói programok tárhelye (alkalmazások, könyvtárak, dokumentumok stb.).
/var	A felhasználó által létrehozott változó és ideiglenes fájlok tárolására szolgál. Ilyen fájlok például az internetről letöltött, csak ideiglenesen tárolandó állományok vagy egy CD-íráshoz létrehozott CD-kép.

Honnan lehet tudni, hogy egy könyvtár melyik partíción van? Használjuk a df parancsot egy pont opcióval kiegészítve.

Ebből nemcsak a kérdésünkre kapjuk meg a választ, hanem az adott partíció információihoz is hozzájutunk:

```
sikosl:/lib> df -h .
Filesystem            Size  Used  Avail  Use%  Mounted on
/dev/hda5             860M  155M  705M   18%   /
```

Általánosságban elmondható, hogy a gyökérkönyvtár minden könyvtára a root partíción van, hacsak más elérése nincs jelölve a df teljes listás üzemmódjában (vagy a df -h kiadásakor).

3.1.2.4. A fájlrendszer a gyakorlatban

A legtöbb felhasználónak, sőt sok esetben még a rendszergazdának is elegendő a fájlrendszert fastruktúrában elképzelni. A számítógép azonban egyáltalán nem tudja, mi is az a faszerkezet.

Minden partíciónak saját fájlrendszere van. Ha az összes fájlrendszerre egyszerre gondolunk, az egész rendszert fel lehetne fogni egy fastruktúraként, de nem ilyen egyszerű a helyzet. Egy fájlrendszerben egy fájlt egy un. *inode* reprezentál, ami egyfajta sorozatszám. Ez tartalmazza a fájl aktuális információit, a tulajdonosától kezdve a merevlemezen elfoglalt helyéig.

Minden partíció saját inode-halmazokkal rendelkezik. Több partíciós rendszerekben lehetnek azonos inode-számú fájlok is.

Minden inode egy adatszerkezetet ír le a merevlemezen. Egy merevlemez adattárolás elérésére történő inicializálásakor az inicializáló rendszer telepítési folyamata alatt vagy egy létező rendszer újabb lemezekkel történő kiegészítése során az egyes partíciókon az inode-ok száma rögzített. Ez a szám az összes típusú fájlok maximális száma (beleértve a könyvtárakat, a hivatkozásokat, a különleges fájlokat stb.), melyek egyidejűleg létezhetnek ugyanazon a partíción. Egy inode a legtöbbször 2-8 KB tárfoglalású.

Minden alkalommal, amikor egy új fájlt hozunk létre, kap egy inode-ot. Ebben az alábbi információk vannak:

- a fájl tulajdonosa és csoporttulajdonosa,
- a fájl típusa (szokásos, könyvtár stb.),
- jogosultságok a fájlhoz,
- a létrehozás, az utolsó olvasás és módosítás dátuma és ideje,
- az inode ezen információinak módosítási dátuma és ideje,
- ezen fájlra hivatkozó linkek száma,
- a fájl mérete,
- a fájl aktuális helye a fájlrendszerben.

Az egyetlen információ, amit nem tartalmaz az inode, a fájl neve és a könyvtár. Ezeket a különleges könyvtárfájlok tárolják. A fájlnevek és inode számok összevetésénél a rendszer képes a felhasználó számára érthető fa-

struktúra létrehozására. A felhasználó az inode számokat az ls parancs -i opciójával kérhetik le. Az inode-ok saját tárhellyel rendelkeznek a merevlemezen.

3.2. Tájékozódás a fájlrendszerben

3.2.1. Elérési út megadása

Ha azt várjuk a rendszertől, hogy futtasson egy parancsot, szinte sohasem kell megadni a teljes elérési utat. Ha például tudjuk, hogy az ls parancs a /bin könyvtárban van (ellenőrizhető a which -a ls paranccsal), az aktuális könyvtár tartalmának megjelenítéséhez nem szükséges a /bin/ls parancsot kiadnunk.

Ezt a PATH környezeti változó biztosítja. Ez a változó tartalmazza a rendszer azon könyvtárainak listáját, melyek futtatható fájlokat tartalmaznak, így a felhasználónak nem kell rengeteget gépelnie és a parancsok fájlrendszerbeli helyét sem kell megjegyeznie.

A PATH változó tartalmát ($) az echo parancs jeleníti meg:

```
sikosl:> echo $PATH
/opt/local/bin:/usr/X11R6/bin:/usr/bin:/usr/sbin/:/bin
```

Ebben a példában következetesen az /opt/local/bin, az /usr/X11R6/bin, az /usr/bin, az /usr/sbin és a /bin könyvtárakban keresi a rendszer a szükséges programot. Amint megtalálja, a keresés befejeződik, még akkor is, ha nem lett átnézve az összes könyvtár. Ez furcsa szituációkat eredményezhet.

Egy a felhasználó tudja, hogy egy sendsms nevű programmal küldhet SMS-eket. Ugyanezen rendszerben egy másik felhasználó tudja használni a parancsot, ő viszont nem. Az ok a PATH változó konfigurációjában keresendő:

```
[timfi@disz timi]$ sendsms
bash: sendsms: command not found
[timi@disz timi]$ echo $PATH
/bin:/usr/bin:/usr/bin/X11:/usr/X11R6/bin:/home/timi/bin
[timi@disz timi]$ su - sikosl
Password:
sikosl:~>which sendsms
sendsms is /usr/local/bin/sendsms

sikosl:~>echo $PATH
/home/sikosl/bin.Linux:/home/sikosl/bin:/usr/local/bin:/usr/local/sbin:\
/usr/X11R6/bin:/usr/bin:/usr/sbin:/bin:/sbin
```

Az su (switch user, azaz felhasználó-átkapcsolás) lehetőséget nyújt egy shell futtatására egy másik felhasználói környezetben (természetesen csak abban az esetben, ha tudjuk a felhasználó jelszavát).

A \ azt jelzi, hogy a sor többsoros, a következő sorban folytatódik (nem az Enter válassza el a két sort egymástól).

Egy másik esetben egy felhasználó a wc (word count, azaz szószámlálás) paranccsal szeretné ellenőrizni a fájl sorainak számát, de semmi sem történik, ezért a Ctrl + C billentyűkombinációval leállította a folyamatot:

```
sikosl:~> wc -l test
(Ctrl-C)
sikosl:~> which wc
wc is hashed (/home/jumper/bin/wc)
sikosl:~> echo $PATH
/home/jumper/bin:/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin:\
/usr/bin:/usr/sbin:/bin:/sbin
```

A which parancs mutatja, hogy ennek a felhasználónak van egy bin könyvtára a home könyvtárában, ami tartalmaz egy másik wc nevű programot. Mivel a home könyvtárában levő programot találja meg a rendszer a wc meghívásakor az elérési útvonalak között, ez a program futott le, melynek hatását nem értve azt leállított a felhasználó. A probléma megoldására számos megoldás kínálkozik (mint ahogy az a Linux-ban nem meglepő). Átnevezhetnénk a felhasználó wc nevű programját, esetleg használhatnánk a kívánt parancsunk teljes elérési utját (a parancsot a which parancs -a opciójával találhatjuk meg):

```
sikosl:~> /usr/bin/wc -l test
       10 test
```

Ha a felhasználó a másik könyvtárak programjait gyakrabban használja, módosíthatja az elérési utat:

```
sikosl:~> export PATH=/usr/local/bin:/usr/local/sbin:/usr/X11R6/bin:\
/usr/bin:/usr/sbin:/bin:/sbin:/home/sikosl/bin
```

Nagyon fontos, hogy a változtatások nem állandóak!

Az export parancs shell-ben történő használata esetén a módosítások nem tartósak és csak az aktuális esetre vonatkoznak (miután kijelentkeztünk).

Természetesen van lehetőség a környezet marandandó megváltoztatására is, de ezzel a későbbiekben foglalkozunk.

3.2.2. Abszolút és relatív elérési út

Az útvonalat, ami egy fájl elérési útja a fájlstruktúrában, a fa „gyökerétől" (a /-től vagy root könyvtártól) lehet leírni. Ilyenkor az útvonal egy / karakterrel kezdődik. Ezt hívjuk abszolút útvonalnak, mivel ez teljesen egyértelműen beazonosítja a rendszer egy fájlját.

A másik eset, amikor az elérési út nem / karakterrel kezdődik. Ekkor előfordulhat két fájl összekeverése, például a ~/bin/wc (a felhasználó home könyvtárában) és a bin/wc (az /usr könyvtárban). Ezt hívjuk relatív útvonalnak. A relatív útvonalakban a . és a .. jelzi az aktuális, illetve a szülőkönyvtárat.

3.2.3. A fontosabb fájlok és könyvtárak

3.2.3.1. A kernel

A kernel vagy mag a rendszer szíve. Ez bonyolítja a kommunikációt a hardverrel, a perifériákkal. A kernel biztosítja továbbá a processzusok és démonok (szerver processzusok) megfelelő időben történő indulását és leállását. A kernelnek számos további fontos tulajdonsága van, de a leglényegesebb, hogy tudjuk: a kernel a legfontosabb fájl a rendszeren.

3.2.3.2. A shell

A shell vagy burok egyrészt egy parancsnyelvi interpreter (soronkénti értelmező) és menedzseli a rendszer és a felhasználó közötti interakciókat. Másrészt ennél jóval többet is tud. Felfogható egy olyan nyelvként, amelyen a számítógéppel kommunikálhatunk (grafikus felületen ez a klikkelés és pozicionálás nyelvének felel meg).

A shell egy fejlett módszert valósít meg a rendszerrel történő kommunikációhoz. Rajta keresztül mintegy „társaloghatunk" a rendszerrel. A shell kezdeményező, a „beszélő" felek pedig egyenlőek. A shell lehetővé teszi a felhasználó számára, hogy rugalmas módon kezelje a rendszert.

A shellnek több típusa létezik, mint ahogy az emberek is beszélhetnek több nyelven és dialektusban:

- sh: más néven Bourne Shell. Az eredeti, UNIX rendszereken használt alapvető shell. Kisméretű program számos tulajdonsággal. POSIX-kompatibilis módban a bash ezt a shellt emulálja.
- bash: más néven Bourne Again Shell. A standard GNU shell, mely intuitív és rugalmas. A kezdő felhasználók számára ugyanolyan megfelelő, mint a haladó vagy profi felhasználóknak. Linux-ban a felhasználók standard shellje. A bash ún. főhalmaza a Bourne Shell-nek (kiegészítőkkel, beépülő modulokkal). A Bourne Again Shell tehát kompatibilis a Bourne Shell-lel (az sh-ban működő parancsok a bash-ben is elvégzik dolgukat, de fordítva már nem igaz).
- csh: C Shell. Szintaxisa a C programozási nyelvéhez hasonlít. A programozók használják néha.
- tcsh: Turbo C Shell. A C Shell főhalmaza, fejlettebb felhasználó-barátsággal és sebességgel.

- ksh: Korn Shell. A UNIX-os ismeretekkel rendelkezők használják. A Bourne Shell főhalmaza, a standard konfigurációval a kezdő felhasználók rémálma.

Az /etc/shells fájl áttekintést nyújt a Linux rendszer shell-jeiről:

```
sikosl:~> cat /etc/shells
/bin/bash
/bin/sh
/bin/tcsh
/bin/csh
```

Megjegyzendő, hogy a /bin/sh általában egy hivatkozás a bash-re, ami így meghívva Bourne Shell-kompatibilis módban fut.

Az alapértelmezett shell az /etc/passwd fájlban van.

Az egyik shell-ről a másikra átkapcsolhatunk, ha az aktív terminálon megadjuk az új shell nevét. A rendszer megtalálja a könyvtárat, ahol a név előfordul (a PATH beállításokat használja). Ha a shell egy futtatható fájl (program), az aktuális shell aktiválja azt és az lefut. Általában kapunk egy új promptot is, mivel minden shell-nek van egy jellemző megjelenése. Például

```
sikosl:~> tcsh
[sikosl@disz ~]$
```

Ha nem tudnánk, melyik shell-t használjuk éppen, ellenőrizzük az /etc/passwd megfelelő sorát vagy gépeljük be az alábbi parancsot:

```
echo $SHELL
```

3.2.3.3. A home könyvtárunk

A home könyvtárunk az alapértelmezett célunk a rendszerhez történő csatlakozásnál. A legtöbb esetben ez a /home alkönyvtára, bár ez változhat. A home könyvtárunk egy távoli fájlszerver merevlemezén is lehet, ekkor a home könyvtárunk a /nethome/felhasznaloi_nevunk. Természetesen a rendszergazda egy teljesen más (esetleg nem teljesen érthető) könyvtárat is kijelölhet home könyvtárunknak (pl. /tanulo/kovacs).

Bármi is legyen home könyvtárunk elérési útvonala, nem kell vele túl sokat törődnünk. A home könyvtárunk elérési útja a HOME környezeti változóban van letárolva (néhány programnak szüksége van rá). Az echo paranccsal jeleníthetjük meg a változó tartalmát:

```
sikosl:~> echo $HOME
/nethome/sikosl
```

A home könyvtárunkban azt csinálunk, amit csak akarunk. Annyi könyvtárat és fájlt tehetünk bele, amennyit szeretnénk, bár a teljes adatmennyiséget a hardver és a partíciók mérete befolyásolja, néha pedig a rendszergazda által

alkalmazott kvóta-rendszernek köszönhetően ütközünk korlátokba. A lemezterület limitálása elterjedt gyakorlat, hiszen a merevlemez nem olcsóság. A nagyobb rendszerekben azonban a korlátok mérete is óriási, így nyugodtan tudunk dolgozni. A quota paranccsal nézhetjük meg az esetleges korlátozásokat:

```
sikosl@egyetem:/> quota -v
Diskquotas for user sikosl (uid 501): none
```

Ha nincsenek kvóták beállítva, a korlátozott partíciókról és korlátozásaikról kapunk egy listát. A korlátozások határai lehetnek viszonylag elmosottak, de az is előfordulhat, hogy egyáltalán nem lehet túllépni a korlátot. Részletesebb információkat az info quota és a man quota parancsok adnak.

A home könyvtárunkat egy tilde karakter (~) jelzi, ami az /utvonal_a_home_konyvtarhoz/felhasznaloi_nev szintaktikájú útvonal rövidítése. Ugyanez az útvonal van tárolva a PATH környezeti változóban, így nem kell semmit sem tennünk az aktiválásához.

3.2.4. A fontosabb konfigurációs fájlok

Mint korábban már volt róla szó, a legtöbb konfigurációs fájl az /etc könyvtárban található. A tartalmat a cat paranccsal tudjuk megnézni, ami a szövegfájlokat az alapértelmezett kimenetre (általában a monitorra) küldi. Szintaktikája:

```
cat fajl 1, fajl 2, ..., fajl N
```

A következőkben a teljesség igénye nélkül megnézzük a legfontosabb konfigurációs fájlokat:

Fájl	Információ / szolgáltatás
aliases	Levél-aliasz fájl a Sendmail levelezőszerverrel történő használathoz. A UNIX-os világban már hosszú ideje bevált szokás egy levelezőszerver futtatása minden rendszeren. Ettől nem marad le a Linux sem: szinte az összes disztribúció tartalmazza a Sendmail csomagot. Ebben a fájlban helyi felhasználói nevek megegyeznek az igazi nevekkel, ahogy azok az e-mail címekben vagy más helyi címekben előfordulnak.
bashrc	A Bourne Again Shell egész rendszerben használt konfigurációs fájlja. Az összes felhasználóhoz függvényeket és aliaszokat definiál. Más shell-eknek is megvan a saját konfigurációs fájljuk, ilyen például a cshrc.
crontab és a cron.* könyvtárak	Olyan taszkok konfigurációja, melyeknek periodikusan kell futniuk (például biztonsági mentések, rendszeradatbázisok frissítése stb.).

Fájl	Információ / szolgáltatás
default	Bizonyos parancsok alapértelmezett opciói (pl. useradd).
filesystems	Ismert fájlrendszerek (pl. ext3, vfat stb.).
fstab	Partíciók és becsatolási pontjaik listája.
ftp*	Az ftp-szerver konfigurációja (ki tud kapcsolódni, a rendszer mely része elérhető stb.).
group	Felhasználói csoportok konfigurációs fájlja, melyet a groupadd, a groupmod és a groupdel eszközökkel szerkeszthetünk. Manuálisan csak akkor nyúljunk bele, ha tényleg tudjuk, mit csinálunk.
grub.conf	Rendszerindulási információk.
hosts	Egy lista azon gépekről, melyek a hálózaton DNS nélkül csatlakozni tudnak. (Ne keverjük a rendszer hálózati konfigurációjával, mely az /etc/sysconfig-ban van.)
httpd	Konfigurációs fájlok az Apache webszerverhez.
inittab	Információk a rendszerinduláshoz (üzemmód, konzolok száma stb.).
issue	Információk a disztribúcióról (release verzió- és/vagy kernel-információ).
ld.so.conf	Könyvtárfájlok helyének leírása.
lilo.conf, silo.conf, aboot.conf ...	Boot információ a LILO (LInux LOader, azaz Linux betöltő) részére (a GRUB előtt a LILO-t használták betöltési rendszerként).
logrotate.*	Egy rendszer a nagyméretű naplófájl-gyűjtemények megelőzésére.
mail	Olyan könyvtár, mely a levelezőszerver működését tartalmazza.
modules.conf	Olyan modulok konfigurációja, melyek különleges tulajdonságokat (meghajtókat) támogatnak.
motd	A nap üzenete (Message Of The Day). A rendszerhez karakteres módban csatlakozó összes felhasználónak megmutatott üzenet. A rendszergazda így közölheti a szolgáltatásokat, a rendszer teljesítményét stb.
mtab	Az aktuálisan becsatolt fájlrendszerek. Soha ne szerkesszük ezt a fájlt!
nsswitch.conf	Névfeloldás olyan esetekre, amikor a processzus egy hosztnév feloldását igényli.
pam.d	Hitelesítő modulok konfigurációja.
passwd	Helyi felhasználók listája. A useradd, a usermod és a userdel eszközökkel szerkeszthető. Csak akkor nyúljunk bele, ha tudjuk, mit csinálunk!

Fájl	Információ / szolgáltatás
printcap	Régi, de gyakran használt nyomtató-konfigurációs fájl. Ne módosítsuk manuálisan, amíg nem tudjuk, mit csinálunk!
profile	A shell környezet rendszerkonfigurációja (változók, új fájlok alapértelmezett tulajdonságai, erőforrás-korlátozások stb.).
rc*	Az egyes futási szintek szolgáltatásainak aktiválását definiáló könyvtárak.
resolv.conf	A DNS (Domain Name Service) szerverekhez való csatlakozás módja.
sendmail.cf	A Sendmail szerver fő konfigurációs fájlja.
services	A gép által elfogadott kapcsolatok (nyitott portok).
sndconfig és sound	A hangkártya és a hangesemények konfigurációja.
ssh	A Secure Shell kliens és szerver konfigurációs fájljait tartalmazó könyvtár.
sysconfig	A rendszerkonfigurációs fájlokat tartalmazó fájlok (egér, billentyűzet, asztal, rendszeróra, hálózat stb.).
X11	Beállítások a grafikus szerverhez, az X-hez. A Red Hat Linux az XFree-t használja, ami látszik a fő konfigurációs fájlja nevében is, ami az XFree86Config. Tartalmazza továbbá a rendszeren elérhető ablakozók (például gdm, fvwm, twm stb.) általános irányítását is.
xinetd.*	Konfigurációs fájlok az internet-szolgáltatásokhoz, melyek a rendszer kiterjesztett internet-szolgáltatás démonjaiból futnak (ezen szerverek nem futtatnak független démont).

Extra csomagokat telepítve további konfigurációs fájlokat is tartalmazhat az /etc könyvtár. A konfigurációs fájlok olvasásakor segítenek bennünket a magyarázatok, kommentek. Egyes fájlok további dokumentációkkal is rendelkeznek man oldalak formájában (pl. man aliases).

3.2.5. A fontosabb eszközök

Az eszközök általánosságban egy személyi számítógép összes a processzoron kívüli, vagyis perifériikus tartozékát jelentik. A Linux rendszerekben ezen eszközök a /dev könyvtáron keresztül érhetők el (a devices, vagyis eszközök rövidítéséből). A UNIX-eszközkezelés egyik nagy előnye, hogy sem a felhasználónak, sem a rendszernek nem kell foglalkoznia az eszközök specifikációjával.

A leggyakoribb eszközök a következők:

Elnevezés	A reprezentált eszköz
cdrom	CD meghajtó.
console	Különleges elérés az aktuálisan használt konzolhoz.
cua*	Soros portok.
dsp*	Eszközök mintavételezéshez és felvételhez.
fd*	A leggyakoribb hajlékonylemezes meghajtók. Alapértelmezett a /dev/fd0H1440, ami a legelterjedtebb, vagyis az 1,44 MB-os lemezmeghajtó.
hd[a-t][1-16]	A standard IDE meghajtók a lehető legtöbb partícióval.
ir*	Infravörös eszközök.
isdn*	ISDN kapcsolatok menedzselése.
js*	Botkormány(ok).
lp*	Nyomtatók.
mem	Memória.
midi*	MIDI lejátszó.
mixer* és music	Egy keverő idealizált modellje (jelek kombinálásához vagy összegzéséhez).
modem	Modem.
mouse (msmouse, logimouse, psmouse)	Mindenfajta egér.
null	Korlátlan hulladékgyűjtő.
par*	Párhuzamos port.
pty*	Álterminálok.
radio*	Rádióamatőrök eszköze.
ram*	Rendszerbetöltési eszköz.
sd*	SCSI lemezek partícióikkal.
sequencer	A hangkártya szintetizátor-képességeit kihasználó audio-alkalmazások részére (MIDI eszközvezérlő).
tty*	vt100 terminálokat szimuláló virtuális konzolok.
usb*	USB kártya és scanner
video*	Grafikus kártyával történő használatra (video-támogatás).

3.2.6. A fontosabb változó-fájlok

A /var könyvtárban található egy olyan könyvtárcsoport, melyek specifikus, nem állandó adatok tárolására hivatottak (ellentétben az ls programmal vagy a rendszerkonfigurációs fájlokkal, melyek nagyon ritkán vagy soha nem változnak). Az összes gyakran változó fájl (naplófájlok, mailbox-ok stb.) a /var alkönyvtárban vannak.

Biztonsági okokból ezek a fájlok a fő rendszerfájloktól elkülönítve kerülnek tárolásra, így jobban figyelemmel kísérhetjük őket és szükség esetén szigoríthatjuk a hozzáférést is. Ezen fájlok közül soknak nem megfelelő a szokásos megszorítások beállítása. A /var/tmp-nek például mindenki számára írhatónak kell lennie. Ez a sok felhasználói tevékenység színtere, ahol a rendszerünkhöz az interneten keresztül kapcsolódott, névtelen felhasználó munkájával is számolni kell. Többek között ez az egyik oka annak, hogy a /var könyvtár és annak összes alkönyvtára általában egy önálló partíción szokott lenni. Így nem hozzáférhetők fájlrendszerünk fontosabb részei, mint például programjaink vagy konfigurációs fájljaink.

A /tmp könyvtárban levő fájlok minden további nélkül törölhetők. Egyes (testreszabott) rendszerekben a /tmp/var viselkedése kiszámíthatatlan. Mivel azonban ez nem túl gyakori, a /var/tmp könyvtárban nyugodtan tárolhatunk ideiglenes fájlokat (érdemes a rendszergazdával is egyeztetni). Ha saját rendszerünk urai vagyunk, akkor ez a könyvtár meglehetősen biztonságos hely, hacsak tudatosan nem módosítottuk a /var/tmp beállításait (természetesen rendszergazdaként, hiszen egy normál felhasználó ezt nem is tudja megtenni).

Bármilyen jellegű tevékenységet is végzünk a rendszeren, ne mentsünk fájlokat közvetlenül a fájlrendszer gyökérkönyvtárába (/) és ne tegyük őket az /usr könyvtárba vagy annak alkönyvtárába, illetve egyéb fenntartott helyekre sem!

Egy Unix rendszer egyik legfőbb biztonsági rendszere (mely természetesen a Linux rendszerekbe is be van építve) a napló megtartásának tulajdonsága, melynek révén minden felhasználói tevékenység, processzus, rendszeresemény stb. naplózásra kerül. Az úgynevezett syslogdaemon konfigurációs fájlja meghatározza, mely naplózási információk vannak megtartva és meddig. Az összes napló alapértelmezett helye a /var/log, ami különféle fájlokat tartalmaz az elérési naplóról, a szerver naplókról, a rendszerüzenetekről stb.

A /var könyvtárban általában szerveradatokat találunk, így elkülönítve az olyan kritikus adatoktól, mint maga a szerver program vagy annak konfigurációs fájljai. Tipikus példa a Linux rendszereken a /var/www, ami egy webszerver által nyújtott aktuális HTML oldalakat, scripteket és képeket tartalmaz. Egy FTP-szerver FTP-fájának (egy távoli kliens által letölthető adatok) is a /var alkönyvtárainak egyikében van a legjobb helye. Mivel ezen adatok publikusak és gyakran módosítják névtelen felhasználók, biztonságosabb itt tárolni, elkülönítve a kritikusabb adatokat tartalmazó partícióktól, könyvtáraktól.

A legtöbb munkaállomásos telepítésen a /var/spool minimum egy at és egy cron könyvtárat tartalmaz, melyekben ütemezett feladatok vannak. Irodai környezetekben ez a könyvtár az lpd-t is tartalmazza, amiben nyomtatási sorok, nyomtató-konfigurációs fájlok és nyomtató-naplófájlok is vannak.

Kiszolgáló rendszereken általában találunk egy /var/spool/mail könyvtárat is, mely helyi felhasználók bejövő leveleit tartalmazza. Minden felhasználó levelei egy külön fájlban vannak (ezt nevezik a felhasználó inbox-ának). Egy hasonló

könyvtár az mqueue, ami az elküldetlen leveleket tárolja. Sokfelhasználós rendszerekben ezek a könyvtárak rendkívül gyakran vannak használva. Az új felhasználók használhatják a /var/spool könyvtárat is a feldolgozandó adatok rendkívüli mennyisége miatt.

A /var/lib/rpm könyvtár az RPM-alapú disztribúciókra (RedHat Package Manager) jellemző. Itt tárolódnak az RPM csomagok információi.

3.3. Fájlműveletek

3.3.1. Fájltulajdonságok megtekintése

Korábban már foglalkoztunk az ls paranccsal. Az ls azonban jóval többre képes, számos egyéb információt tud megjeleníteni a fájlnevek mellett (fájltípusok, jogosultságok, fájlméretek, inode számok, létrehozási dátumok és időpontok, tulajdonosok, hivatkozások a fájlokra). Az ls -a opciójával az alapesetben rejtett fájlok is megjeleníthetők. Ezek a ponttal kezdődő fájlok (például a home könyvtárunk konfigurációs fájljai). Ha egy adott rendszeren dolgozunk, feltűnhet, hogy több tíz fájl és könyvtár jött létre, melyek a könyvtárindexben nem jelennek meg automatikusan. Minden könyvtár tartalmaz továbbá egy pont (.) nevű, illetve egy két pont (..) nevű fájlt, melyek inode számaikkal kombinálva beazonosítják a könyvtár helyét a fájlrendszer feszerkezetében.

Érdemes elolvasnunk az ls parancs info oldalait, mert nagyon gyakran használt és hasznos opciókkal felvértezett parancsról van szó. Az opciók kombinálhatók is, mint ahogy a legtöbb Linux-parancs opciói. Gyakori kombináció az ls -al. A parancs hosszú listát ad a fájlokról és azok tulajdonságairól, valamint a szimbolikus hivatkozásokról is. Ha ugyanezt a listát szeretnénk, de az utolsó módosítás fordított sorrendje szerint, az ls -latr parancsra van szükségünk (a leggyakrabban módosított fájlok a lista végére kerülnek). Néhány példa:

```
sikosl:~/mp3> ls
Albumok/  Radio/  Szolok/  percy/  index.html

sikosl:~/mp3> ls -a
./  .thumbs     Radio  percy/
../ Albumok/    Szolok/ index.html

sikosl:~/mp3> ls -l Radio/
total 5
drwxr-xr-x      2 sikosl sikosl   4243 Jan 28     2005 Loves/
drwxr-xr-x      2 sikosl sikosl   3524 Feb 25     2005 Tender/

sikosl:~/mp3> ls -ld Radio/
drwxr-xr-x      4 sikosl sikosl   6453 Jan 28     2005 Radio/
```

```
sikosl:~/mp3> ls -ltr
total 22
drwxr-xr-x    4 sikosl sikosl  6453 Jan 28   2005 Radio/
-rw-r--r--    1 sikosl sikosl   315 Jan 24   2003 index.html
drwxrwxr-x   30 sikosl sikosl  5463 Sep 19  18:43 Szolok/
drwxr-xr-x    2 sikosl sikosl  3657 Dec 13  23:11 percy/
drwxrwxr-x   13 sikosl sikosl  2856 Dec 15  10:38 Albumok/
```

A legtöbb Linux-verzióban az ls helyett alapból a color-ls hajtódik végre (az aliasz révén). Emiatt az ls parancs egyéb opciói nélkül is láthatjuk az egyes fájlok típusát. A standard színsémát az /etc/DIR_COLORS tartalmazza:

Szín	**Fájltípus**
kék	könyvtár
vörös	tömörített archívum
fehér	szövegfájl
rózsaszín	képfájl
ciánkék	hivatkozás
sárga	eszköz
zöld	futtatható fájl
villogó vörös	hibás hivatkozás

Aki többre is kíváncsi, annak jó szolgálatot tehet a man oldal. Az ls parancs teljes funkcionalitását és összes lehetőségét az info ls paranccsal tudhatjuk meg. A korábbiak ismertetett karakteres jelölések (/, *, @, =, |) az egyszínű (pl. nyomtatott) listánál tehetnek jó szolgálatot, könnyű olvashatóságuk révén.

Ha nem elégszünk meg az ls parancs által nyújtott fájlinformációkkal, használjuk a file parancsot. A file megpróbálja meghatározni a fájl formátumát, ehhez különféle teszteket futtat le. Nézzük néhány példát!

```
sikosl:~> file kezikonyv.pdf
kezikonyv.pdf: PDF document, version 1.4

sikosl:~> file felkelo.rm
felkelo.rm: RealMedia file

sikosl:~> file iroda.sdw
iroda.sdw: Microsoft Office Document

sikosl:~> file november.xls
november.xls: Microsoft Office Document

sikosl:~> file logo.xcf
logo.xcf: GIMP XCF image data, version 0, 148 x 42, RGB Color

sikosl:~> file suli.zip.gz
suli.zip.gz: gzip compressed data, deflated, original filename,
`suli.zip', last modified: Wed Oct 8 11:34:32 2003, os: Unix
```

```
sikosl:~> file eletrajz.txt
eletrajz.txt: ISO-8859 text

sikosl:~> file cim.png
cim.png: PNG image data, 345 x 756, 8-bit grayscale, non-interlaced

sikosl:~> file /dev/log
/dev/log: socket

sikosl:~> file vers
vers: ASCII text

sikosl:~> file ballag01.jpg
ballag01.jpg: JPEG image data, JFIF standard 1.01, resolution (DPI),
"22 May 1999", 800 x 600

sikosl:~> file cimer.gif
cimer.gif: GIF image data, version 89a, 266 x 122

sikosl:~> file keret
keret: HTML document text
sikosl:~> file konyv.ps
konyv.ps: PostScript document text conforming at level 2.0
sikosl:~> file Documentumok/
Documentumok/: directory

sikosl:~> file /dev/hda
/dev/hda: block special (3/0)
```

A file parancsnak számos opciója van. Az egyik legfontosabb a -z opció, amivel tömörített fájlokba is belenézhetünk. Ne felejtsük azonban, hogy a file parancs által adott fájlformátum-meghatározás nem mindig tökéletes (csak egy tipp), a parancsot egyes fájlok „becsaphatják".

A fájlformátumokat érdemes mindenképp kideríteni, mert ha például egy egyszerű szövegfájlok szerkesztéséhez készült programmal egy speciális, formázott szöveges dokumentumot nyitunk meg, az nem fog működni. Legroszszabb esetben még a terminál is elromlik, sőt akár sípoló hangok is jelentkezhetnek. Ilyenkor zárjuk be a terminált és indítsunk egy újat.

3.3.2. Fájlok, könyvtárak létrehozása és törlése

Manapság a legtöbb rendszer hálózatba van kötve, így a közönséges fájlokat egyik gépről át lehet másolni egy másik gépre. Különösen a grafikus környezetekben igaz, hogy a felhasználó tudta, illetve beleegyezése nélkül is jönnek létre új fájlok. A probléma kivizsgálásához nézzük meg egy standard Red Hat Linux rendszeren létrehozott új felhasználó felhasználói könyvtárát:

```
[ujfelhasznalo@disz user]$ ls -al
total 32
```

```
drwx------    3 user    user      4096  Feb 24 16 11:30 .
drwxr-xr-x    6 root    root      4096  Feb 24 16 11:30 ..
-rw-r—r--     1 user    user        52  Feb 24 11:30  .bash_logout
-rw-r—r--     1 user    user       191  Feb 24 11:30  .bash_profile
-rw-r—r--     1 user    user       124  Feb 24 11:30  .bashrc
drwxr-xr-x    3 user    user      4096  Feb 24 11:30  .kde
-rw-r—r--     1 user    user      3511  Feb 24 11:30  .screenrc
-rw-------    1 user    user        61  Feb 24 11:30  .xauthDqztLr
```

Vessük össze egy régi felhasználó home könyvtárával:

```
regifelhasznalo:~> ls
Asztal/          Desktop/         iroda/           OpenOffice.org/
bin/             Dokumentum/      kepek/           staroffice6.0/
C/               Emacs/           levelezes/       tesztek/
cikkek/          GNUstep/         Machines/        webstart/
crossover/       gyakorol/        Mail/            xml/
default/         hasznos/         mp3/             zarolt/
```

Ha az összes pont karakterrel kezdődő könyvtárat és fájlt összeszámoljuk, előfordulhat, hogy 165 elemet találunk ebben a könyvtárban. Ez azért van így, mert a legtöbb alkalmazás saját könyvtárakkal és/vagy fájlokkal rendelkezik, melyek a felhasználó-specifikus beállításokat a felhasználó home könyvtárában helyezik el. Ezen fájlok általában akkor jönnek létre, amikor először indítunk egy alkalmazást. Egyes esetekben a rendszer figyelmeztet minket, hogy szükség lenne egy könyvtár létrehozására, de a legtöbbször minden automatikusan megy végbe.

Az új fájlok szinte folyamatosan jönnek létre, hiszen a felhasználók fájlokat mentenek, munkáik különböző változatait szeretnék megőrizni, internetes alkalmazásokat futtanak, fájlokat töltenek le a helyi gépükre.

A leggyakrabban használt fájlmenedzserek Linux alatt:

- konqueror: tipikusan egy KDE asztalon használt fájlmenedzser
- mc: Midnight Commander, a Norton Commander-hez hasonló Unix-os fájlmenedzser
- nautilus: az alapértelmezett fájlmenedzser a Gnome-ban, a GNU asztalon.

3.3.2.1. Könyvtárak létrehozása

A logikailag összetartozó fájlok csoportosításának legjobb módja a könyvtárak, alkönyvtárak létrehozása (Windows-ban ez a mappa, illetve az almappa). A könyvtárkészítéshez az mkdir parancsra van szükség:

```
sikosl:~> mkdir archiv
```


```
sikosl:~> ls -ld archiv
drwxrwxrwx 2 sikosl sikosl 4096 Jan 24 13:07 archiv/
```

Könyvtárak és alkönyvtárak egy lépésben is létrehozhatók, a -p opcióval:

```
sikosl:~> cd archiv

sikosl:~/archiv> mkdir 2002 2003 2004

sikosl:~/archiv> ls
2002/ 2003/ 2004/

sikosl:~/archiv> mkdir 2004/iskola/ballagas/
mkdir: cannot create directory `2004/iskola/ballagas/':
No such file or directory

sikosl:~/archiv> mkdir -p 2004/iskola/ballagas/

sikosl:~/archiv> ls 2004/iskola/
ballagas/
```

Ha az új fájlhoz más jogosultságokra van szükség, mint az alapértelmezett létrehozási jogosultságok, az mkdir parancsot továbbra is használhatjuk részletesebb információk az info oldalakon).

A könyvtár elnevezésére ugyanazok a szabályok vonatkoznak, mint a hagyományos fájlnevekre. Tartalmára vonatkozóan az egyik legfontosabb megkötés, hogy ugyanazon könyvtárban nem lehet két megegyező nevű fájlunk (de ne felejtsük, hogy a Linux kisbetű/nagybetű-érzékeny rendszer). Egy fáljnév hosszára virtuálisan nincsenek megkötések, de általában 80 karakternél rövidebb neveket használunk, így egyetlen terminálsorba is kifférnek. Bármely tetszőleges karaktert használhatunk a fájlnevekben, de tanácsos mellőzni azokat, melyeknek a shellben speciális jelentése van.

3.3.2.2. Fájlok áthelyezése, átnevezése

A fájlok áthelyezése Linux-ban az mv paranccsal lehetséges (a move, azaz átmozgatás rövidítése):

```
sikosl:~/archiv> mv ../iskola[1-4].doc iskola/ballagas/
```

Ugyanez a parancs szolgál a fájlok átnevezésére is:

```
sikosl:~> ls tennivalo
-rw-rw-r-- 1 sikosl sikosl 1432 Feb 28 18:56 tennivalo

sikosl:~> mv tennivalo kesz

sikosl:~> ls -l kesz
-rw-rw-r-- 1 sikosl sikosl 1432 Feb 28 18:56 done
```

A példából egyértelműen látszik, hogy a parancs hatására kizárólag a fájl neve változik meg. Minden egyéb tulajdonság változatlan marad.

3.3.2.3. Fájlok másolása

A fájlok és könyvtárak másolására a cp parancs szolgál. Hasznos lehetőség az úgynevezett rekurzív másolás (az összes fájl és alkönyvtár másolása), melyet a cp parancs -R opciója vált ki. Általános szintaxisa:

```
cp[-R]fajlbol fajlba
```

Példaként tételezzük fel, hogy egy ujsrac nevű felhasználó ugyanazon Gnome asztal-beállításokat szeretné, amilyen a regisrac nevű felhasználóé. Az egyik lehetőség a regisrac beállításainak az ujsrac home könyvtárába történő másolása:

```
balazs:~> cp -R ../regisrac/.gnome/ .
```

Ekkor problémák jelentkezhetnek a fájljogosultságok miatt, de a hibák azokkal a saját fájlokkal vannak, melyre az ujsrac felhasználónak már nincs szüksége.

3.3.2.4. Fájlok eltávolítása

Az egyszerű fájlok eltávolítására szolgál az rm parancs. Az üres könyvtárakat az rmdir tünteti el (annak ellenőrzésére, hogy egy könyvtár üres-e vagy sem, használhatjuk az ls -a parancsot). Az rm paranccsal nem üres könyvtárakat is eltávolíthatunk, összes alkönyvtárukkal együtt, de vigyázzunk erre a veszélyes parancsra (-r opció).

A pont (.) és kettő pont (..) könyvtárak nem távolíthatók el, hiszen szükség van rájuk az üres könyvtárban is a fastruktúrában való helyzetük meghatározásához.

Mivel a Linux karakteres felületén a törlést általában nem lehet visszavonni, hacsak nem készítettünk biztonsági másolatot, a rendszer beállítható úgy is, hogy az rm, a cp, illetve az mv parancsok minden egyes fájlnál külön megerősítést kérjenek a felhasználótól (-i opció). Ekkor minden egyes műveletet egy Enter leütésével kell engedélyeznünk:

```
nori:~> rm -ri archive/
rm: descend into directory `archiv'? y
rm: descend into directory `archiv/iskola'? y
rm: remove directory `archiv/iskola'? y
rm: descend into directory `archiv/mentes'? y
rm: remove `archiv/mentes/rendszer0412.tar'? y
rm: remove directory `archiv/mentes'? y
rm: remove directory `archiv'? y
```

3.3.3. Fájlkeresés

3.3.3.1. A shell lehetőségei

A többszörös fájlmanipulációknál a shell automatikusan kitalálja, mely fájlokra gondolunk, ha a [és] zárójelek között jelezzük őket. A DOS és egyéb operációs rendszerekben megszokott helyettesítő (dzsóker) karakterek használatára is van lehetőség. A csillag (*) tetszőleges számú karaktert, a kérdőjel (?) egyetlen karaktert helyettesít.

Bármely típusú helyettesítésből bármennyit alkalmazhatunk a shell jóvoltából. A Bash Shell például még az olyan viszonylag bonyolult kifejezésekkel is tud dolgozni, mint az ls konyvtarnev/*/*[4-5].

Más shellekben a csillag a gépelés lerövidítésére is használatos. A cd directory parancs helyett például elegendő a cd dir* beírása. A Bash-ben erre nincs szükség, hiszen a GNU shell rendelkezik a fájlnév-kiegészítés képességével (a korábbiakban már volt róla szó). Ha egy olyan könyvtárban, ahol számos fájl van, kíváncsiak vagyunk az A betűvel kezdődő fájlokra, az ls A után ne az Enter-t, hanem a Tab billentyűt üssük le, mégpedig kétszer. Ha csak egyetlen A kezdetű fájl van, az azonnal bekerül az ls (vagy bármely más) parancs argumentumába.

3.3.3.2. A which parancs

A fájlkeresés rendkívül egyszerű módja a which (jelentése: melyik) parancs használata. A felhasználó keresési útvonalában megadott könyvtárakban kereshetünk egy adott fájlt. Természetesen ha csak futtatható programokat tartalmazó könyvtárak útvonalai vannak a keresési útvonalban, a which parancs nem működik a szokásos fájlokra. A which parancsot használjuk akkor is, amikor a Command not found (Parancs nem található) üzenetet kapjuk.

Nézzünk egy példát, melyben peti nem találja az acroread programot, mikor társának ugyenezen a rendszeren nincs ilyen jellegű problémája. A probléma a korábbiakban tárgyalt PATH problémához hasonló, hiszen peti társa azt mondta neki, hogy az /opt/acroread/bin könyvtárban található a szükséges program, de ez a könyvtár nincs benne az ő útvonalában:

```
peti:~> which acroread
/usr/bin/which: no acroread in (/bin:/usr/bin:/usr/bin/X11)
```

A probléma megoldható, ha a parancs futtatásakor a teljes útvonalat megadjuk vagy ha újraexportáljuk a PATH változó tartalmát:

```
peti:~> export PATH=$PATH:/opt/acroread/bin
peti:~> echo $PATH
/bin:/usr/bin:/usr/bin/X11:/opt/acroread/bin
```

A which paranccsal az is ellenőrizhető, ha egy parancs egy másik parancs aliasza:

```
leslie:~> which -a ls
ls is aliased to `ls -F --color=auto'
ls is /bin/ls

leslie:~> which -a which
which is aliased to `type'
which is /usr/bin/which

leslie:~> which type
type is a shell builtin
```

Ez azt is jelenti, hogy a which be van építve a shellbe, de van egy különálló változatú which is. A beépített verziójú which-nek elsőbbsége van az /usr/bin-ben levő változattal szemben, ami a UNIX-szal való kombatibilitás miatt maradt itt.

3.3.3.3. A find és a locate parancsok

A find és a locate az igazi eszközök a keresési útvonalon kívüli útvonalak keresésére. A find a UNIX-ban is ismert, nagyon hatékony eszköz. A GNU find nemcsak fájlnevek keresését engedélyezi, de keresési feltételként fájlméretet, az utolsó módosítás dátumát és egyéb fájltulajdonságokat is tud használni. A leggyakrabban azonban a fájlneves keresésnél alkalmazzuk:

```
find <utvonal> -name <keresostring>
```

Az utasítás értelmében a megadott útvonal összes fájljában és alkönyvtárában keresünk, majd kiírjuk azon fájlok neveit, melyek neve tartalmazza a keresőstringet (nem a tartalmukban keresünk).

A find parancs egy másik felhasználási területe egy meghatározott fájlméret alapján történő keresés. Tegyük fel, hogy az aktuális könyvtárban és annak alkönyvtáraiban az összes 3 MB-nál nagyobb fájlra van szükségünk. Az alábbi parancsot kell kiadni:

```
laci:~> find . -size +3000k
you_and_i.mp3
```

A find parancs ennél többre is képes: a talált fájlokon műveleteket tud végezni. Jó példa erre a fájlok eltávolítása. Először célszerű az -exec opcióval ellenőrizni, hogy a megfelelő fájlok vannak-e kijelölve. Ezután a parancsot újrafuttathatjuk a kijelölt fájlok törléséhez. Keressünk ideiglenes fájlokat (.tmp végződéssel):

```
laci:~> find . -name "*.tmp" -exec rm {} \;
laci:~>
```

A parancs annyiszor hívja meg az rm-et, ahány fájltalálatunk van. A legrosszabb esetben akár több ezer találat is lehet. Ez természetesen leterhelheti a rendszert.

A fentinél jobb megoldás a | karakter és az xargs argumentum használata az rm-mel. Ekkor ugyanis az rm parancs csak akkor hívódik meg, ha a parancssor megtelt, nem pedig minden egyes fájlnál.

1999-ben készült el a find utóda, a locate. Ez a program egyszerűbben használható, de jóval korlátozottabb, mint társa, hiszen kimenete egy olyan fájl index adatbázison alapul, ami csak naponta egyszer frissül. Előny viszont a find parancshoz képest, hogy a locate adatbázisban gyorsabb a keresés, a processzort pedig kevésbé terheli le.

Manapság a legtöbb Linux disztribúció az slocate-et használja, ami a locate biztonságosabb változata és meggátolja a felhasználókat abban, hogy olyan kimenetet kapjanak, amelyhez nincs jogosultságuk. A rendszergazda home könyvtárának fájljai például alapesetben nem érhetők el mindenki számára. Ha van egy felhasználónk, aki szeretne találni valakit, aki ért a C Shell-hez, kiadhatja a locate .cshrc parancsot, így megtekintheti az összes olyan felhasználót, akinek saját konfigurációs fájlja van a C Shell-hez. Tételezzük fel, hogy a rendszergazda és az attila nevű felhasználó futtat C Shell-t. Ekkor csak a /home/attila/.cshrc fájl fog megjelenni, a rendszergazda home könyvtárában levő fájl nem. A legtöbb rendszeren a locate egy szimbolikus hivatkozás az slocate programra:

```
viki:~> ls -l /usr/bin/locate
lrwxrwxrwx 1 root slocate 7 May 28 20:37 /usr/bin/locate -> slocate*
```

A pali felhasznaló a locate használatával megtalálhatja az általa keresett alkalmazást:

```
pali:~> locate acroread
/usr/share/icons/hicolor/16x16/apps/acroread.png
/usr/share/icons/hicolor/32x32/apps/acroread.png
/usr/share/icons/locolor/16x16/apps/acroread.png
/usr/share/icons/locolor/32x32/apps/acroread.png
/usr/local/bin/acroread
/usr/local/Acrobat/Reader/intellinux/bin/acroread
/usr/local/Acrobat/bin/acroread
```

A bin nevet nem tartalmazó könyvtárakban nem lehet a program, hiszen azok nem tartalmaznak futtatható fájlokat. Három lehetőség maradt:

```
pali:~> file /usr/local/bin/acroread
/usr/local/bin/acroread: symbolic link to ../Acrobat/bin/acroread

pali:~> file /usr/local/Acrobat/bin/acroread
/usr/local/Acrobat/bin/acroread: Bourne shell script text executable
```

```
pali:~> file /usr/local/Acrobat/Reader/intellinux/bin/acroread
/usr/local/Acrobat/Reader/intellinux/bin/acroread: ELF 32-bit LSB
executable, Intel 80386, version 1, dynamically linked (uses
shared libs), not stripped
```

Nyilván az /usr/local/bin-ben levő fájlt is használhatja pali, hiszen ez egy hivatkozás az aktuális programot indító shell scriptre.

Ha az útvonalat a lehető legrövidebben szeretnénk megadni (és így a rendszernek sem kell túl sokáig keresni minden alkalommal, mikor egy felhasználó futtani szeretné a parancsot), az /usr/local/bin-t adjuk az útvonalhoz (nem pedig a másik könyvtárakat, melyek csak az adott program bináris fájljait tartalmazzák, míg az /usr/local/bin más hasznos programokat is tartalmaz).

3.3.3.4. A grep parancs

Egy egyszerű, de ugyanakkor hatékony program a grep. Általános sorszűréshez használható, valamilyen adott mintával tér vissza. A grep programnak több ezer alkalmazása van. A timi nevű felhasználó a grep használatával nézi meg, mit is csinált a find-dal:

```
timi:~> grep -a find .bash_history
find . -name userinfo
man find
find ../ -name common.cfg
```

Ilyen esetekben hasznos lehet a bash keresőfunkciója is. A Ctrl + R billentyűkombinációval aktiválhatjuk:

```
erika:~> ^R
(reverse-i-search)`find': find `/home/erika` -name *.xml
```

Írjuk be keresőstringünket a keresőpromptnál. Minél több karaktert gépelünk, annál szigorúbb a keresés. Ekkor olvashatunk a kiadott parancsok listájában (ezt nevezik command history-nek, vagyis parancstörténetnek, ami a home könyvtárunk .bash_history-jában van). Keresőstringünk legújabb előfordulását láthatjuk. Ha az előző, ugyanezt a stringet tartalmazó parancsra van szükségünk, üssünk még egy Ctrl + R billentyűkombinációt.

Kihez tartozhat ez a home könyvtár itt az enyém mellett? A grep parancscsal kideríthető akár a telefonszáma is:

```
nandi:~> grep slaszlo /etc/passwd
slaszlo:x:981:981:Sikos Laszlo, tel 301234:/home/slaszlo:/bin/bash
```

Na és mi Orsi e-mail címe?

```
petya:~/mail> grep -i orsi *
sent-mail: To: <Orsika23@freemail.hu>
sent-mail: On Tue, 24 Jan 2005, Orsika23@freemail.hu wrote:
```

A find és a locate parancsokat gyakran használjuk a grep paranccsal kombinálva. Így komolyabb lekérdezéseket is készíthetünk.

Egyes esetekben szükség van speciális karakterek használatára (ezeket hívjuk escape-karaktereknek is). A shell számára különleges jelentéssel bíró karaktereket a Bash-ben (és a legtöbb shell-ben) a \ karakterrel megelőzve azok speciális jelentése figyelmen kívül marad. A leggyakoribb speciális karakterek a /, a ., a ? és a *. Például a

"keresostring*" (ahol a * a csillag karakter)

sort szeretnénk megjeleníteni a

"keresostring"* (ahol a * bármely karakter helyett álló dzsókerkarakter)

helyett, az alábbi parancsra van szükség:

grep "keresostring*" fajl(ok)

3.3.4. A fájltartalom megtekintésének további lehetőségei

A cat parancs mellett (ami a fájlok tartalmát az alapértelmezett kimenetre irányítja) más lehetőségeink is vannak a fájltartalmak megtekintésére.
A legegyszerűbb lehetőség értelemszerűen a grafikus eszközök használata a parancssor helyett. Ilyenek többek között:

- GIMP (GNU Image Manipulation Program): képszerkesztő
- xpdf: PDF nézegető
- gv (GhostView): PostScript fájl-nézegető
- Mozilla, Konqueror, Opera: (web)böngészők
- CDPlay, XMMS: multimédia fájlokhoz
- AbiWorld, Gnumeric, KOffice, OpenOffice: irodai alkalmazások

Mielőtt valaki behatóbban is hatulmányozná a grafikus programokat, mindenképpen tanácsos először a karakteres programokkal megismerkednie, mivel ezek képezik az összes program alapját. Ez elsősorban a szöveges fájlokra vonatkozik.
A leggyakoribb szöveges programok időrendi sorrendben:

- cat: a legelső ilyen jellegű eszköz. Kimenete szabályozhatatlan volt.
- pg: a régebbi UNIX-okon is megtalálható program. A szöveges dokumentumokat oldalanként jelenítette meg.
- more: a pg továbbfejlesztett változata. Minden Linux rendszeren elérhető.
- less: a more GNU változata. Többletfunkciókkal látták el (pl. visszafelé görgetés lehetősége).

A less használata a legcélszerűbb. Szintaxisa:

less fajl

Az egyszerű szöveges eszközöknél sajnos gyakran ütközhetünk karakterkódolási problémákba. Az amerikai kódolástól eltérő karakterek (az ékezetes

magyar karakterek is) helytelenül jelenhetnek meg. A legrosszabb esetben az egész fájl olvashatatlanná válhat.

3.3.4.1. A fájl első és utolsó sorai

A head (fej), illetve tail (farok) parancsok segítségével megjeleníthetjük egy fájl első *n* vagy utolsó *n* sorát. Nézzük meg például az utoljára kiadott 10 parancsot:

```
geza:~> tail -10 .bash_history
mkdir laci
locate configure | grep bin
man bash
cd laci
grep usable /usr/share/dict/words
grep advisable /usr/share/dict/words
info quota
man quota
echo $PATH
frm
```

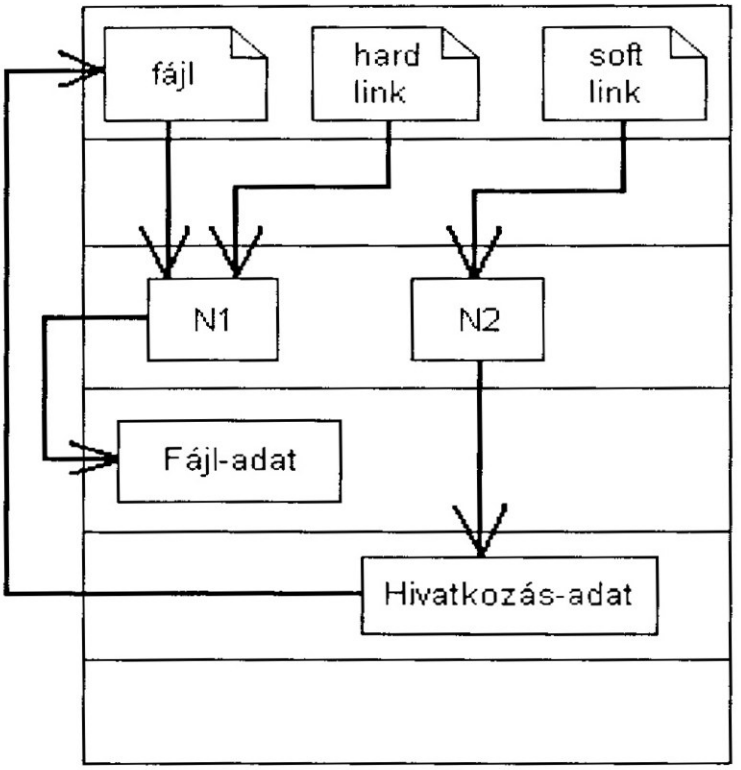
A tail parancs különösen alkalmas olyan esetekben, amikor folyamatosan változó adatokat szeretnénk megnézni. A head parancs teljesen hasonlóan működik.

3.3.5. Hivatkozások

Miután már jobban is megismertük a fájlkezelést és a fájlrendszert, nincs túl nehéz dolgunk, ha hivatkozásokat szeretnénk használni. A hivatkozás segítségével egy fájlnév látszólag két vagy több helyen is megjelenhet a fájlrendszerben. Linux alatt két lehetőségünk van:

- *hard link*: két vagy több fájlnevet ugyanazzal az inode-dal társít. A hard linkek ugyanazt az adatblokkot osztják meg a merevlemezen, de független fájlként viselkednek. Ebből következik a hard linkek hátránya is, nevezetesen, hogy nem lehetnek különböző partíción (mivel az inode számok csak egy adott partíción belül egyediek).
- *soft link*: más néven szimbolikus hivatkozás. Egy kisméretű fájl, ami egy másik fájlra mutat. A szimbolikus link csak a célfájl elérési útvonalát tartalmazza a merevlemezen történő fizikai elhelyezkedés helyett. Mivel itt nincsenek használva inode-ok, a soft linkek más partíciókra is mutathatnak.

Bár a két hivatkozás-típus hasonló, mégsem ugyanaz:



Megjegyzendő, hogy egy szimbolikus hivatkozás célfájlját átmozgatva más helyre a hivatkozás használható marad.

Linux alatt létezik egy harmadik típusú hivatkozás is, amit azonban csak a grafikus fájlmenedzser tud értelmezni (ez a hivatkozás-típus hasonlít a Windows parancsikonjaihoz). Ezen fájlok metaadatokat tartalmaznak. A kernel és a shell részére ezek csak egyszerű fájlok. Végződésük .desktop vagy .lnk. Nézzünk egy példát a KDE felületről:

```
[laci@disz Desktop]$ cat camera
[Desktop Entry]
Dev=/dev/sda1
FSType=auto
Icon=memory
MountPoint=/mnt/camera
Type=FSDevice
X-KDE-Dynamic-Device=true
```

A grafikus környezetben ilyen hivatkozást nem nagy ördöngősség előállítani.

3.3.5.1. Szimbolikus hivatkozások létrehozása

A szimbolikus link különösen a kezdők számára érdekes. Használatuk teljesen evidens és nem kell foglalkoznunk a partíciókkal sem.

A hivatkozás-készítés parancsa az ln. Szimbolikus hivatkozás készítése esetén az -s opcióra is szükség van:

```
ln -s celfajl hivatkozasnev
```

A szimbolikus linkek mérete általában igen kicsi, míg a hard linkek mérete az eredeti fájl méretével megegyező.

A szimbolikus hivatkozások használata széleskörben elterjedt. Gyakran használjuk őket helytakarékossági okokból vagy ha egy fájlról szeretnénk „másolatot" készíteni, hogy egy új program telepítésekor az a fájlt egy másik helyen is megtalálhassa. Használhatók továbbá olyan scriptek esetében is, amelyeknek egy másik környezetben azonnal kell futniuk (ráadásul egy sor munkát is megtakaríthatunk velük). Ha a rendszergazda valamilyen megfontolásból a felhasználók home könyvtárait egy új helyre (pl. tarolo2) áthelyezi, de szeretné, hogy minden úgy működjün, mint korábban (pl. az /etc/passwd fájl), készíteni fog egy szimbolikus hivatkozást a /home-ból az új helyre (pl. /tarolo2/home).

3.4. Biztonsági kérdések

3.4.1. Hozzáférési jogok

A Linux biztonsági modellje a robosztus UNIX rendszerekére épül. Linuxban minden fájlt egy felhasználó és egy felhasználó-csoport birtokol. Létezik egy harmadik felhasználói típus is, mely nem a tulajdonos felhasználója a fájlnak és nem tartozik a fájlt birtokló felhasználó-csoportba sem. Az összes felhasználónak adható írási, olvasási és futtatási jogosultság, de bármelyiktől meg is vonható.

A korábbiakban már használtuk az ls -l parancsot hosszú lista lekéréséhez. Ez a parancs a tárgyaltakon túl kijelzi a fájlhoz való hozzáférést is mindhárom felhasználó-típushoz (az első karaktert követő kilenc karakter). Nézzünk két példát:

```
robi:~> ls -l tennivalo
-rw-rw-r--      1 robi     users          5 Aug 25   18:05 tennivalo
robi:~> ls -l /bin/ls
-rwxr-xr-x      1 root     root       65748 May 5    14:27 /bin/ls*
```

A fájlt birtokló aktuális felhasználó elérési jogait a szóban forgó kilenc karakter első három karaktere jelzi. A második három karakter a felhasználó-csoport hozzáférési jogosultsága, az utolsó három pedig az egyéb felhasználóké. A jogosultságok mindig az olvasás (r, a Read rövidítéséből), írás (w, a Write rövidítéséből), futtatás (x, az eXecute rövidítéséből) sorrendben kerülnek kiírásra.

Példánk első fájlja egy szokásos fájl (első - karakter). A robi nevű felhasználók vagy a felhasználói csoportba tartozó felhasználók olvashatják vagy írhatják a fájlt (módosítás, átnevezés, törlés), de nem futtathatják azt (a 2. és 3. - karakter). Az összes többi felhasználó csak olvashatja a fájlt, de nem írhatja és nem futtathatja (4. és 5. - karakter).

A második példa egy futtatható fájl. Ezt mindenki futtathatja, de csak rendszergazdaként módosíthatjuk.

A parancsok egyszerűsítése és lerövidítése érdekében minden hozzáférési jognak és felhasználó-típusnak van egy kódja. A hozzáférési módok kódjai:

Kód	Jelentés
0 vagy -	a jogosultság nincs megadva
4 vagy r	olvasási jog
2 vagy w	írási jog
1 vagy x	futtatási jog

A felhasználói csoportok kódjai pedig a következők:

Kód	Jelentés
u	user, azaz felhasználó
g	group, azaz csoport
o	others, azaz egyéb felhasználók

Ez a biztonsági rendszer még a hálózati biztonság nélkül is magasszintű biztonságot nyújt. Más feladatai mellett ez a rendszer szabályozza a felhasználók programokhoz való hozzáférését, védi a kritikus adatokat (home könyvtárakat, konfigurációs fájlokat).

Nem árt tudnunk saját felhasználói nevünket. Ha nem tudnánk, az id paranccsal jeleníthetjük meg, sőt ekkor még azt is megtudhatjuk, melyik felhasználói csoport(ok)ba tartozunk:

```
szabi:~> id
uid=504(szabi) gid=504(szabi) groups=504(szabi),100(users),2051(org)
```

Felhasználói nevünket egyébként a USER környezeti változó is tárolja:

```
szabi:~> echo $USER
szabi
```

3.4.2. Biztonsági eszközök

3.4.2.1. A chmod parancs

A szigorú fájlhozzáférési jogosultságok (és más tényezők) egyenes következménye, hogy a jogokat a legkülönfélébb okokból meg kell változtatni. Ehhez használható a chmod parancs (*ch*anging of access *mode*, elérési mód megváltoztatása). A chmod mind a számmal, mind a kóddal történő jogosultság-megadást végrehajtja.

A következő példában alfanumerikus megadást használunk olyan esetekre, melyek az új felhasználókkal fordulnak elő:

```
lali:~> ./hello
bash: ./hello: bad interpreter: Permission denied

lali:~> cat hello
#!/bin/bash
echo "Hello, World"

lali:~> ls -l hello
-rw-rw-r--        1 lali     lali        32 Feb 27 14:35 hello

lali:~> chmod u+x hello

lali:~> ./hello
Hello, vilag

lali:~> ls -l hello
-rwxrw-r--        1 lali     lali        32 Feb 27 14:35 hello*
```

Egy felhasználó-csoport jogainak megadásához vagy megvonásához a + és a - operátorokat használhatjuk. Ha kombinációkat adunk meg, vesszővel kell tagolni.

Az előző példa fájljából csináljunk egy saját fájlt a lali felhasználónak:

```
lali:~> chmod u+rwx,go-rwx hello

lali:~> ls -l hello
-rwx------        1 lali     lali        32 Feb 27 14:35 hello*
```

Ha olyan problémával találkozunk, ami egy „hozzáférés megtagadva" jellegű hibaüzenettel lép fel, a legtöbb esetben helytelen jogosultságokkal van dolgunk. Ha valami tegnap még működött és rendszergazdaként most is működik, más felhasználóként viszont nem, valószínűleg szintén jogosultsági problémák vannak.

A chmod-ot numerikus argumentumokkal használva a megtagadott hozzáférési jogok együtt csoportba szervezhetők. Így egy háromjegyű számot kapunk. A leggyakoribb kombinációk a következők:

Parancs	**Jelentés**
chmod 400 fajl	Fájl védelme véletlenszerű felülírás ellen.
chmod 500 konyvtar	A könyvtárból történő véletlenszerű fájláthelyezés és fájlátnevezés elleni védelem.
chmod 600 fajl	Csak a parancsot kiadó felhasználó által változtatható privát fájl beállítása.
chmod 644 fajl	Publikusan olvasható, de csak a felhasználó által módosítható fájl beállítása.

Parancs	Jelentés
chmod 660 fajl	Csak a csoportunkhoz tartozó felhasználók módosíthatják a fájlt, másik pedig egyáltalán nem érik el.
chmod 700 fajl	A felhasználó teljes hozzáférése, más felhasználók minden jogának megvonása.
chmod 755 konyvtar	Mások által olvasható és futtatható, de csak a felhasználó által módosítható könyvtár beállítása.
chmod 755 fajl	Egy csoport standard fájlmegosztása.
chmod 777 fajl	Bárki bármit tehet a fájllal.

Ha a chmod argumentumaként 3-nál kisebb számot adunk meg, az elhagyott karakterek balról kezdve 0-val helyettesítődnek.

Valójában létezik egy negyedik számjegy is, ami az első hármat előzi meg és különleges hozzáférési módokat állít.

3.4.2.2. Bejelentkezés másik csoportba

Ha a parancssorba beütjük az id-t, megkapjuk a felhasználói nevünket és azonosítónkat, annak a csoportnak a nevét és azonosítóját, melyhez aktuálisan kapcsolódtunk, továbbá az összes olyan csoport listáját, melyhez jelenleg tartozunk. Számos Linux rendszeren csak egyetlen csoportban lehet aktív bejelentkezésünk. Alapesetben ez az aktív (elsődleges) csoport az egyetlen, amit az /etc/passwd fájlunkból megkapunk. A fájl negyedik mezője tartalmazza a felhasználó elsődleges csoport-azonosítóját, ami az /etc/group fájlban van. Egy példa:

```
petya:~> id
uid=501(petya) gid=501(petya) groups=100(users),501(petya),3400(web)

petya:~> grep petya /etc/passwd
petya:x:501:501:Kovacs Peter:/home/petya:/bin/bash

petya:~> grep 501 /etc/group
petya:x:501:
```

Az /etc/passwd negyedik mezője az 501 értéket tartalmazza, ami a fenti példában a petya csoportot reprezentálja. Az /etc/group-ból megkaphatjuk az ezzel a csoportazonosítóval egyező nevet. Mikor kezdetben csatlakozunk a rendszerhez, ez lesz az a csoport, melyhez petya tartozik.

A Linux rendszerek rendkívüli rugalmasságának biztosítására a legtöbb disztribúció az úgynevezett felhasználói privát csoport sémát követi, ami minden felhasználót elsődlegesen a saját csoportjához ad. Ez a csoport egy olyan csoport, ami csak ezt a szóban forgó felhasználót tartalmazza (a privát csoport elnevezés kissé megtévesztő lehet).

A saját csoportja mellett a példában szereplő petya a users és a web csoportokba is tartozhat. Mivel ezek a felhasználó másodlagos csoportjai, a newgrp használatával jelentkezhet be ezekbe a csoportokba. A példánál maradva petya olyan fájlokat szeretne létrehozni, melyeket a web csoport birtokol:

```
petya:/var/www/xhtml> newgrp web
petya:/var/www/xhtml> id
uid=501(petya) gid=3400(web) groups=100(users),501(petya),3400(web)
```

Ha petya most új fájlokat hoz létre, azok a web csoport tulajdonába kerülnek, a petya csoport helyett:

```
petya:/var/www/xhtml> touch test
petya:/var/www/xhtml> ls -l test
-rw-rw-r--          1 petya web      0 Jan 25 17:49 test
```

Egy új csoportba történő bejelentkezéssel elkerülhetjük a chown használatának szükségességét (később még visszatérünk rá) és nem kell a rendszergazdának sem kalapolnunk, hogy változtassa meg tulajdonos-beállításainkat.

3.4.2.3. A fájlmaszk

Ha valahol lementünk egy új fájlt, egy alapvető biztonsági procedúrán esik át. A Linux operációs rendszeren nem léteznek jogosultság nélküli fájlok. A standard fájlhozzáférési jogosultságokat a maszk határozza meg az új fájlok létrehozásához. Ezen maszk értéke az umask paranccsal jeleníthető meg:

```
jozsi:~> umask
0002
```

A szimbolikus értékek egymáshoz adása helyett (mint ahogy az a chmodnál működött) egy új fájl jogosultságainak kiszámításához ki kell őket vonni a teljes elérési jogokból. A fenti példában négy értéket látunk, annak ellenére, hogy csak három hozzáférési kategória van (felhasználói, csoport, egyéb). Az első nulla a speciális fájl attribútum-beállítások része. Az is előfordulhat, hogy rendszerünk az umask parancs hatására ezt az első zérust nem jeleníti meg és csak a három alapértelmezett fájllétrehozási maszkot látjuk.

Minden UNIX-alapú rendszernek van egy rendszer-funkciója a fájlok létrehozására, mely minden alkalommal meghívódik, amikor egy felhasználó használ egy olyan programot, mely új fájlokat hoz létre (pl. fájlletöltés az internetől, új szöveges dokumentum mentése stb.). Ez a funkció új fájlokat és új könyvtárakat is létrehoz. Egy új könyvtár létrehozásakor a teljes olvasási, írási és futtatási engedély mindenkitől meg lesz vonva. Egy új fájl létrehozásakor ez a funkció mindenkitől megtagadja az olvasási és írási jogokat, de beállítja a futtatási jogokat az összes felhasználói kategóriára. Másképpen fogalmazva egy maszk alkalmazása előtt egy könyvtár jogosultsága 777 (vagy rwxrwxrwx), míg egy fájlé 666 (vagy rw-rw-rw-).

Az umask érték ezen alapértelmezett jogosultságokból vonandó le, miután a funkció létrehozta az új fájlt vagy könyvtárat. Ha tehát a maszk értéke (0)002, egy könyvtár alapértelmezett jogosultsága 775, míg egy fájlé 664 lesz. Nézzük meg az alábbi példát:

```
jozsi:~> mkdir ujkonyvtar

jozsi:~> ls -ld ujkonyvtar
drwxrwxr-x       2 jozsi  jozsi                4096 Dec 26 11:34 ujkonyvtar/

jozsi:~> touch ujfajl

jozsi:~> ls -l ujfajl
-rw-rw-r--       1 jozsi  jozsi                   0 Dec 26 11:41 ujfajl
```

Ha a newgrp parancs használatával bejelentkezünk egy másik csoportba, a maszk változatlan marad. Ekkor tehát a 002 beállítással azok a fájlok és könyvtárak, melyeket az új bejelentkezéssel hoztunk létre, a csoport más tagjai számára is elérhetők lesznek (nem kell használnunk a chmod-ot).

A rendszergazda általában ennél szigorúbb alapértelmezett fájlhozzáférési jogosultságokkal rendelkezik:

```
[root@disz root]# umask
022
```

Ezen értékek rendszerszerte, a rendszer erőforrás-konfigurációs fájljaiban vannak beállítva (pl. /etc/bashrc, /etc/profile). Ezeket megváltoztatva saját shell-konfigurációs fájlunkban shell-környezetünket testreszabhatjuk.

3.4.2.4. Felhasználói és csoporttulajdonos megváltoztatása

Ha egy fájlt egy rossz felhasználó vagy csoport birtokol, a hiba a chown (change owner, tulajdonos váltása) és a chgrp (change group, csoport váltása) parancsokkal oldható meg. Egy fájl tulajdonosának megváltoztatása a rendszergazdák gyakori feladata olyan környezetekben, ahol a fájlokat egy csoportban kell megosztani. Mindkét parancs nagyon rugalmas.

A chown parancs egy fájl felhasználói és csoporttulajdonosának megváltoztatására egyaránt alkalmas. A chgrp viszont csak a csoporttulajdonos változtatására képes. A rendszer természetesen ellenőrzni, ha egy felhasználó használja a két parancs valamelyikét, hogy elegendő jogosultsággal rendelkezik-e ahhoz a fájlhoz, aminek tulajdonosán módosítani szeretne.

Ha csak a felhasználói tulajdonjogot szeretnénk módosítani egy fájlon, az alábbi szintaktikát kövessük:

```
chown ujfelhasznalo fajl
```

Ha a felhasználó neve után egy kettőspontot írunk, a csoporttulajdonos is megváltozik (a parancsot kiadó felhasználó elsődleges csoportjának megfelelően). A Linux-ban minden felhasználónak van saját csoportja, így a fájlokat saját használatúvá is tehetjük:

```
rita:~> id
uid=1304(rita) gid=(1304) groups=1304(rita),2034(projekt)

rita:~> ls -l munka
-rw-rw-r--    1 rita    projekt        45362 Feb 16 08:08 munka

rita:~> chown rita: munka
rita:~> chmod o-r munka

rita:~> ls -l munka
-rw-rw----    1 rita    rita           45362 Feb 16 08:08 munka
```

Ha rita meg szeretné osztani ezt a fájlt anélkül, hogy bárkinek is megadná az írási jogot, a chgrp parancsot használhatja:

```
rita:~> ls -l munka050216.xls
-rw-rw----    1 rita    rita           73425 Feb 16 08:09 munka050216.xls

rita:~> chgrp projekt munka050216.xls

rita:~> chmod o= munka050216.xls

rita:~> ls -l munka050216.xls
-rw-rw----    1 rita    projekt        73425 Feb 16 08:09 munka050216.xls
```

Ekkor a projekt csoport felhasználói is tudnak dolgozni a fájllal. A csoporton kívüli felhasználók a fájllal semmit sem tudnak kezdeni.

Mind a chown, mind a chgrp rekurzívan is meg tudják változtatni a tulajdonost (-R opció). Ebben az esetben az adott könyvtárban levő összes fájl és alkönyvtár egy adott felhasználóhoz és/vagy csoporthoz fog tartozni.

A legtöbb rendszerben a chown és a chgrp parancsok használata a privilegizált felhasználók kiváltsága. Ha nem mi vagyunk a rendszergazdák, biztonsági okokból kifolyólag nem fogjuk tudni megváltoztatni sem a felhasználói, sem a csoportra vontakozó tulajdonosokat.

3.4.2.5. Különleges módok

Hogy a rendszergazdának ne kelljen állandóan jogosultsági problémákkal küzdenie, különleges hozzáférési jogok adhatók teljes könyvtárakhoz vagy különálló programokhoz. Három különleges mód létezik:

1. „Ragadós bit" mód (sticky bit): egy munka futtatása után a parancs a rendszermemóriában marad. Eredetileg ez sok memória megspórolását szolgálta. Manapság a memóriaméretek igencsak megnőttek, így ez a felhasználás nem szükséges. Egy teljes könyvtárra alkalmazva

a „ragadós bit"-nek más jelentése van: a felhasználó csak akkor módosíthatja a könyvtár fájljait, ha ő a fájl tulajdonosa vagy ha a fájlokhoz ilyen jogosultságok vannak. Ez a mód az olyan könyvtáraknál használatos, mint a /var/tmp, nevezetesen, amit mindenki elér, de ahol a felhasználók nem módosíthatják és nem törölhetik egymás adatait. A „ragadós bit"-et egy t betű jelzi a fájl-jogosultság mező végén:

```
andras:~> ls -ld /var/tmp
drwxrwxrwt      15 root  root      4352 Feb 13 10:13 /var/tmp/
```

A „ragadós bit" beállítása a chmod o+t konyvtar paranccsal történik. (A t karakternek itt történeti eredete van: a UNIX rendszereken a save Text access rövidítése volt.)

2. SUID (set user ID) és SGID (set group ID): a felhasználói, illetve csoportjogosultsági mezőben az s karakter jelzi. Ezt a módot beállítva egy futtatható fájlra az a fájl felhasználói és csoportjogosultságaival fog futni (és nem a parancsot futtató felhasználó jogosultságaival). Ez a rendszererőforrások elérésénél használatos.

3. SGID egy könyvtáron: ebben a különleges esetben a könyvtárban létrehozott összes fájlnak ugyanaz lesz a csoporttulajdonosa, mint magának a könyvtárnak (normál esetben ugyebár ezen új fájlokat az őket létrehozó felhasználók birtokolnák). Ezt a módot alkalmazva a felhasználóknak nem kell foglalkozniuk a fájlok tulajdonjogával könyvtárak megosztása esetén:

```
gyula:~> ls -ld /opt/docs
drwxrws---      4 root   users      4096 Jun 10 2005 docs/

gyula:~> ls -l /opt/docs
-rw-rw----      1 gyula users    435264 Jul 12 2005-egyetem.doc
```

Ez a UNIX rendszerek alapvető fájlmegosztási módja.
A létező fájlok változatlanok maradnak! Azok a fájlok, melyeket bemozgattunk az SGID könyvtárba, de máshol voltak létrehozva, megtartják eredeti felhasználói és csoporttulajdonosaikat!

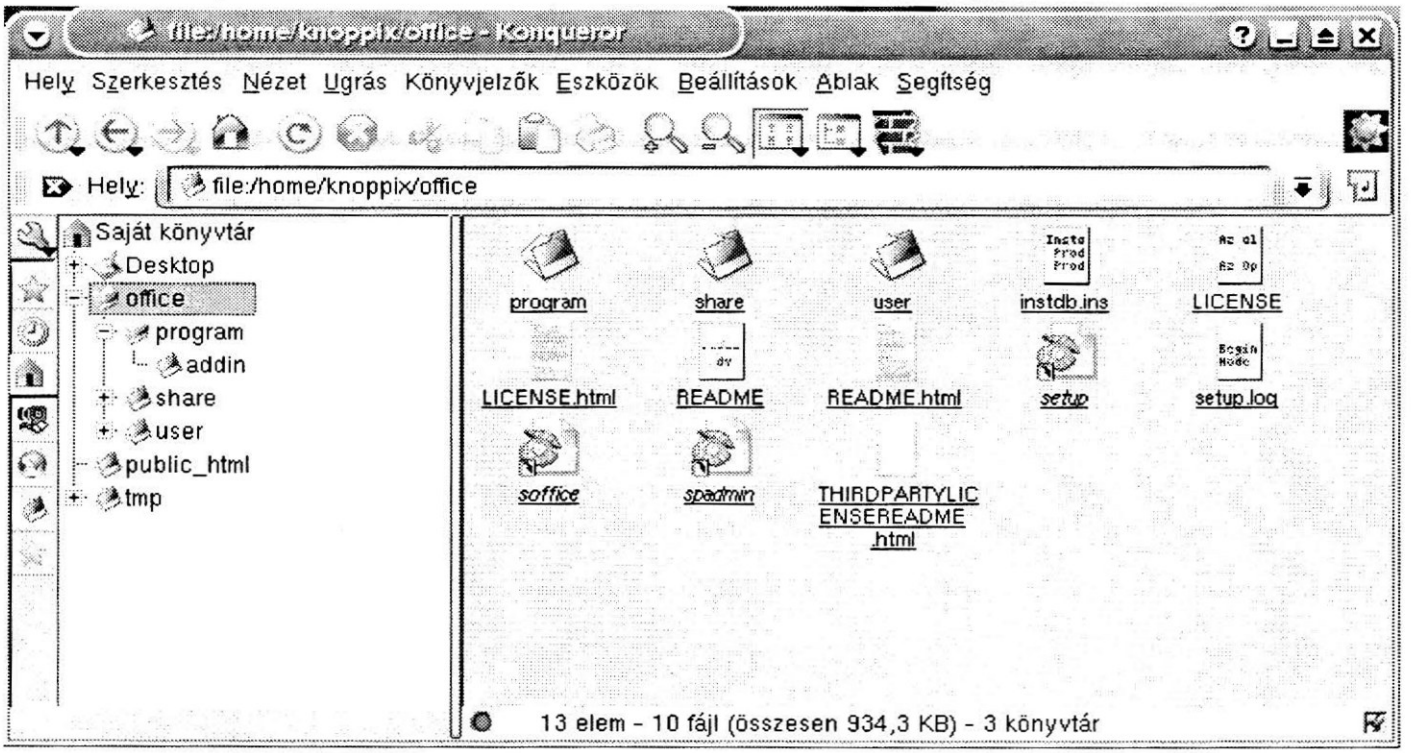
3.5. Fájlkezelés grafikus felületen

Grafikus felületen a meghajtókat, a könyvtárakat és a fájlokat hierarchikus rendszerben látjuk. Linux alá számos fájlmenedzser létezik, komoly szolgáltatásokkal. A Konqueror például nemcsak fájlok, könyvtárak átnevezésére, másolására, mozgatására alkalmas, de képes behívni a KFind nevű külső programot, mellyel kereshetünk is. A fentieken túl számos egyéb képességgel is fel

van vértezve, így akár az interneten is böngészhetünk vele (hasonlóan a Windows Sajátgépéhez).

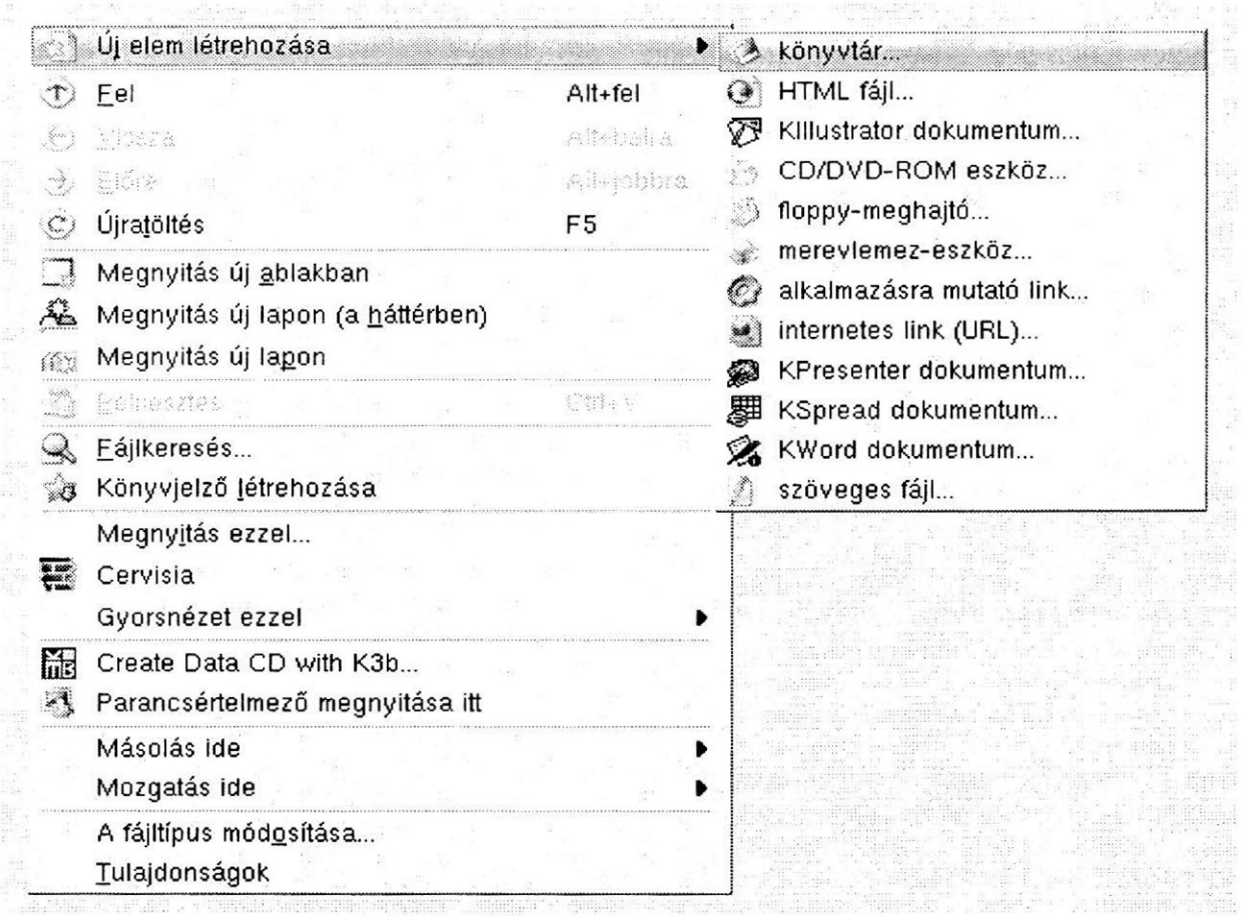
A keresésnek más módja is létezik. Használhatjuk a Konqueror jobb egérgombos helyi menüjét, ahol a fájlkeresést kell kiválasztani. Lehívhatjuk továbbá az alkalmazás menüjét is, melyben a szintén a Fájlkeresés-t kell kiválasztani. Mindegyik esetben a KFind nevű külső keresőprogram indul el, ami külön is indítható a K menüből (hasonló az eset, mint az F3-mal történő keresésnél Windows alatt).

Ne felejtsük, hogy Linux alatt számos művelethez rendszergazdai (illetve írási/módosítási) jogosultság szükséges!



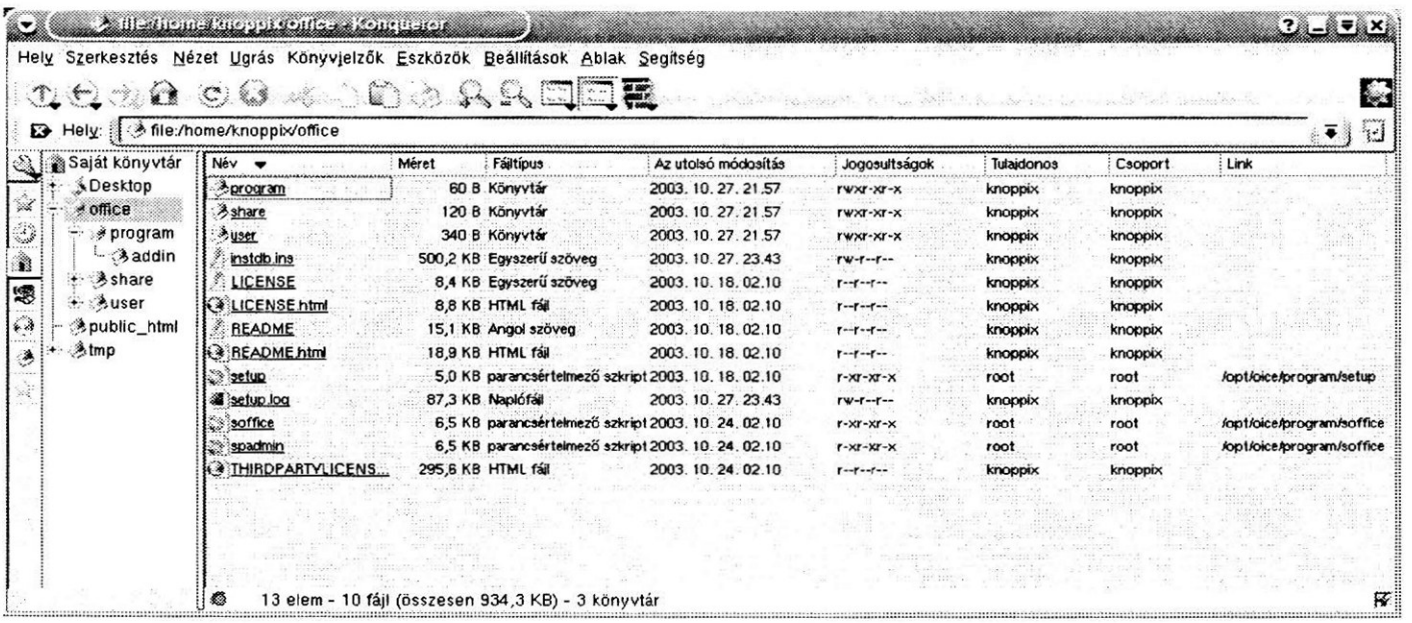
3.5.1. Fájlok, könyvtárak létrehozása

Könyvtárakat, alkönyvtárakat legegyszerűbben a Konqueror helyi menüjéből hozhatunk létre az Új elem létrehozása menü könyvtár... menüpontjával. Emellett számos fájltípust (ezek listája az aktuális rendszerünk alá telepített programoktól függ), akár hivatkozást is létrehozhatunk.



3.5.2. Fájl- és könyvtárinformációk

A fájlmenedzserben lehetőség van különféle nézetek közötti váltásokra. Ennek köszönhetően egy könyvtár tartalmát egyszer ikonszerűen jeleníthetjük meg, máskor pedig részletes leírásokkal tarkítva böngészhetjük. Utóbbi esetben a fájl neve mellett annak fizikai mérete, típusa, utolsó módosításának dátuma, a rá vonatkozó jogosultságok, tulajdonosa és tulajdonos csoporja is megjelenik. Ha hivatkozásról van szó, azt is láthatjuk, hogy az mire mutat:

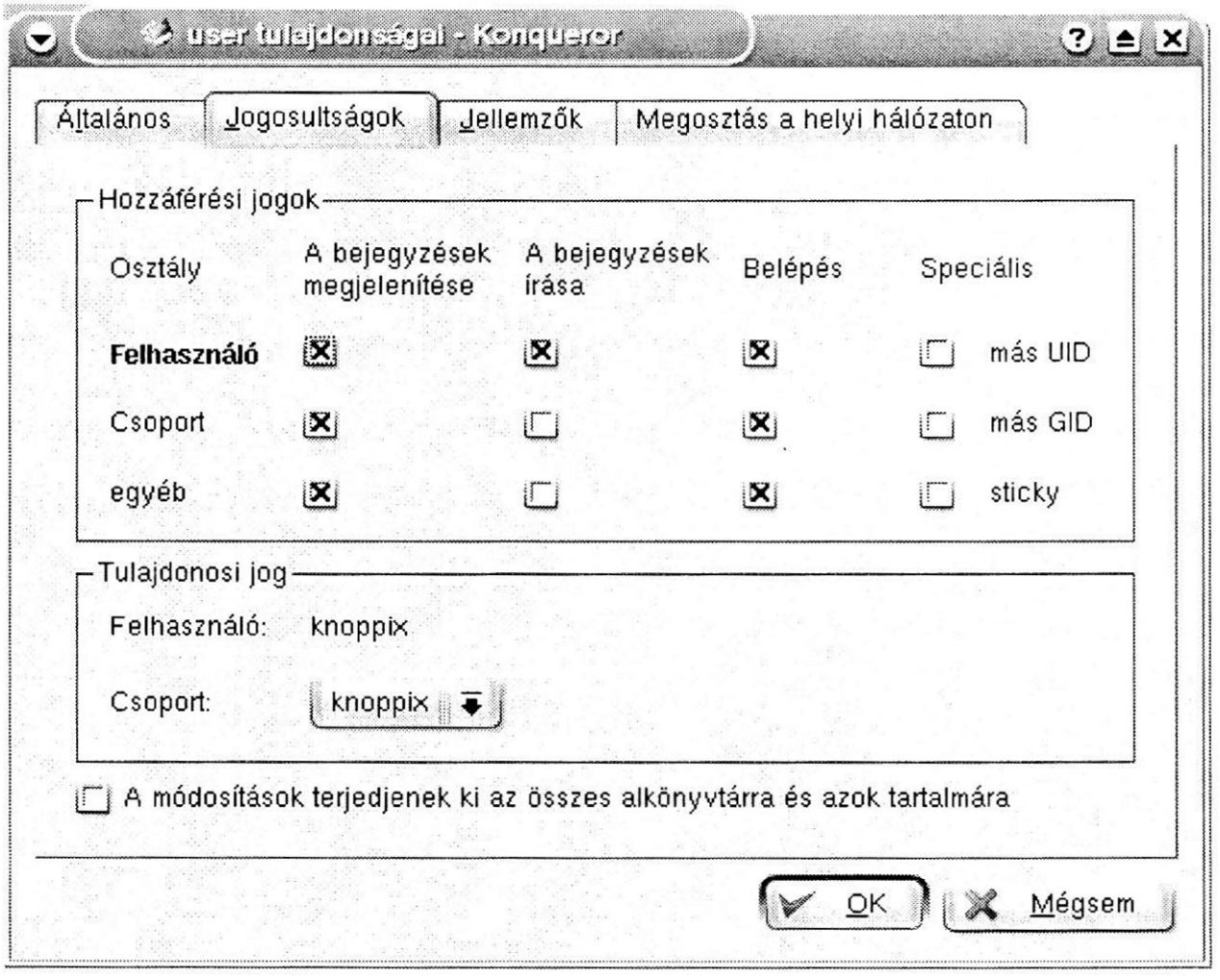


Az egyes objektumokat név, méret, típus, módosítás dátuma szerint rendezhetjük. Ehhez a megfelelő fejlécen kell egyet kattintani, ekkor a nyíl ellentétesre vált:

Fáltípus	Az utolsó módosítás ▼	Jogosultságok
Könyvtár	2003. 10. 27. 21.57	rwxr-xr-x
Könyvtár	2003. 10. 27. 21.57	rwxr-xr-x
Könyvtár	2003. 10. 27. 21.57	rwxr-xr-x
Egyszerű szöveg	2003. 10. 18. 02.10	r--r--r--
HTML fájl	2003. 10. 18. 02.10	r--r--r--
Angol szöveg	2003. 10. 18. 02.10	r--r--r--
HTML fájl	2003. 10. 18. 02.10	r--r--r--
parancsértelmező szkript	2003. 10. 18. 02.10	r-xr-xr-x
parancsértelmező szkript	2003. 10. 24. 02.10	r-xr-xr-x
parancsértelmező szkript	2003. 10. 24. 02.10	r-xr-xr-x
HTML fájl	2003. 10. 24. 02.10	r--r--r--
Naplófájl	2003. 10. 27. 23.43	rw-r--r--
Egyszerű szöveg	2003. 10. 27. 23.43	rw-r--r--

Fáltípus	Az utolsó módosítás ▲	Jogosultságok
Könyvtár	2003. 10. 27. 21.57	rwxr-xr-x
Könyvtár	2003. 10. 27. 21.57	rwxr-xr-x
Könyvtár	2003. 10. 27. 21.57	rwxr-xr-x
Egyszerű szöveg	2003. 10. 27. 23.43	rw-r--r--
Naplófájl	2003. 10. 27. 23.43	rw-r--r--
HTML fájl	2003. 10. 24. 02.10	r--r--r--
parancsértelmező szkript	2003. 10. 24. 02.10	r-xr-xr-x
parancsértelmező szkript	2003. 10. 24. 02.10	r-xr-xr-x
parancsértelmező szkript	2003. 10. 18. 02.10	r-xr-xr-x
HTML fájl	2003. 10. 18. 02.10	r--r--r--
Angol szöveg	2003. 10. 18. 02.10	r--r--r--
HTML fájl	2003. 10. 18. 02.10	r--r--r--
Egyszerű szöveg	2003. 10. 18. 02.10	r--r--r--

Ha fájlok vagy könyvtárak attribútumait szeretnénk módosítani, a jobb egérgombbal az objektumra kattintva előhívjuk a helyi menüt, majd onnan kiválasztjuk a Tulajdonságok-at:

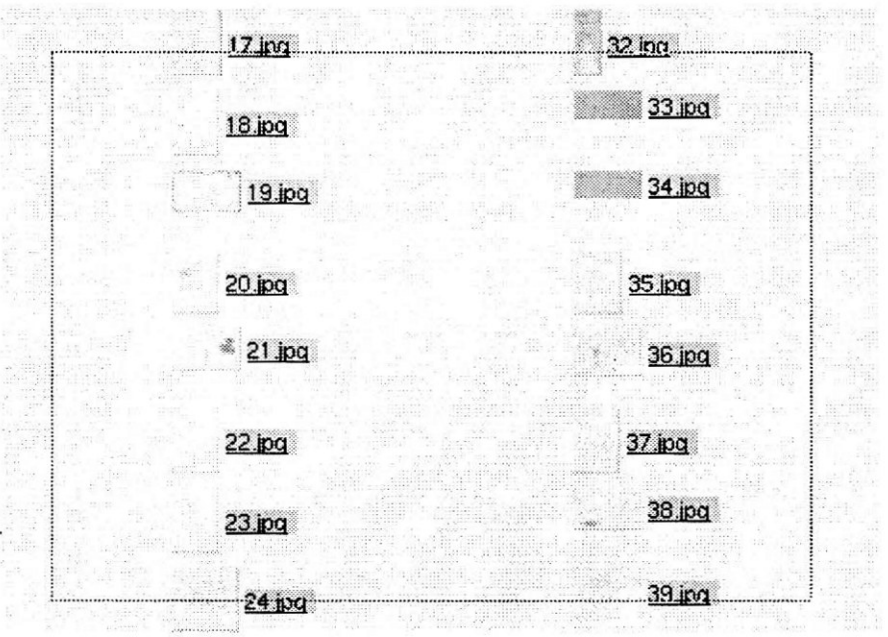


3.5.3. Átnevezés

Nagyon fontos, hogy más operációs rendszerekkel ellentétben a Linux alatt nem szükséges fájlkiterjesztéseket alkalmazni a fájltípusok meghatározásához. A különleges fájloknak és rendszerfájloknak speciális kiterjesztései lehetnek, sőt lehetőség van többszörös kiterjesztések megadására is. Ebből kifolyólag egy-egy fájl átnevezésénél érdemes igen nagy figyelmet szentelni az esetleges kiterjesztés(ek)re, hiszen ezt felülírva a magunk dolga nehezedik meg, mivel nehezebben azonosítjuk be a fájlt. A rendszerfájlokhoz – néhány különleges esetet leszámítva – értelemszerűen nem kell hozzányúlni (egyébként is csak szakavatott kezeknek ajánlott).

Az átnevezés a Linux grafikus felületein hasonlóan zajlik, mint a Windows fájlkezelőiben. A Konqueror alatt is alkalmazhatjuk az F2 funkcióbillentyűt, majd beírhatjuk a kívánt nevet. A másik lehetőség a jobb egeres helyi menü Átnevezés menüpontja (illetve az ezzel egyenértékű alkalmazás-menü megfelelő menüpontja).

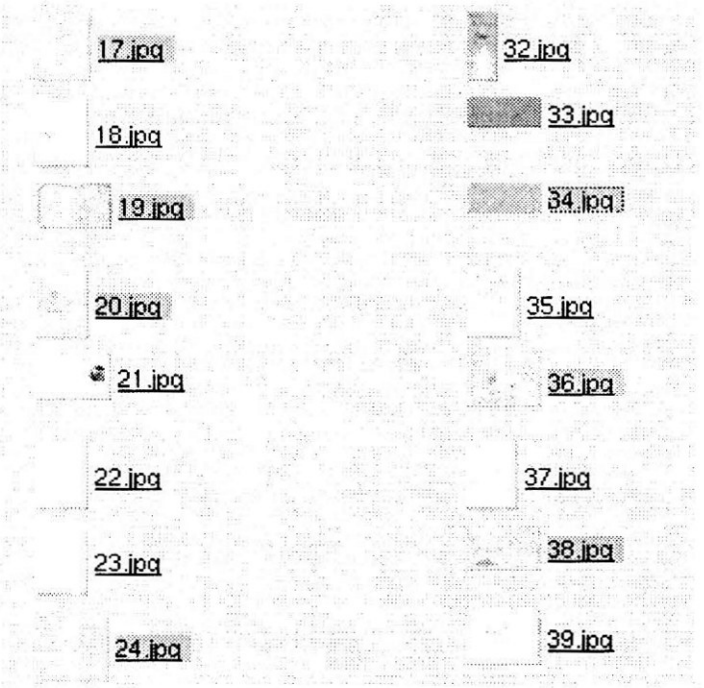
3.5.4. Másolás, áthelyezés



A fájlokat, könyvtárakat akár egyesével, akár csoportosan másolhatjuk vagy átmozgathatjuk egy új helyre. Ehhez értelemszerűen ki kell jelölni a megfelelő objektumokat. Ha egy fájlra egyet kattintunk, az kijelöltté válik, de sok esetben annak tartalma is megjelenik, ami másolási-áthelyezési műveleteknél nem éppen kívánatos. Ilyenkor több lehetőségünk is van, például a kattintás helyett a bal egérgombot lenyomva tartva körbekerítjük az adott objektumot, majd ezután engedjük fel a gombot és leütjük a megfelelő forróbillentyűt (másolásnál Ctrl+C, kivágásnál Ctrl+X).

Ha egyidejűleg több, a fájlkezelőben egymás mellett, illetve egymás alatt elhelyezkedő objektumot szeretnénk kijelölni, a Shift billentyű nyomva tartása mellett kell körbekeríteni az egérrel a megfelelő fájlokat, könyvtárakat.

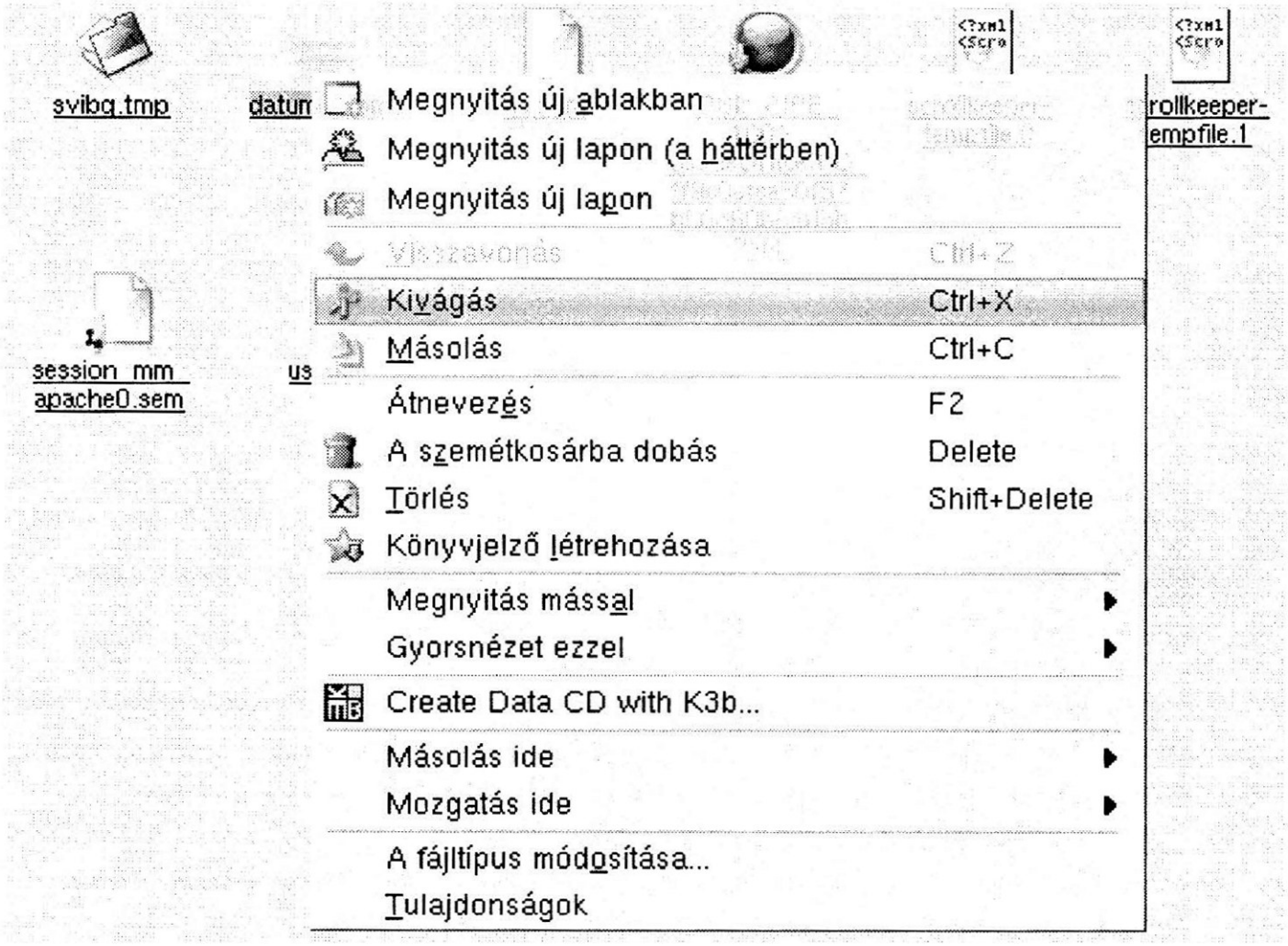
Az is előfordulhat, hogy nem egymás melletti fájlokat szeretnénk kijelölni. A Ctrl billentyű nyomvatartása mellett egyesével „csipegetjük össze" a megfelelő fájlokat úgy, hogy azokra egyesével kattintunk a bal egérgombbal.



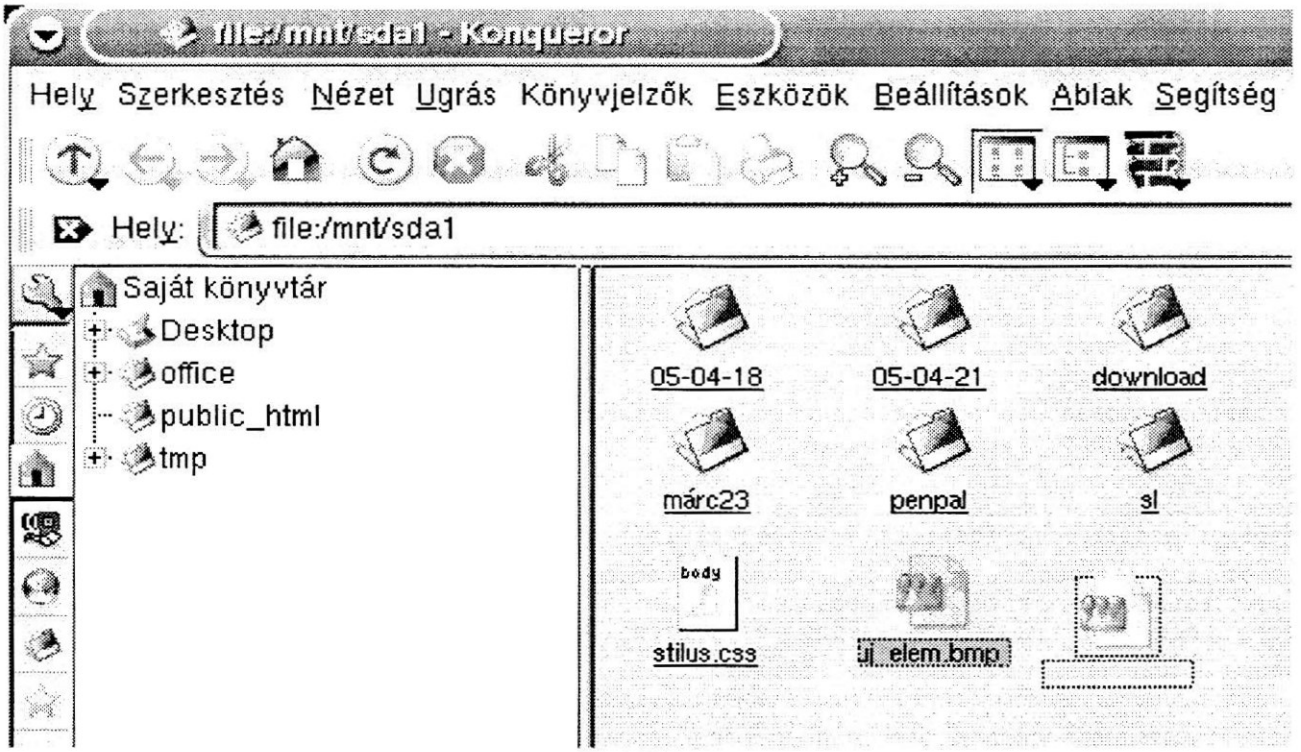
A másolás, illetve áthelyezés egyik kényelmes formája, ha két fájlkezelő ablakot nyitunk meg. Ekkor egyszerűvé válik a különböző, akár más-más meghajtón található könyvtárak közötti másolás, áthelyezés is. Mint azt már megszokhattuk, több lehetőség közül választhatunk.

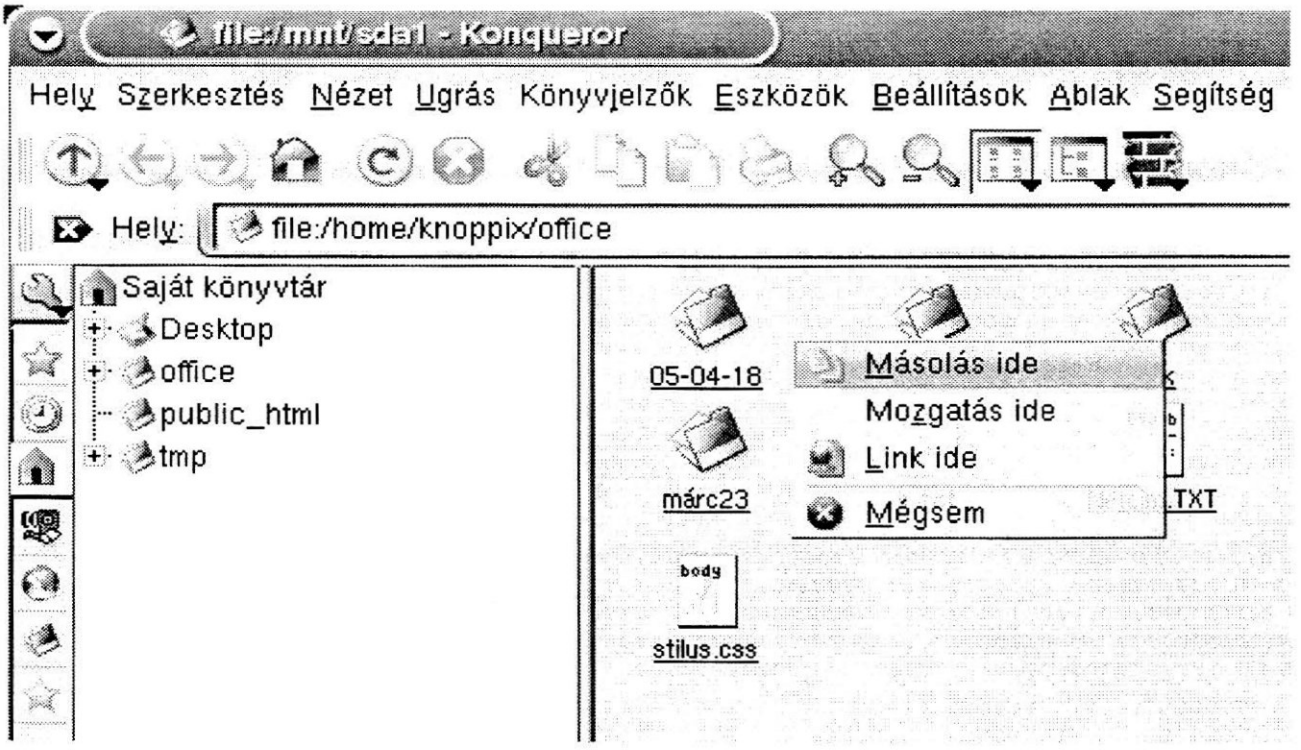
Egyik lehetőség, hogy a forráshelyen a kijelölt elem(ek)re kiadjuk a fájlkezelő Szerkesztés menüjének Másolás műveletét vagy leütjük a Ctrl + C billentyűkombinációt. A célhelyen a Szerkesztés menü Beillesztés műveletét választva (vagy leütve a Ctrl + V) billentyűkombinációt a művelet végrehajtódik.

Másik lehetőség a jobb egérgombos helyi menü lehívása és ott a kívánt művelet kiválasztása, majd a célkönyvtár meghatározása és a beillesztés parancs kiadása akár menüből, akár billentyűkombinációval.



A harmadik, és egyben a legpraktikusabb megoldás, hogy két fájlkezelő ablakot nyitunk, majd az egyikben kiválasztjuk a forrásfájl(oka)t/könyvtára(ka)t, a másikban pedig megkeressük a célkönyvtárat. A forrásfájl(ok)/könyvtár(ak) kijelölt ikonja(i) fölött lenyomjuk a bal egérgombot, majd nyomva tartva a panelen megkeressük a célkönyvtár reprezentációját, fölé helyezzük az egérmutatót, várunk és mikor megnyílik a célkönyvtár, azon felengedjük az egérgombot. Ekkor megjelenik egy helyi menü, amin választhatunk a másolás, az átmozgatás, a hivatkozás létrehozása és a művelet elvetése között.

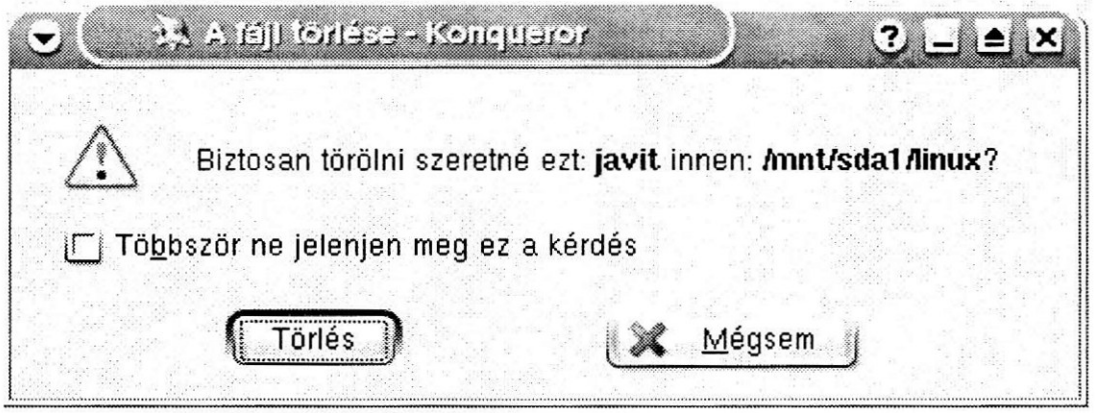




3.5.5. Törlés

Ahhoz, hogy kijelölt elemeink törlésük során a Kukába kerüljenek, a kijelölést követően a Delete billentyűt kell megnyomnunk vagy valamelyik menüből a Törlés műveletet választanunk. Ekkor megjelenik egy kérdés, hogy valóban a Kukába szeretnénk-e helyezni a kijelölt eleme(ke)t.

A Shift + Delete billentyűkombinációval történő törlés eredményeképpen a kijelölt elem(ek) véglegesen törlődik (törlődnek), nem kerülnek a Kukába, tehát nem állíthatók vissza (hasonlóan a Windows alatti törléseknél). A végleges törlés megerősítő kérdésén jól látszik, hogy nem a Kukába kerülnek az elemek:



Mivel a Kukába való törlés valójában egy speciális könyvtárba (Trash) való áthelyezést jelent, ezért a törlés később még visszavonható. A műveletet nagyobb mennyiségű adat törlésénél láthatjuk:

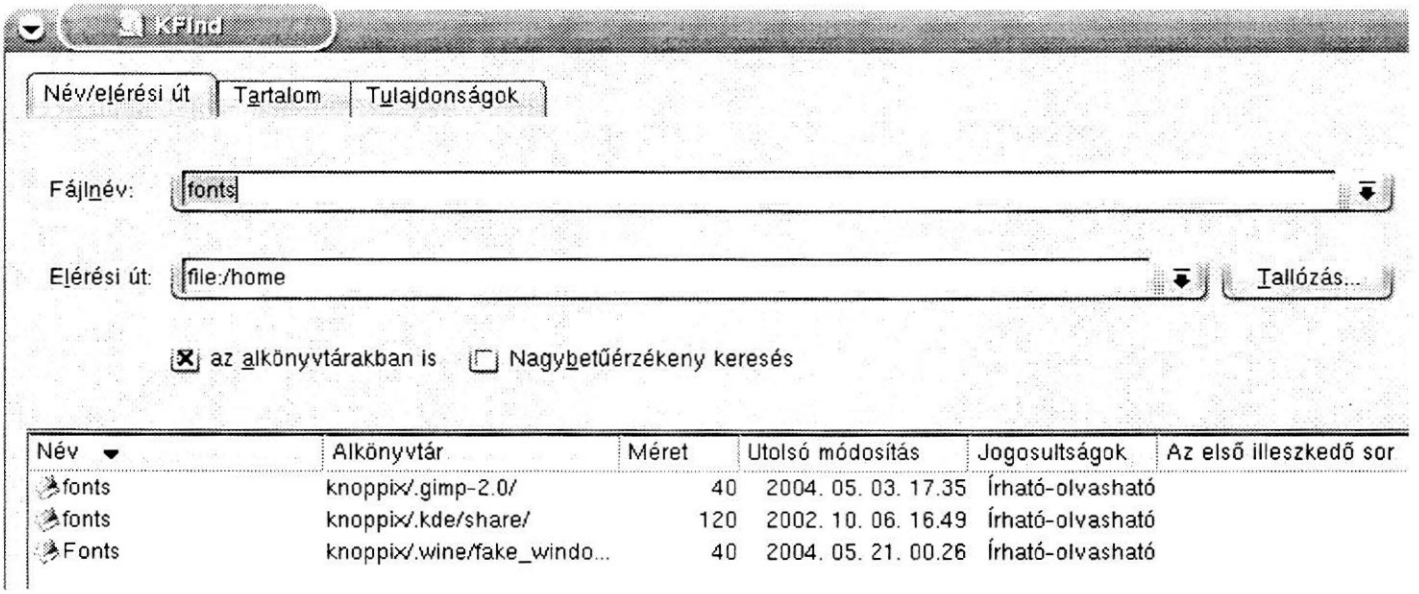
A törölt elemek visszaállításának egyik lehetséges módja, hogy a Kuka ikonjára kattintunk a jobb egérgombbal, majd az ott megjelenő helyi menüből kiválasztjuk a megfelelő menüpontot.

3.5.6. Keresés

A keresési műveletről a fájlműveletek bevezetésénél már volt szó. A K menüből közvetlenül indítható keresés a KFind programmal történik. Helyettesítő karakterek itt is alkalmazhatók, mint más operációs rendszereknél (?: egyetlen karakter, *: bármi helyett állhat). Az egyik fontos dolog, amiről nem szabad megfeledkezni, hogy Linux-ban a kiterjesztés nem olyan szerepet tölt be, mint például Windows-ban.

Ha például az összes fonts nevű fájlt keressük a home könyvtárban, a fájlnevet és az elérési utat kell megadnunk. Ez közvetlenül begépelhető, de a gördítőlistából is kiválasztható, illetve tallózható is.

Egy másik fontos dolog, hogy a Linux kisbetű-nagybetű-érzékeny rendszer, bár az erre vonatkozó szűrő állítható a keresés során:



Ha átlépünk a Tartalom lapfülre, a fájl tartalma alapján kereshetünk.

A 3. lapfül a Tulajdonságok, ahol a fizikai méret vagy akár a módosítás dátuma alapján is kereshetünk.

4. FOLYAMATOK

A fájlok után a Linux rendszereken a folyamatok (processzusok) a legfontosabbak.

4.1. A folyamatkezelés áttekintése

4.1.1. Többfelhasználós és multi-tasking rendszer

Miután kicsit jobban megismertük Linux-környezetünket és valamelyest tudunk kommunikálni a rendszerrel, a folyamatokat is behatóbban kell tanulmányoznunk. Nem minden parancs indít még egy egyszerű folyamatot sem (pl. az ls parancs egy egyszerű parancsként fut le). Mások (pl. a mozilla) egy sor folyamatot is inicializálnak.

A UNIX-on alapulva a Linux is többfelhasználós operációs rendszer, melyen egyidejűleg többen végrehajthatják ugyanazt a parancsot is. Mivel a processzor menedzseli ezeket a processzusokat, a felhasználók nyugodtan átválthatnak a folyamatok között. Egyes esetekben egy folyamat akkor is fut tovább, amikor az őt elindító felhasználó már kijelentkezett. A felhasználóknak néha szükségük lehet egy megakadt folyamat újraindításában. Többek között ezért is kell részletesebben foglalkoznunk a folyamatokkal.

4.1.2. Folyamattípusok

4.1.2.1. Interaktív folyamatok

Az interaktív folyamatok egy terminálon keresztül indíthatók és kontrollálhatók. Ezen folyamatok elindításához tehát valakinek kapcsolódnia kell a rendszerhez (nem indulnak el automatikusan a rendszerfunkciókkal). Ezek a folyamatok futhatnak az előtérben, de ezalatt más alkalmazásokat nem futtathatunk. Van azonban lehetőség arra is, hogy ezek a processzusok a háttérben fussanak. Ekkor a terminál, amelyen a programot elindítottuk, új parancsokat tud fogadni a program futása alatt.

Eddig csak olyan alkalmazásokkal foglalkoztunk, melyek az előtérben futnak. A terminál lefoglalására jó példa lehet a less parancs, ahol az aktivált program a mi reakciónkra vár. A program kapcsolódott ahhoz a terminálhoz,

ahonnan el lett indítva. Ez a terminál csak arra jó, hogy az ezen program által megértett parancsokat adhassunk ki. Más parancsok hibákhoz vagy tétlenséghez vezetnek.

Ha egy folyamat a háttérben fut, a felhasználó mást is tehet a terminállal, melyen a programot elindította, amíg az fut.

A shell kínál egy egyszerű processzus-kezelési módot. Ez a mechanizmus (melyet job control-nak is neveznek) az előtér és a háttér között váltja a folyamatokat. Ezt a rendszert használva a programok a háttérben is azonnal indíthatók.

A háttérben történő futás csak az olyan programok számára hasznos, melyek nem kérnek be adatokat a felhasználótól (a shell-en keresztül). A háttérbe akkor szokás egy tevékenységet tenni, ha annak futtatása hosszú időt vesz igénybe.

Egy terminál parancsmegadás utáni felszabadításához a parancs után írjunk egy & karaktert. Grafikus módot használva nyissunk egy plusz terminál ablakot a létezőből:

```
robi:~> xterm &
[1] 26558

robi:~> jobs
[1]+ Running                 xterm &
```

A leggyakrabban használt folyamatvezérlési lehetőségek:

Parancs vagy annak része	**Jelentés**
altalanos_parancs	A parancsot az előtérben futtatja.
parancs &	A parancsot a háttérben futtatja (felszabadítja a terminált).
jobs	Megmutatja a háttérben futó parancsokat.
Ctrl + Z	Felfüggeszt egy előtérben futó folyamatot (megállítja, de nem lép ki).
Ctrl + C	Megállít egy előtérben futó folyamatot (megállítja és kilép).
%n	Minden, a háttérben futó processzus kap egy számot. A % jel használatával egy folyamatra annak számával hivatkozhatunk (pl. fg %2).
bg	Újraindít egy felfüggesztett programot a háttérben.
fg	A folyamat újra előtérbe kerül.
kill	Befejez egy folyamatot.

A legtöbb rendszerben valószínűleg futtatható a screen is, amikor egy másik shell-lel akarunk parancsokat futtatni. A screen-t futtatva az összes program és művelet az adott shell-től függetlenül működik. Ha elhagyjuk ezt a felületet, az ott elindított programok tovább futnak, még ha ki is jelentkeztünk az eredeti shell-ből (a screen akármikor újra használható).

A program eredete arra az időszakra tehető, amikor a virtuális terminálok még nem léteztek és mindent egyetlen szöveges terminálról kellett elvégezni.

4.1.2.2. Automatikus folyamatok

Az automatikus (vagy kötegelt) folyamatok nem kapcsolódnak egy terminálhoz. Ezek olyan taszkok, melyek egy sorba rendezhetők és FIFO (először be, először ki) elven futtathatók. Két lehetőség van:

1. adott a dátum és az idő: az at parancs használatával
2. mikor az egész rendszer le van terhelve és nem tud foglalkozni a plusz folyamatokkal: a batch parancs használatával. Alapesetben a taszkok egy sorba kerülnek és addig várnak a futtatásra, amíg a rendszer terheltsége 80% alatt nem lesz. Nagy környezetekben a rendszergazdák előnyben részesítik a kötegelt feldolgozást, különösen akkor, ha nagy erőforrás-igényű folyamatokkal kellene dolgozni egy amúgy is leterhelt rendszeren. A kötegelt feldolgozás a rendszer-teljesítmény optimalizálására is felhasználható.

4.1.2.3. Démonok

A démonok folyamatosan futó szerverfolyamatok. A legtöbbször a rendszer indulásakor inicializálódnak, majd a háttérben várnak, amíg szolgáltatásukra igény nincs. Egy klasszikus példa a hálózati démon, az xinetd, ami szinte minden rendszerinduláskor elindul. Miután a rendszer elindult, a hálózati démon vár a kapcsolatot igénylő kliens programokra (pl. FTP kliens).

4.1.3. Folyamat-attribútumok

A processzusoknak számos jellemzőjük van:

- folyamatazonosító (PID): egy egyedi azonosító szám a folyamatra való hivatkozásokhoz
- a szülőprocesszus azonosítója (PPID): azon folyamat száma, mely elindította ezt a folyamatot
- „kedvességi" szám: a folyamat „barátsági foka" más folyamatokkal szemben (ne keverjük a folyamat-prioritással, ami ezen számmal és a folyamat legújabb CPU-használatával számítható ki)
- terminál (TTY): az a terminál, amihez a folyamat kapcsolódott
- az igazi és az effektív felhasználó felhasználói neve (RUID, EUID): a folyamat tulajdonosa. Az igazi tulajdonos a parancsot kiadó felhasználó, az effektív felhasználó az, aki a rendszer-erőforrásokhoz hozzáfér. A

RUID és az EUID általában ugyanaz a felhasználó és a folyamat ugyanazokkal a jogokkal rendelkezik, mint a felhasználó. Vegyünk egy példát! A mozilla böngészőt az /usr/bin-ben a rendszergazda birtokolja:

```
gyuri:~> ls -l /usr/bin/mozilla
-rwxr-xr-x  1 root   root      4996 Nov 20 18:28 /usr/bin/mozilla*

gyuri:~> mozilla &
[1] 26595

gyuri:~> ps -af
UID        PID  PPID  C STIME TTY          TIME CMD
gyuri    26691  34689 0  15:04 pts/5    00:00:00 /usr/lib/mozilla/mozilla-bin
gyuri    26703  34679 0  15:04 pts/5    00:00:00 ps –af
```

Amikor a gyuri felhasználó elindítja ezt a programot, maga a folyamat és minden, az inicializáló folyamat által elindított folyamat tulajdonosa gyuri lesz, nem pedig a rendszergazda. Amikor a mozilla hozzá akar férni a fájlokhoz, a hozzáférést gyuri jogai határozzák meg (és nem a rendszergazda jogai).

- igazi és effektív csoporttulajdonos (RGID és EGID): egy folyamat igazi csoporttulajdonosa a folyamatot elindító felhasználó elsődleges csoportja. Az effektív csoporttulajdonos általában ugyanez, kivéve, ha az SGID hozzáférési mód van alkalmazva a fájlhoz.

4.1.4. Folyamat-információk megjelenítése

A folyamatok megjelenítésének egyik eszköze a ps parancs. Számos opciója van, melyek kombinálva különböző folyamat-attbibútumokat jelenítenek meg. Opciók megadása nélkül a ps parancs csak az aktuális shell-ről és az esetleges folyamatokról ad információkat:

```
gyuri:~> ps
  PID  TTY          TIME CMD
 5463  pts/7    00:00:00 bash
 6574  pts/7    00:00:00 ps
```

Ha ebből nem tudunk meg minden, számunkra fontos információt (például mert mintegy száz folyamat fut a rendszeren), általában bizonyos folyamatokat ki szokás jelölni a grep parancs használatával:

```
ps -ef | grep username
```

Ekkor egy adott felhasználó összes folyamatát kijelöltük és megjelenítettük.

Nézzük meg a Linux rendszerek legelterjedtebb bejelentkező shell-jét, az összes bash nevű folyamatot:

```
gyuri:> ps auxw | grep bash
gina     31970 0.0 0.3  6080  1556 tty2     S    Jan26   0:00 -bash
root     32043 0.0 0.3  6112  1600 tty4     S    Jan26   0:00 -bash
gyuri    32581 0.0 0.3  6384  1864 pts/1    S    Jan26   0:00 bash
gyuri    32616 0.0 0.3  6396  1896 pts/2    S    Jan26   0:00 bash
gyuri    32629 0.0 0.3  6380  1856 pts/3    S    Jan26   0:00 bash
gyuri     2214 0.0 0.3  6412  1944 pts/5    S    16:18   0:00 bash
gyuri     4245 0.0 0.3  6392  1888 pts/7    S    17:26   0:00 bash
gyuri     5427 0.0 0.1  3720   548 pts/7    S    19:22   0:00 grep bash
```

A grep parancs ekkor a bash stringet tartalmazó sorokat az olyan rendszereken is gyakran kiírja, amelyek sok tétlen idővel rendelkeznek.

A Bash shell-ek ilyen szempontból különlegesek. Ez a folyamatlista a login shell-eket is mutatja (ahol meg kell adnunk a felhasználói nevünket és jelszavunkat szöveges módban történő bejelentkezésnél vagy egy távoli bejelentkezésnél, szemben a nem login-shell-ekkel, melyek például egy terminálablak ikonjára történő kattintással indulnak). Ezeket a bejelentkező shelleket a - karakter előzi meg.

A GNU ps más stílusú opciókat támogat (de ettől a fenti példák még nem hibásak).

Megjegyzendő, hogy a ps csak egy pillanatnyi állapotot mutat az aktív folyamatokról. A top program kijelzése sokkal pontosabb: a ps eredményeit 5 másodpercenként frissíti, új folyamatlistát generálva. Amellett, hogy pontosabb, több információt is tartalmaz (a használatban levő swap hely, a processzor állapota stb.):

```
12:40pm up 9 days, 6:00, 4 users, load average: 0.21, 0.11, 0.03
89 processes: 86 sleeping, 3 running, 0 zombie, 0 stopped
CPU states: 2.5% user, 1.7% system, 0.0% nice, 95.6% idle
Mem: 255120K av, 239412K used, 15708K free, 756K shrd, 22620K buff
Swap: 1050176K av, 76428K used, 973748K free, 82756K cached

  PID USER     PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM   TIME COMMAND
 5005 root      14   0 91572  15M 11580R       1.9  6.0   7:53 X
19599 jeff      14   0  1024 1024   796R       1.1  0.4   0:01 top
19100 jeff       9   0  5288 4948  3888R       0.5  1.9   0:24 gnome-terminal
19328 jeff       9   0 37884  36M 14724S       0.5 14.8   1:30 mozilla-bin
    1 root       8   0   516  472   464S       0.0  0.1   0:06 init
    2 root       9   0     0    0     0SW      0.0  0.0   0:02 keventd
    3 root       9   0     0    0     0SW      0.0  0.0   0:00 kapm-idled
    4 root      19  19     0    0     0SWN     0.0  0.0   0:00 ksoftirqd_CPU0
    5 root       9   0     0    0     0SW      0.0  0.0   0:33 kswapd
    6 root       9   0     0    0     0SW      0.0  0.0   0:00 kreclaimd
    7 root       9   0     0    0     0SW      0.0  0.0   0:00 bdflush
    8 root       9   0     0    0     0SW      0.0  0.0   0:05 kupdated
    9 root      -1 -20     0    0     0SW<     0.0  0.0   0:00 mdrecoveryd
   13 root       9   0     0    0     0SW      0.0  0.0   0:01 kjournald
   89 root       9   0     0    0     0SW      0.0  0.0   0:00 khubd
  219 root       9   0     0    0     0SW      0.0  0.0   0:00 kjournald
  220 root       9   0     0    0     0SW      0.0  0.0   0:00 kjournald
```

A top első sora ugyanazt az információt tartalmazza, mint az uptime parancs hatása:

```
zoli:~> uptime
4:41pm, up 10 days, 23:29, 5 users, load average: 0.01, 0.02, 0.00
```

Ezen programok adatai egyebek mellett a /var/run/utmp-ben (az aktuálisan csatlakozott felhasználók információi) és a /proc virtuális rendszerfájlban (például a /proc/loadavg az átlagos töltés információja) vannak tárolva. Számos grafikus alkalmazás létezik ezen adatok megtekintésére (pl. Gnome System Monitor).

A folyamatok közötti kapcsolatokat a pstree parancs fastruktúrába rendezve jeleníti meg:

```
moni:~> pstree
init-+-amd
     |-apmd
     |-2*[artsd]
     |-atd
     |-crond
     |-deskguide_apple
     |-eth0
     |-gdm---gdm-+-X
     |           `-gnome-session-+-Gnome
     |                           |-ssh-agent
     |                           `-true
     |-geyes_applet
     |-gkb_applet
     |-gnome-name-serv
     |-gnome-smproxy
     |-gnome-terminal-+-bash---vim
     |                |-bash
     |                |-bash---pstree
     |                |-bash---ssh
     |                |-bash---mozilla-bin---mozilla-bin---3*[mozilla-bin]
     |                `-gnome-pty-helper
     |-gpm
     |-gweather
     |-kapm-idled
     |-3*[kdeinit]
     |-keventd
     |-khubd
     |-5*[kjournald]
     |-klogd
     |-lockd---rpciod
     |-lpd
     |-mdrecoveryd
```

```
|-6*[mingetty]
|-8*[nfsd]
|-nscd---nscd---5*[nscd]
|-ntpd
|-3*[oafd]
|-panel
|-portmap
|-rhnsd
|-rpc.mountd
|-rpc.rquotad
|-rpc.statd
|-sawfish
|-screenshooter_a
|-sendmail
|-sshd---sshd---bash---su---bash
|-syslogd
|-tasklist_applet
|-vmnet-bridge
|-xfs
`-xinetd-ipv6
```

Az -u és az -a opciók további információkat is adnak.

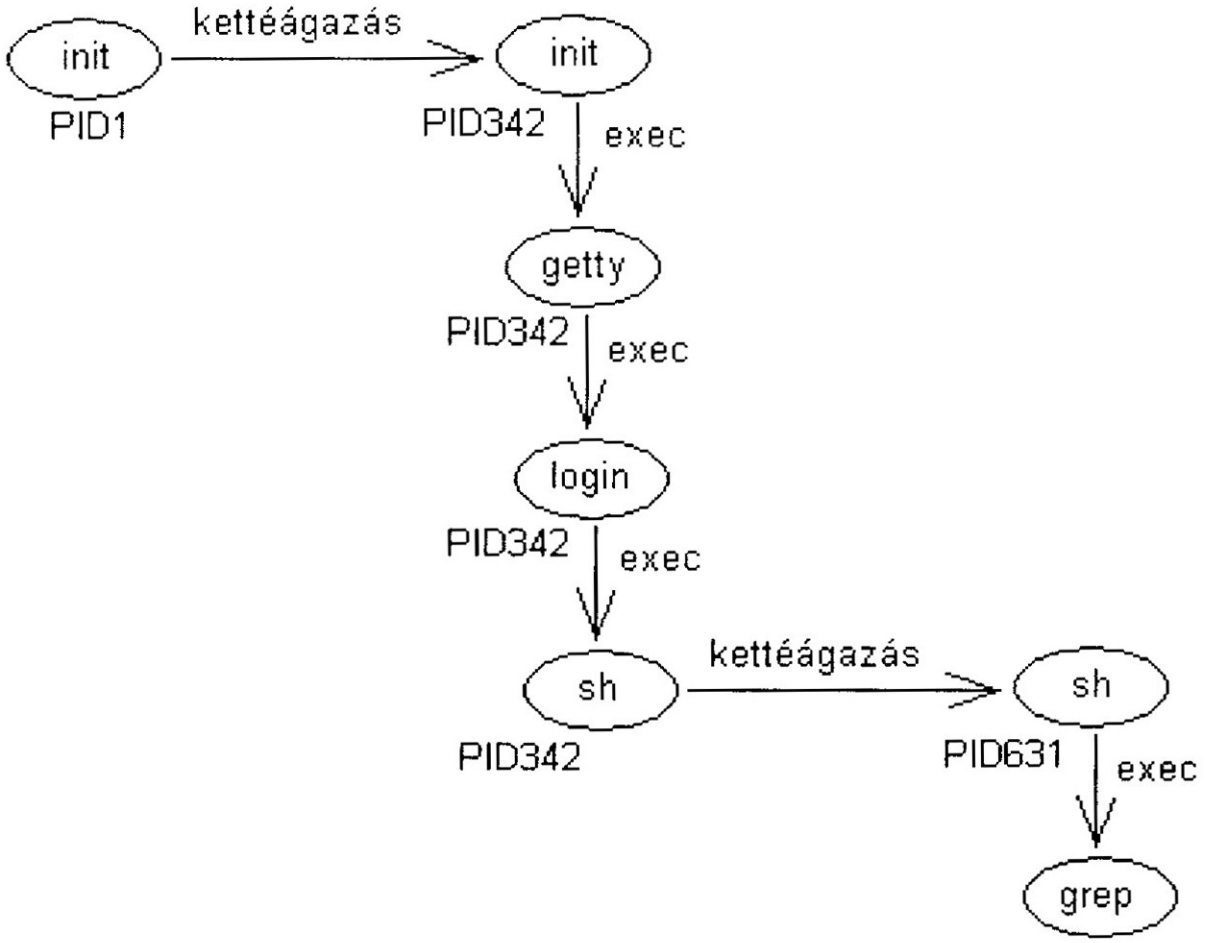
4.1.5. Egy folyamat élete

Egy új folyamat jön létre, ha egy létező folyamat egy exakt másolatot készít magáról. A gyerekprocesszus környezete megegyezik szülője környezetével, egyedül a folyamatazonosítója más. Ezt az eljárást hívjuk kettéágazásnak.

A kettéágazás folyamata után a gyerekprocesszus címhelye az új folyamat-adattal felülíródik. Ez egy exec híváson keresztül megy végbe.

A kettéágazás és az exec mechanizmus egy régi parancsot egy újra cserél, míg a környezet (kimeneti és bemeneti eszközök konfigurációja, környezeti változók, prioritás), mely futtatja az új programot, a régi maradt. Még az első folyamat, az 1-es folyamatazonosítójú init is kettéágazik a rendszerindulás alatt.

Nézzük meg a kettéágazás és az exec mechanizmusának sematikus rajzát! A folyamatazonosító a kettéágazási eljárás után megváltozik:



Számos esetben az init egy folyamat szülője lesz, holott a folyamatot nem az init indította (a példában is látható). Sok program akkor is futhat, amikor a szülője már leállt. Klasszikus példa erre egy ablakozó, ami egy parancsokat fogadó shell-t generáló xterm folyamatot indít. Ezután az ablakozó a gyerekfolyamatot átadja az init-nek. Ezt a mechanizmust használva lehetséges az ablakozók váltása a futó alkalmazások megszakítása nélkül.

Kivételes esetben egy folyamatnak úgy is vége lehet, hogy a szülője nem várja meg a befejeződését (zombi folyamat).

4.1.5.1. Folyamatok befejezése

Ha egy folyamat normál módon áll le (nem lett kilőve és nem szakadt meg), a program az exit status-ával (kilépési állapot) tér vissza a szülőhöz. Ez egy olyan szám, amit a program ad vissza futásának eredményeiről.

A visszaadott kódot értelmezheti a szülő vagy scriptek is. A visszaadott kódok értékei programfüggőek. Ezen információkhoz általában az adott program man oldalaiból jutunk hozzá. A grep parancs -1 visszatérési értéke például azt jelenti, hogy nincs találat, ennek eredménye a No matches found (Nincs találat) üzenet kiírása lesz.

4.1.5.2. Jelek

A folyamatok akkor állnak le, ha kapnak egy jelet. Egy folyamatnak jelet a kill paranccsal küldhetünk. A kill -l a jelek listáját adja. A legtöbb jel a rendszer belső használatára szolgál (esetleg fejlesztőknek programírás közben). Felhasználóként az alábbi szignálokra lehet szükségünk:

Jelnév	Jelszám	Jelentés
SIGTERM	15	A szokásos módon befejezi a folyamatot.
SIGINT	2	Megszakítja a folyamatot. A folyamat figyelmen kívül hagyhatja ezt a jelet.
SIGKILL	9	Megszakítja a folyamatot. A folyamat nem hagyhatja figyelmen kívül ezt a jelet.
SIGHUP	1	Jel démonok számára (újraolvassa a konfigurációs fájlt).

4.1.6. SUID, SGID

Ez a két speciális mód lehetővé teszi a felhasználóknak olyan taszkok futtatását, melyeket normál esetben nem futtathatnának a UNIX-alapú rendszerek szigorú fájljogosultsági rendszere miatt. Ideális esetben a különleges módok a lehető legritkábban vannak csak használva, mivel biztonsági kockázatokat jelenthetnek. A ps például a mindenki számára elérhető /proc-ban tárolt információkat használja, így az érzékeny rendszeradatokat és erőforrásokat a nagyközönség elől elzárja. Korábban, a régebbi UNIX rendszereken a ps programnak olyan fájlokat kellett elérni, mint például a /dev/mem vagy a /dev/kmem, mely hátrányos helyzetet eredményezett a fájlok jogosultságai és tulajdonosai kapcsán:

```
vera:~> ls -l /dev/*mem
crw-r-----    1 root   kmem       1,    2 Sep 18 21:43 /dev/kmem
crw-r-----    1 root   kmem       1,    1 Sep 18 21:43 /dev/mem
```

A régebbi ps verziókban általános felhasználóként lehetetlen volt egy programot elindítani, amíg különleges módokat nem alkalmaztak rá.

Bár nagy általánosságban igaz, hogy nem célszerű használni a különleges módokat, néha mégis szükség van egy SUID használatára. Jó példa erre a jelszavak megváltoztatásának mechanizmusa. A jelszavát természetesen mindenki maga szeretné változtatni, nem pedig a rendszergazdára bízni. Mint tudjuk, a felhasználói nevek és jelszavak listája az /etc/passwd fájlban van, melynek az alábbi hozzáférési jogosultságai és tulajdonosai vannak:

```
ica:~> ls -l /etc/passwd
-rw-r--r--    1 root   root      1267 Jan 27 10:27 /etc/passwd
```

A felhasználóknak pedig ebben a fájlban kell saját információkat megváltoztatniuk. Ez úgy érhető el, hogy a passwd programnak különleges jogokat adunk:

```
akos:~> which passwd
passwd is /usr/bin/passwd

akos:~> ls -l /usr/bin/passwd
-r-s--x--x          1 root   root      13476 Apr 6 05:21 /usr/bin/passwd*
```

Meghívásakor a passwd parancs a rendszergazda hozzáférési jogosultságaival fog futni, így lehetséges, hogy egy egyszerű felhasználó tudja módosítani a jelszófájlt, melynek a rendszergazda a tulajdonosa.

Egy fájlon az SGID módok még ritkábban fordulnak elő, mint az SUID, mert az SGID gyakori következménye a plusz csoportok létrehozása. Egyes esetekben egy elegáns megoldással oldjuk meg a problémát (a szükséges csoportok többnyire a telepítés alatt létrejönnek). Ez a helyzet a write és a wall programokkal, melyek más felhasználók termináljaira való üzenettovábbításra képesek. A write parancs egy üzenetet küld egy egyszerű felhasználónak, a wall pedig az összes kapcsolódott felhasználónak.

Szöveg küldése egy másik felhasználó termináljára vagy grafikus felületére alapesetben nem engedélyezett. A problémát azonban megoldhatjuk egy olyan csoport létrehozásával, mely az összes terminálészközt birtokolja. Ha a write vagy a wall parancsok engedélyezik az SGID-t, a parancsok az adott csoport hozzáférési jogosultásaival fognak futni. Mivel ennek a csoportnak írási jogosultsága van a célterminálra, még egy olyan felhasználó is küldhet üzenetet a terminálra, akinek nincsenek jogosultságai ezen terminál használatára.

A következő példában a laci nevű felhasználó először azt nézi meg (a who parancs használatával), hogy társa melyik terminálra kapcsolódott. Azután küld neki egy üzenetet a write paranccsal. Nézzük meg a write program és a kérdéses felhasználó által lefoglalt terminál felhasználói jogosultságait is!

```
laci:~> which write
write is /usr/bin/write

laci:~> ls -l /usr/bin/write
-rwxr-sr-x 1 root tty 8744 Dec 5 00:55 /usr/bin/write*

laci:~> who
timi     tty1     Aug 25 10:30
timi     pts/1    Aug 25 11:10 (:0)
timi     pts/2    Aug 25 11:11 (:0)
timi     pts/3    Aug 25 11:11 (:0)
laci     pts/0    Aug 22 09:02 (sikisoft.com)

laci:~> ls -l /dev/tty1
crw--w----          1 timi   tty   4,          1 Aug 25 10:30 /dev/tty1
```

```
laci:~> write timi tty1
Timi, ebedelhetunk ma egyutt?
^C
```

Egyértelmű, hogy a tulajdonos felhasználón kívül senkinek sincsenek jogosultságai az eszközön, kivéve a csoporttulajdonost, aki írhat rá.

A timi nevű felhasználó az alábbi üzenetet kapja a képernyőjén:

```
Message from laci@sikisoft.com on ptys/1 at 11:25 ...
Timi, ebedelhetunk ma egyutt?
EOF
```

Egy üzenet fogadása után a terminál a Ctrl + L billentyűkombinációval törölhető. Ha nem szeretnénk üzeneteket kapni (kivéve természetesen a rendszergazdát), használhatjuk a **mesg** parancsot. Ha meg szeretnénk nézni, mely kapcsolódott felhasználók fogadnak üzeneteket másoktól, adjuk ki a **who -w** parancsot.

Vegyük figyelembe, hogy a csoportnevek változhatnak! A csoportelnevezések disztribúciónként eltérhetnek. Egyes disztribúciók más neveket (esetleg más megoldásokat) alkalmaznak.

4.2. Az indulási folyamattól a leállásig

A Linux operációs rendszer egyik nagy előnye a rendszer indításának és leállításának szabadsága, adott programok meghatározott konfigurációval történő indítása. A rendszerbetöltést vezérlő konfigurációk megváltoztathatók, a leállítás pedig szervezett keretek között zajlik.

A rendszerindulási és -leállási folyamatok vezérlésén túl a Linux nyitott jellege egyszerűvé teszi az ezen folyamatokkal kapcsolatos problémák feltárását. Ezen folyamatok alapszintű ismerete minden Linux-felhasználó számára szükséges.

Számos Linux rendszer a lilo-t (*Linux Loader*) használja az operációs rendszer boot-olásához. Érdemesebb azonban az egyszerűbben használható és rugalmasabb GRUB-bal megismerkedni. Mindkét rendszer támogatja a kettős rendszerindítást.

4.2.1. Az indulási folyamat

Mikor egy x86-os számítógép indul, a processzor a rendszermemória végén megkeresi a BIOS-t (Basic Input/Output System, alapvető beviteli/kiviteli rendszer), majd futtatja azt. A BIOS program egy állandó csak olvasható memóriába van írva és mindig elérhető. A BIOS képviseli a legalacsonyabb szintű interfészt a perifériákhoz és a BIOS ellenőrzi a rendszerbetöltési folyamat első részét is.

A BIOS ellenőrzi a rendszert, ellenőrzi a perifériákat, majd megkeresi a rendszerbetöltésre alkalmas meghajtót. Általában először a floppy-meghajtót ellenőrzi (vagy az újabb rendszereken a CD-ROM-ot), majd a merevlemezt. A boot-sorrend a BIOS beállításaitól függ. Ha a rendszer merevlemezére a Linux van telepítve, a BIOS megkeresi az MBR-t (Master Boot Record, ami a merevlemez első szektoránál kezdődik), majd betölti annak tartalmát a memóriába.

Az MBR tartalmazza a GRUB (vagy LILO) rendszerbetöltő betöltésének módját, egy előre beállított operációs rendszer használatával. Ezután az MBR betölti a rendszerbetöltőt, ami folytatja a folyamatot (ha a rendszerbetöltő az MBR-be van telepítve). A Red Hat Linux alapértelmezett konfigurációjánál a GRUB az MBR beállításait felhasználva a rendszerbetöltési opciókat egy menübe rendezve jeleníti meg. Ha egyszer a GRUB feldolgozta a megfelelő instrukciókat az operációs rendszer indításához (akár a parancssorból, akár a konfigurációs fájlból), megtalálja a szükséges indítási fájlt és a vezérlést átadja az operációs rendszernek.

4.2.2. GRUB

A betöltési folyamatot (boot) nevezik közvetlen betöltésnek is, mivel az utasítások közvetlenül töltik be az operációs rendszert (a rendszerbetöltők és az operációs rendszer fő fájljai, mint például a kernel között nincs közbülső kód). Más operációs rendszerek betöltési folyamata ettől eltérő. A Microsoft operációs rendszerei például (a DOS és a Windows rendszerek) mindent felülírnak az MBR-ben, ha az aktuális MBR konfiguráció figyelembe vétele nélkül vannak telepítve. Ez a megoldás a más operációs rendszerek (pl. a Linux) által az MBR-ben tárolt információkat megsemmisíti. A Microsoft operációs rendszerei más kereskedelmi rendszerekhez hasonlóan egy láncolt betöltési folyamattal indíthatók, ha több rendszert is szeretnénk számítógépünkön használni. Ekkor az MBR az operációs rendszert tartalmazó partíció első szektorára mutat, ahol megtalálhatók azok a speciális fájlok, melyek ezen operációs rendszer betöltéséhez szükségesek.

A GRUB mindkét betöltési módszert támogatja, így szinte bármely operációs rendszerrel, a leggyakoribb fájlrendszerekkel és a BIOS-unk által felismert merevlemezekkel használhatjuk.

A GRUB számos tulajdonsággal van felvértezve. Ezek közül a legfontosabbak az alábbiak:

- a GRUB egy igazi parancs-alapú, operációs rendszer előtt környezet x86-os gépekre, ami maximális rugalmasságot nyújt az operációs rendszerek megadott opciókkal történő indítására és a rendszerről történő információgyűjtésre.
- támogatja az LBA (Logical Block Addressing, logikai blokkcímzés) módot, amivel a legtöbb IDE és SCSI merevlemez elérhető. (Az LBA

előtt a meghajtóknak legfeljebb 1024 cilinderük lehetett, a BIOS az ezutáni részeken nem látta a fájlokat.)

- a GRUB konfigurációs fájlja a rendszer minden indulásakor beolvasásra kerül a lemezről, így elkerülhető, hogy minden alkalommal, amikor a rendszerindítási opciókat megváltoztatjuk, felül kelljen írni az MBR-t.

4.2.3. Init

Ha a kernel egyszer betöltődött, megtalálja az init-et az sbin-ben és futtaja azt. Amikor az init elindul, a Linux rendszerünkön automatikusan elinduló összes folyamat szülője vagy nagyszülője lesz. Az első dolog, amit az init tesz, hogy elolvassa az inicializációs fájlját (/etc/inittab). Az itt talált utasítások hatására az init elolvassa a környezet inicializáló konfigurációs scriptjét (ez állítja be a keresési útvonalakat, a swap-pelés kezdetét, ellenőrzi a fájlrendszert stb.). Tulajdonképpen ez az a lépés, ami mindent megtesz, amire a rendszernek szüksége van a rendszer inicializálásához (soros portok inicializálása, az óra beállítása stb.).

Az init ezután folytatja az /etc/inittab fájl olvasását, ami leírja a rendszer egyes futási szintjeinek beállításait és beállítja az alapértelmezett futási szintet. A futási szint nem más, mint folyamatok konfigurációja. Minden UNIX-alapú rendszer különféle folyamat-konfigurációkkal is futhat. Ilyen például az egyszerű felhasználó mód, ami az 1-es futási szint, run level 1 vagy más néven run level S. Ebben a módban csak a rendszergazda csatlakozhat a rendszerhez. Ez lehetővé teszi karbantartási taszkok futtatását a rendszer- vagy a felhasználói adatok megsérülésének veszélye nélkül. Természetesen ebben a módban nem kell felhasználói szolgáltatásokat kínálnunk, így azok itt le vannak tiltva. Egy másik fontos futási szint a 6-os (reboot run level vagy run level 6), ami az összes futó szolgáltatást bezárja, majd újraindítja a rendszert.

A 3-as futási szint a Linux rendszer szöveges módja, az 5-ös pedig a grafikus bejelentkezést és környezetet inicializálja. (A futtatási szintekről később még lesz szó.)

Miután az init megállapította az alapértelmezett futtatási szintet, a rendszer futásához szükséges összes háttérfolyamatot elindítja (az rc könyvtárban nézi meg a futási szintet). Az init az összes kill scriptet futtatja (melyek neve K-val kezdődik) egy stop paraméterrel. Ezután elindítja az összes indulási scriptet (melyek neve S-sel kezdődik) a megfelelő futási szint-könyvtárban, így az összes szolgáltatás és alkalmazás helyesen indul el. Ugyanezeket a scripteket természetesen manuálisan is futtathatjuk a rendszer elindulása után rendszergazdaként bejelentkezve (a webszerver leállítható például a /etc/rc.d/init.d/httpd stop vagy a service httpd stop paranccsal).

Az aktuális indulási és leállási scriptek egyike sem az /etc/rc.d/rc<x>.d-ben van. Az itt levő fájlok ugyanis szimbolikus hivatkozások, melyek az /etc/rc.d/init.d-ben levő aktuális scriptekre mutatnak. Mivel a szimbolikus hivat-

kozás csak egy mutató az aktuális scriptekre, ilyenkor nagyon hasznos, mivel anélkül hozható létre vagy törölhető, hogy bármi hatása is lenne a szolgáltatásokat elindító vagy kilövő aktuális scriptekre. A különféle scriptekre mutató szimbolikus linkek egy meghatározott sorrendben be vannak számozva, így ebben a sorrendben indulnak el. A szolgáltatások indulási és leállási sorrendjét természetesen meg is változtathatjuk. Ehhez nem kell mást tennünk, mint átnevezni az adott szolgáltatást vezérlő scriptre mutató szimbolikus hivatkozást. Ha azt szeretnénk, hogy egy adott szolgáltatás egy másik előtt vagy után induljon vagy álljon le, ugyanazt a számot többször is használhatjuk.

Nézzünk egy példát az /etc/rc5.d tartalmára, ami az 5-ös futási szinten induló szolgáltatások listáját tartalmazza:

```
[csilla@disz /etc/rc5.d] ls
K15httpd@       K45named@       S08ipchains@    S25netfs@       S85gpm@
K16rarpd@       K46radvd@       S08iptables@    S26apmd@        S90crond@
K20nfs@         K61ldap@        S09isdn@        S28autofs@      S90xfs@
K20rstatd@      K65identd@      S10network@     S30nscd@        S95anacron@
K20rusersd@     K74ntpd@        S12syslog@      S55sshd@        S95atd@
K20rwalld@      K74ypserv@      S13portmap@     S56rawdevs@     S97rhnsd@
K20rwhod@       K74ypxfrd@      S14nfslock@     S56xinetd@      S99local@
K25squid@       K89bcm5820@     S17keytable@    S60lpd@
K34yppasswdd@   S05kudzu@       S20random@      S80sendmail@
```

Miután az init megállapította az alapértelmezett futási szintet, az /etc/inittab script elágazik egy getty folyamatba minden virtuális konzolhoz (szöveges módban a bejelentkező prompt). A getty megnyitja a tty sorokat, beállítja módjukat, kiírja a bejelentkező promptot, bekéri a felhasználó nevét, majd ha ezt beírtuk, elkezd egy bejelentkeztetési folyamatot. E lehetőséggel azonosíthatjuk magunkat a rendszer számára, ha használni szeretnénk. Alapesetben a legtöbb rendszer 6 virtuális konzolt kínál, de ez állítható (az inittab fájlban meg tudjuk nézni).

Az init az /etc/inittab-ból tudja meg továbbá, hogy mi történjen, ha egy felhasználó a konzolon leüti a Ctrl + Alt + Del billentyűkombinációt. Mivel jobb, ha a rendszer helyesen áll le és indul újra, mintha azonnal leállna, az init-tel itt közölhető, hogy a fenti billentyűkombináció hatására például az /sbin/shutdown -t3 -r now parancs hajtódjon végre. Az /etc/inittab közli az init-tel azt is, hogy mi a teendő áramszünet esetén, ha rendszerünk szünetmentes tápegységgel van felszerelve.

Az 5-ös futási szinten az /etc/inittab futtat egy /etc/X11/prefdm nevű scriptet. A predfm script az /etc/sysconfig/desktop könyvtár tartalma alapján futtatja az előnyben részesített ablakozót. Ez általában a gdm (ha a GNOME-ot futtatjuk) vagy kdm (ha a KDE-t használunk), de ezek vegyíthetők is. Az xdm-et pedig a standard X telepítés tartalmazza.

Az /etc/sysconfig könyvtár számos funkció és szolgáltatás bejegyzéseit tartalmazza, ezek a rendszerbetöltéskor kerülnek beolvasásra. (Ez az utolsó rész Linux disztribúciónként eltérhet valamelyest.)

A grafikus felhasználói környezet mellett számos más szolgáltatás is elindulhat. Ha minden rendben van, a rendszerbetöltési folyamat végeztével egy bejelentkezői promptot vagy képernyőt kell látnunk.

Megjegyzendő, hogy a korábbiakban a SysV init-et néztük meg x86 alapú számítógépeken. Egyes eljárások más architektúrákon és disztribúciókkal eltérhetnek. Más rendszerek BSD-stílusú init-et használnak, ahol az indulási fájlok többrészes /etc/rc.d/rc<*SZINT*>.d könyvtárakban vannak. Az is előfordulhat, hogy rendszerünk az /etc/init.d-t használja az /etc/rc.d/init.d helyett.

4.2.4. Init futtatási szintek

Az elgondolás mögött, miszerint különböző szolgáltatások különböző futási szinteken működnek, lényegében az a tény húzik meg, hogy a különféle rendszerek különféle módon használhatók. Egyes szolgáltatások nem használhatók addig, míg a rendszer egy bizonyos állapotban (vagy módban) van.

Egyes esetekben a rendszert egy alacsonyabb módban szeretnénk üzemeltetni. Erre lehet példa lemezproblémák rögzítése az 1-es szinten, ekkor más felhasználók értelemszerűen nem lehetnek a rendszerben. Ilyen és ehhez hasonló esetekben egy magasabb rendszermódtól függő futó szolgáltatásokkal nem kell foglalkozni, hiszen úgy sem megfelelően működnek. Ha már minden szolgáltatás elindul, mikor a megfelelő futási szintje elérhető, az indulási folyamat biztosított és gyorsan válthatunk a gép módjai között anélkül, hogy aggódnunk kellene, melyik szolgáltatást kell manuálisan elindítanunk vagy leállítanunk.

Az elérhető futási szinteket általában az /etc/inittab eleje tartalmazza:

```
#
# inittab This file describes how the INIT process should set up
#          the system in a certain run-level.

# Default runlevel. The runlevels are:
#    0 - halt (Do NOT set initdefault to this)
#    1 - Single user mode
#    2 - Multiuser, without NFS
#        (The same as 3, if you do not have networking)
#    3 - Full multiuser mode
#    4 - unused
#    5 - X11
#    6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:
...
```

A 2-es és a 4-es futási szinteket tetszés szerint nyugodtan konfigurálhatjuk. Sok felhasználó ezeket a futási szinteket úgy konfigurálja, hogy a standard 3-as és 5-ös futási szinteket egyedül hagyja. Ezzel a megoldással gyorsan alkalmazni tudják saját konfigurációjukat anélkül, hogy zavarnák a standard futási szintek működését.

Ha gépünk nem indul, mert az /etc/inittab beállításai helytelenek, esetleg nem tudunk bejelentkezni egy hibás /etc/passwd fájl miatt (ne adj Isten, elfelejtettük jelszavunkat), boot-oljunk egyszerű felhasználói módban.

Ha szöveges módban dolgozunk, mert nem kaptunk grafikus bejelentkező képernyőt gépünk konzolán, egyszerűen felválthatunk a 7-es (vagy magasabb számú) konzolra a grafikus bejelentkezéshez. Ha nem ez a helyzet áll fenn, ellenőrizzük az aktuális futási szintet a who -r paranccsal. Ha ez 5-től eltérő, valószínűleg a rendszer alapesetben nem grafikusan indul. Ne felejtsük, hogy a futási szintek között az init paranccsal válthatunk. Egy szöveges konzolról grafikus konzolra (vagy fordítva) váltás nem jelent futási szint-váltást!

4.2.4.1. Eszközök

Az /sbin-ben levő chkconfig egy egyszerű parancssori eszköz az /etc/rc.d/init.d könyvtárhierarchia karbantartására. Így a rendszergazdák mentesülnek az /etc/rc.d könyvtár rengeteg szimbolikus hivatkozásának közvetlen manipulálása alól.

Számos rendszer kínálja az ntsysv eszközt, ami egy szöveg alapú interfész. Egyszerűbben használható, mint a chkconfig parancssori interfésze.

SuSE Linux alatt megtalálható a yast és az insserv eszköz is.

A Mandrake disztribúcióban a könnyű konfigurálást a DrakConf segíti, mely számos más tulajdonsága mellett a 3-as és az 5-ös futási szint közötti átváltást is lehetővé teszi.

A legtöbb disztribúció grafikus felhasználói interfészt is kínál a folyamatok konfigurálásához.

Az összes ilyen eszközt rendszergazdaként kell futtatnunk. A rendszergazda természetesen manuálisan is létrehozhatja a megfelelő hivatkozásokat minden egyes futási szint könyvtárában az egyes szolgáltatások adott szinten történő elindításához vagy leállításához.

4.2.5. Leállítás

A UNIX és az azon alapuló rendszerek folyamatos futásra vannak tervezve. Ha mégis le szeretnénk állítani (pl. otthoni felhasználás), a shutdown parancsot kell kiadnunk. A leállási folyamat után a -h opcióval megáll a rendszer, az -r opcióval újraindul.

A reboot és a halt parancsok meghívhatják a shutdown-t (ha fut), ha a rendszer az 1-5 futási szintek valamelyikében van, a rendszer teljesen leáll (nem minden Linux verzió támogatja).

Ha számítógépünk nem kapcsolja ki magát, ne kapcsoljuk ki mi sem, amíg nem jelenik meg az üzenet, hogy a rendszer befejezte a leállást, mivel a rendszernek időre van szüksége az összes partíció lekapcsolásához. Ha türelmetlenkedünk, könnyen adatvesztés lehet a vége.

4.3. Munka a folyamatokkal

4.3.1. A helyi rendszergazda folyamata

Rendszererőforrások menedzselésekor (beleértve a folyamatokat is) van a helyi rendszergazdának egy taszkja. Egy egyszerű felhasználónak persze erről nem is kell tudnia, különösen, hogy mindenkinek a saját folyamatai az érdekesek.

4.3.2. Meddig tart?

A Bash a beépített time parancsot kínálja annak kijelzésére, meddig tart egy parancs futtatása. Ez a mérés egész pontos és bármely parancsra használható. Nézzünk egy példát, ahol egy könyv elkészítése mintegy másfél percig tart:

```
jeno:~/konyv/sajat> time make
Output written on konyv.pdf (198 pages, 1539362 bytes).
Transcript written on konyv.log.

real    1m41.056s
user    1m31.190s
sys     0m1.880s
```

Az /usr/bin-ben levő GNU time parancs (a shell beépített time parancsával ellentétben) több információt jelenít meg. A GNU time parancs megmutatja a kilépési állapotot és a teljes eltelt időt is. Eredménye a fenti esetben az alábbihoz hasonló lenne:

```
jeno:~/konyv/sajat> /usr/bin/time make
Output written on konyv.pdf (198 pages, 1539362 bytes).
Transcript written on konyv.log.
Command exited with non-zero status 2
88.87user 1.74system 1:36.21 elapsed 94%CPU
                                    (0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (2192major+30002minor)pagefaults 0swaps
```

4.3.3. Teljesítmény

A felhasználóknál a teljesítmény parancsok gyors futtatását jelenti. Egy rendszergazdának azonban sokkal többet jelent, hiszen neki kell optimatizálnia a rendszer teljesítményét az egész rendszerre, beleértve a felhasználókat, az összes programot és a démonokat is. A rendszerteljesítményt ezer apró dolog befolyásolja, melyek a time paranccsal nem mérhetők:

- előfordulhat, hogy a program futása helytelenül van megírva, esetleg nem megfelelően használja a számítógépet,
- a lemezekkel, a vezérlőkkel, a kijelzéssel, a különféle interfészekkel nem számol,
- a munkát befolyásolhatja a távoli rendszerek elérhetősége (a hálózati teljesítmény) is,
- ha nagyon sok az egyszerre dolgozó felhasználó a rendszeren,
- stb.

4.3.4. Mit tehet a felhasználó?

Egy nagy környezetben igencsak lelassulhatnak a munkafolyamatok. Ha sok környezeti változónk van beállítva (a shell változók helyett), hosszú, nem optimatizált keresési utakat használunk (a path környezeti változó beállítása hibás) és így tovább, a rendszernek több időre van szüksége az adatok megkeresésére és beolvasására.

A grafikus felületek igazi „processzor-gyilkosok". Egy igazán szép grafikus kezelőfelületnek ára van, még akkor is, ha ingyen töltöttük le a hálózatról, mivel végtelen számú része és kellékei vannak.

4.3.4.1. Prioritás

Egy munka prioritása vagy fontossága az úgynevezett kedvességi számmal mérhető. Egy magas kedvességi számmal rendelkező program „barátságos" más programokkal, más felhasználókkal és a rendszerrel (nem fontos munka). Minél kisebb a kedvességi szám, annál fontosabb a feladat elvégzése és annál kevesebb erőforrást akar megosztani.

Egy program „kedvesebbé" változtatása kedvességi számának növelésével csak az olyan folyamatoknál hasznos, melyek sok CPU-időt használnak (pl. fordítók, matematikai alkalmazások stb.). Azok a folyamatok, melyek mindig sok I/O időt használnak, a rendszertől automatikusan magas prioritást (alacsonyabb kedvességi számot) kapnak. Egy billentyűzet-bemenet feldolgozása például mindig a legnagyobb prioritást kapja a rendszerben.

Egy program prioritásának definiálása a nice paranccsal lehetséges.

A legtöbb rendszer kínálja a BSD-s renice parancsot is, ami lehetővé teszi egy futó parancs kedvességének megváltoztatását.

Nem jó ötlet a nice és a renice használata egy interaktív programmal vagy egy előtérben futó alkalmazással. Ezen parancsokat általában a rendszergazdák használják.

4.3.4.2. CPU-erőforrások

Minden Linux rendszeren egyidejűleg több program is szeretné használni a processzort, még akkor is, ha egyedüli felhasználók vagyunk a rendszerben. Minden programnak egy adott CPU-időre van szüksége futásához. Előfordulhat, hogy nincs szabad CPU-idő, mert a processzor kihasználtsága nagyon magas. Bár az uptime parancs meglehetősen pontatlan (csak átlagokat jelenít meg, nekünk kell tudnunk, mi számít normál értéknek), azért nagyon hasznos tud lenni. Néhány irányadó ötlet a CPU optimális kihasználására:

- akkor futtassunk nagy igényű programokat, mikor a rendszer kevésbé terhelt (pl. éjszaka). Ekkor használható ki jól az ütemezés lehetősége (lásd később).
- előzzük meg a rendszer fölösleges munkáját. Állítsuk le azokat a démonokat és programokat, melyeket nem használunk, használjuk a locate parancsot a find helyett stb.
- a nagyobb programokat kisebb prioritással futtassuk.

Ha ezek közül egyik sem használható a mi esetünkben, újabb processzorra van szükségünk.

4.3.4.3. Memória-erőforrások

Ha az aktuálisan futó folyamatok több memóriát igényelnek, mint amennyivel a rendszer fizikailag rendelkezik, a Linux működése nem lesz hibás. Ekkor ugyanis elkezd lapozni (más néven swap-pelni), vagyis a folyamatok a lemezen levő speciális részt vagy a swap-helyet használják memóriaként. A fizikai memória tartalmát (futó programok részei vagy teljes programok swappelés esetén) a lemezre mozgatják, így a fizikai memória több folyamat kezelésére lesz képes. Az eljárás hátránya, hogy a rendszer sebessége jelentősen romlik, mivel a lemez elérése sokkal több időt vesz igénybe, mint a memóriáé.

A top paranccsal jeleníthetjük meg a használatban levő memóriát és swapet. A glibc-t használó rendszerek a memusage és a memusagestat parancsokkal vizuálisan is meg tudják jeleníteni a memóriahasználatot.

Ha úgy találjuk, hogy rengeteg memória és swap-hely van használatban, az alábbi lehetőségekkel próbálkozhatunk:

- kilőjük, leállítjuk vagy „kedvesebbé tesszük" azokat a programokat, melyek sok memóriát használnak,
- a rendszerben memóriabővítést végzünk (egyes esetekben a swap-helyet is növelhetjük),
- egyéb módon növeljük a rendszer teljesítményét.

4.3.4.4. Ki- és bemeneti erőforrások

Bár az I/O korlátozások nagy gondot okozhatnak a rendszergazdáknak, a Linux operációs rendszer meglehetősen kevés eszközt kínál az I/O teljesítmény mérésére. A ps, a vmstat és a top eszközök adnak némi jelzést arról, hány program vár I/O-ra. A netstat hálózati interfész-statisztikákat jelenít meg. Nincsenek azonban olyan programok, melyek mérnék a rendszer terheltségének következményeit az I/O-ra. Az iostat parancs egy rövid áttekintést ad az általános I/O használatról. Számos grafikus program létezik, mely ezen parancsok kimenetét érthető formában jeleníti meg.

Minden eszköznek megvannak a maga problémái, de alapjában véve két fő dolog határozza meg az I/O teljesítmény szűk keresztmetszetét: a lemezek elérési ideje és a hálózati interfészek sávszélessége.

A lemezek I/O problémái:

- a folyamatonkénti adatátviteli sebesség túl gyenge, már egy egyszerű folyamat olvasási és írási sebessége sem kielégítő
- az összesített adatátviteli sebesség túl gyenge
  Ezt a hibatípust nagyon nehéz detektálni és legtöbbször új hardverelemeket kell beszereznünk, az adatfolyamokat újra el kell osztanunk a buszok, a vezérlők és a lemezek között.
  A lemezproblémák csökkentésének egyik eszköze a bemeneti és kimeneti műveletekre optimatizált RAID konfiguráció. Ekkor megtarthatjuk a régi hardvereszközöket is. Egy további lehetőség a buszok, vezérlők és lemezek frissítése.
  Ha nem a túlterheltség okozza a lemez I/O hibáját, előfordulhat, hogy érintkezési gondok vannak a számítógépünkben. Ellenőrizzük a csatlakozókat, az érintkezőket.

A hálózatok I/O problémái:

- a hálózat túlterheltsége: a hálózaton szállított adat mennyisége nagyobb, mint a a hálózat kapacitása. Ez minden hálózattal kapcsolatos (azzal dolgozó) taszk lassú futását eredményezi az összes felhasználónál. A probléma a hálózat megtisztításával (a nem használt protokollok és szolgáltatások letiltása) vagy újrakonfigurálásával (alhálózatok használata, a hub-ok kicserélése switch-ekre stb.) oldható meg.
- hálózat-integritási problémák: akkor fordulnak elő, ha az adatok nem megfelelően lettek továbbítva. Ez a típusú probléma csak úgy oldható meg, hogy leválasztjuk a hibás elemet és kicseréljük.

4.3.4.5. Felhasználók

A felhasználóknak több típusa van az erőforrás-felhasználás szempontjából:

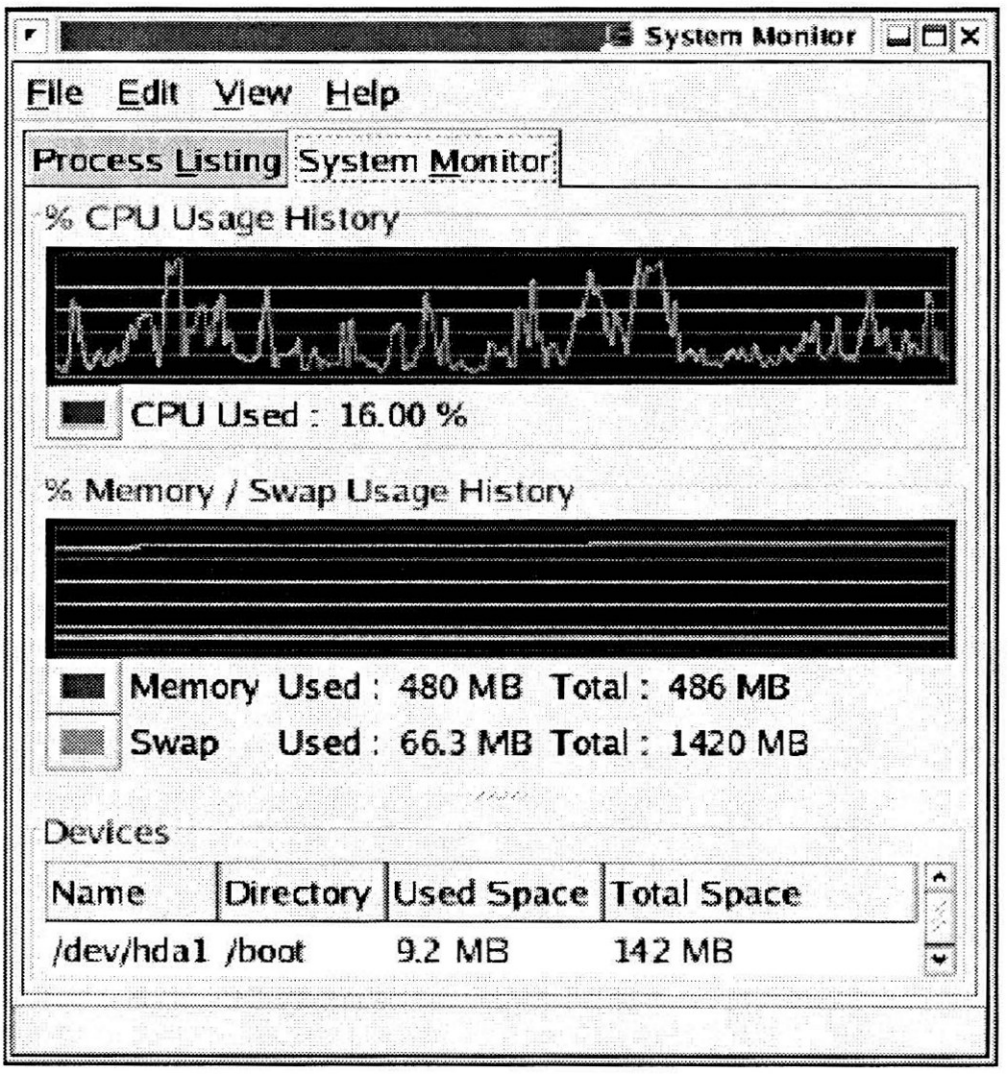
- olyan felhasználó, aki (sok) kisebb programot futtat (pl. kezdő Linux felhasználó)

- olyan felhasználó, aki relatíve kevés, de nagy igényű programokat futtat (pl. számításokat végző, szimuláló, emuláló stb. programok nagy memóriaigénnyel; nagy adatfájlokkal dolgozó programok)
- olyan felhasználó, aki kevés programot futtat, de sok CPU-időt használ (pl. fejlesztők)

Látható, hogy a rendszerkövetelmények igencsak eltérnek az egyes felhasználói típusoknál és nehéz minden igényt kielégíteni. Ha egy többfelhasználós rendszerben vagyunk, hasznos (és néha mókás) más felhasználók szokásait és a rendszer működését megismerni, hogy a lehető legtöbbet tudjuk kihozni belőle saját igényeink kielégítésére.

4.3.4.6. Grafikus eszközök

A grafikus környezetben számos teljesítményt figyelő program áll rendelkezésre. Vessünk egy pillantást például a Gnome rendszermonitorra:



A CPU-használat mellett a memória és a swap felhasználásáról is kapunk információkat, sőt a merevlemezek használt és teljes kapacitását is megtudhatjuk.

A tálcára a lemez vagy a memória állapotát, a rendszer terheltségét tükröző ikonokat telepíthetünk. A rendszerterheltség vizsgálatára kitűnő programocska az xload is.

4.3.4.7. Folyamataink megszakítása

Nem privilegizált felhasználóként csak saját folyamatainkat befolyásolhatjuk. Már láttuk, hogyan jeleníthetjük meg egy adott felhasználó folyamatait és ismerjük a lehetséges megszorításokat is.

Ha észrevesszük, hogy valamely folyamatunk túl sok rendszererőforrást emészt fel, két dolgot tehetünk:

- a folyamatot kevesebb erőforráshoz engedjük anélkül, hogy megszakítanánk;
- leállítjuk a folyamatot.

Abban az esetben, ha szeretnénk, hogy folytatódjon a folyamat (virtuális gép, fordító, emulátor stb.), de a rendszer másik processzusainak is szeretnénk esélyt adni, kedvesebbé tehetjük a folyamatot. Ezt a nice és a renice programok nélkül is megtehetjük: a top használata például egy egyszerű megoldás lehet a prioritás csökkentésére.

Azonosítsuk be a folyamatot az NI oszlopban, majd nézzük meg a prioritást, mely ebben az esetben valószínűleg negatív. Üssünk egy r billentyűt, majd adjuk meg annak a folyamatnak a folyamatazonosítóját, melyet kedvesebbé szeretnénk tenni, majd végül a kedvességi számot. Ha például 20-at adunk meg, akkor ettől kezdve az adott folyamat legfeljebb a CPU-idő 1/5-ét használhatja.

Ha le szeretnénk állítani egy processzust, mert az megakadt vagy nagyon fogyasztja a rendszer erőforrásait (pl. az I/O erőforrásokat), akadályozza a fájllétrehozást stb., használjuk a kill parancsot (kill = ölni). Ha van rá lehetőség, a lehető leggyengébb módon próbáljuk meg kilőni a folyamatot: először próbáljunk meg küldeni neki egy SIGTERM jelet. Ha ez nem működik, durvább eszközhöz folyamodunk:

```
bea:~> ps -ef | grep mozilla
bea      24354          1 0 Mar28 ?     00:32:03 /usr/lib/mozilla-1.4.1/mozilla-

bea:~> kill -15 24354
```

A bea nevű felhasználó leállította a Mozilla böngészőjét, mert az megakadt.

Egyes folyamatoknál nem ilyen egyszerű a helyzet. Ha van időnk, küldhetünk SIGINT jelet. Ha nem érjük el a kívánt hatást, kipróbálhatjuk a legerősebb jelet, a SIGKILL-t is. Ha bea böngészője lefagyott (ami Linux alatt nem túl gyakori), azt így is megállíthatja:

```
bea:~> ps -ef | grep mozilla
bea      25468          1 0 Mar28 ?     00:14:05 /usr/lib/mozilla-1.4.1/mozilla-
```

```
bea:~> kill -9 25468

bea:~> ps -ef | grep 25468
bea      2523 31435  0 17:56 pts/4    00:00:00 grep 25468
```

Ilyenkor érdemes ellenőrizni, hogy a folyamat tényleg leállt-e. Ehhez a grep szűrőt használhatjuk a folyamatazonosítón. Ha ekkor csak a grep folyamatot kapjuk vissza, biztosak lehetünk benne, hogy sikeresen leállítottuk a folyamatot.

A nehezen kilőhető folyamatok közé tartozik a shell-ünk is. Ez azonban nem feltétlenül jelent rosszat. Gondoljunk csak bele: ha nem így lenne, minden alkalommal, mikor véletlenül leütjük a Ctrl + C billentyűkombinációt a parancssorban, elveszítenénk a shell-t, mivel ez egyenértékű egy SIGINT jel küldésével.

A grafikus környezetben egyszerűen használható program az xkill. A parancs begépelése után csak ki kell jelölnünk a bezárandó alkalmazás ablakát. Ez a megoldás meglehetősen veszélyes, mivel alapból a SIGKILL jelet küldi az alkalmazásnak, így csak akkor használjuk, ha az alkalmazás lefagyott.

4.4. Ütemezés

4.4.1. Üresjárat kihasználása

Egy Linux rendszer kihasználtsága elsősorban az irodai munkák idején a legnagyobb (ha irodában használják a rendszert). Ha azonban egy szerverszobában vagy otthon használjuk is a Linux operációs rendszert, a kora reggeli, a késő esti, az éjszakai és a hétvégi időszakban többnyire gyenge kihasználtságú. Ezeket az időszakokat is kihasználhatjuk azonban, és ez a megoldás jóval olcsóbb, mintha olyan komoly felszereltségű számítógépet vásárolnánk, aminek az összes munka egyidejű elvégzése sem okoz gondot.

A késleltetett futtatásnak három fő típusa van:

1. Beiktatunk egy kis szünetet a futtatásba. Ezt a sleep parancs használatával tehetjük meg (sleep = aludni).
2. Egy parancsot egy megadott időben futtatunk. Erre szolgál az at parancs.
3. Egy parancsot óránként, naponta, hetente, havonta rendszeresen futtatunk a cron felhasználásával.

4.4.2. A sleep parancs

Az egyik legegyszerűbb parancs a sleep (még az Info oldala is az egyik legrövidebb). Az egyetlen, amit a sleep parancs eredményez, az a várakozás. Alapesetben a várakozás időtartama másodpercekben van megadva.

Nézzünk néhány példát a sleep alkalmazására:

- Ha elfelejtettünk egy fontos dolgot, például egy találkozót, ideiglenesen felfüggeszthetjük munkánkat, egy fél órás ebédszünetet beiktatva:

(sleep 1800; echo "Ebedido") &

- Ha valamely ok miatt nem használhatjuk az at parancsot. Például munkaidőnk lejárt, hazaindulnánk, de még lenne mit csinálni, csak valaki lefogja a rendszer erőforrásait:

  (sleep 10000; programunk) &

  Ekkor ügyeljünk a kijelentkezésre, illetve az asztal lokkolására.

- Ha rengeteg szöveget vagy óriási fájlokat szeretnénk nyomtatni, de szeretnénk másokat is a nyomtató közelébe engedni:

  lp rengeteg_szoveg; sleep 900; lp oriasi_fajl; sleep 900;
  lp masik_oriasi_fajl

  A nyomtatásról a későbbiekben még részletesen lesz szó.

A programozók gyakran használják a sleep parancsot egy script vagy program adott ideig történő felfüggesztésére.

4.4.3. Az at parancs

Az at parancs megadott időben futtat parancsokat, alapértelmezett shellünket felhasználva (hacsak másként nem rendelkezünk).
Az at parancs opciói meglehetősen felhasználóbarátak:

```
pityu@otthon:~> at tomorrow + 3 days
warning: commands will be executed using (in order) a) $SHELL
        b) login shell c) /bin/sh

at> cat reports | mail fonok@cegem
at> <uzenet>
job 1 at 2005-05-05 17:24
```

A Ctrl + D billenytűkombináció kilép az at-ből és legenerálja az <uzenet>-et.
A pityu egy rendhagyó dolgot művel két parancs kombinálásával:

```
pityu@otthon:~> at 0237
warning: commands will be executed using (in order) a) $SHELL
        b) login shell c) /bin/sh
at> cd uj_programok
at> ./configure; make
at> <uzenet>
job 2 at 2005-08-25 02:00
```

Az -m opció a munka befejezésével egy üzenetet küld a felhasználónak vagy értesíti, ha a munka nem készült el. Az atq parancs listázza ki a feladatokat. A parancsot a feladatok előtt hajtsuk végre, nehogy ugyanabban az időben induljanak, mint mások. Az atrm paranccsal kiiktathatjuk az ütemezett feladatokat, ha meggondoltuk magunkat.

Jó ötlet különleges futási időpontokat beállítani, mert a rendszer munkafolyamatai gyakran kerek órákban indulnak. Rendszerfolyamataink például gyakran futnak pontosan hajlani egykor, így ezt az időpontot (0100) beállítva könnyen előfordulhat, hogy rendszerünk inkább lelassul, nemhogy felgyorsulna. Annak megelőzésére, hogy minden folyamat ugyanabban az időpontban induljon el, használhatjuk a batch parancsot. A batch segítségével a folyamatokat sorba rendezhetjük és így viszonylag kiegyensúlyozott erőforrás-felhasználást tudunk megvalósítani.

4.4.4. Cron, crontab

A cron rendszert a cron démon menedzseli. Információkat kap arról, hogy mely programoknak és mikor kell futniuk a rendszer- vagy felhasználói crontab-ból. Csak a rendszergazda éri el a rendszer crontab-okat. Minden felhasználó csak a saját crontab-jaihoz férhet hozzá. Egyes rendszereken egyes felhasználók nem érhetik el a cron lehetőségét.

A rendszer indulásakor a cron démon a /var/spool/cron/-ban keres crontab-okat, melyek az /etc/passwd azonosítói alapján vannak elnevezve. Ezután keres a /etc/cron.d-ben és a /etc/crontab-ban is, majd minden percben felhasználja ezt az információt annak ellenőrzésére, hogy van-e valami tennivaló. A crontab fájl tulajdonosaként futtat parancsokat, majd kiírja az összes program-kimenetet a felhasználónak.

A Vixie cron-t használó rendszereken az óránként, naponta, hetente, havonta előforduló folyamatok az /etc külön könyvtáraiban vannak tárolva (ellentétben a standard UNIX-os cron-nal, ahol az összes taszk egyetlen nagy fájlban van).

Példa egy Vixie crontab fájlra:

```
[root@disz /etc]# more crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
# commands to execute every hour
01 * * * * root melo /etc/cron.hourly
# commands to execute every day
02 4 * * * root melo /etc/cron.daily
# commands to execute every week
```

```
22 4 * * 0 root melo /etc/cron.weekly
commands to execute every month
42 4 1 * * root melo /etc/cron.monthly
```

Néhány változó beállítása után az aktuális ütemezés van megadva, feladatonként egy-egy sorban. Az első mező a perceket tartalmazza (0-tól 59-ig), a második a futtatás óráját definiálja (0-tól 23-ig), a harmadik a hónap napja (1-től 31-ig), ezt követi a hónap (1-től 12-ig), végül a hét napja (0-tól 7-ig, a 0 és a 7 egyaránt a vasárnapot jelenti). A mezőkben a csillag karakterek a teljes elfogadható sorozatot reprezentálják. Felsorolások is használhatók. Egy feladat hétfőtől péntekig történő futtatásához az utolsó mezőben 1-5 értéket adjunk. Ha egy feladatot hétfőn, szerdán és pénteken szeretnénk futtatni, az 1,3,5 értékeket adjuk meg.

Ezután jön a felhasználó, aki futtatja az utolsó oszlopban felsorolt folyamatokat. A fenti példában a rendszergazda a melo programot futtatja szabályos időközönként a megfelelő könyvtár-opciókkal. Ezekben a könyvtárakban az ütemezett időben futtatandó aktuális feladatok shell scriptekként vannak tárolva, mint például az alábbi script naponta futtatja a locate parancshoz felhasznált adatbázis frissítését:

```
endre@dve cron.daily]$ cat slocate.cron
#!/bin/sh
renice +19 -p $$ >/dev/null 2>&1
/usr/bin/updatedb -f "nfs,smbfs,ncpfs,proc,devpts" -e \
"/tmp,/var/tmp, /usr/tmp,/afs,/net"
```

A felhasználók crontab-jaikat általában biztonságos módon, a crontab -e parancs használatával szerkesztik. Ezzel elkerülhető, hogy a felhasználó véletlenül kétszer nyissa meg a crontab fájlját. Az alapértelmezett szerkesztő a vi, de használhatunk bármilyen más szövegszerkesztőt is (gvim, gedit stb.).

Mikor kilépünk, a rendszer tudatja velünk, hogy egy új crontab lett telepítve.

Miután megadunk egy új ütemezett taszkot, nem kell újraindítanunk a cron démont a változások életbe lépéséhez.

5. I/O ÁTIRÁNYÍTÁS

5.1. Standard bemenet és kimenet

A legtöbb Linux parancs valamilyen bemenetet (fájlt vagy a parancs egy másik attribútumát) olvas be és kimenetet ír ki. Alapértelmezésben a bemenet a billentyűzetről érkezik, a kimenet pedig a képernyőn jelenik meg. Szakszerűbben fogalmazva azt is mondhatjuk, hogy billentyűzetünk a standard bemeneti (standard input, stdin) eszköz, a képernyő vagy egy terminálablak pedig a standard kimenet (standard output, stdout) eszköz.

Mivel a Linux egy rugalmas rendszer, nem feltétlenül szükséges ezeket az alapbeállításokat alkalmazni. Egy nagyobb környezetben például a nyomtató is lehet az alapértelmezett kimeneti eszköz.

5.2. Átirányítás-operátorok

5.2.1. A kimenet átirányítása

Néha szükség lehet rá, hogy egy parancs kimenetét egy fájlba tegyük, esetleg egy másik parancsot szeretnénk felhasználni egy parancs kimenetével. Az ilyen és ehhez hasonló eseteket nevezzük a kimenet átirányításának. Az átirányítás a > (nagyobb, mint) karakterrel vagy a | (pipe) operátorral történhet, melyek egy parancs standard kimenetét egy másik parancs standard bemenetének küldik.

A korábbiakban már megismerkedtünk a cat paranccsal. A cat képes összefűzni a fájlokat és a standard kimeneten megjeleníti azokat. A fájl kimenetének átirányításával a megadott nevű fájl létrejön (ha már létezett, akkor felülíródik!):

```
agi:~> cat vers1
Ki tiltja meg, hogy elmondjam, mi bantott

agi:~> cat vers2
 hazafele menet?
agi:~> cat vers1 vers2 > vers3

agi:~> cat vers3
```

Ki tiltja meg, hogy elmondjam, mi bantott
hazafele menet?

József Attila Levegőt! című versének első két sora egyetlen fájlba került.

Ha „semmit" irányítunk át egy létező fájlra, az egyenértékű a fájl kiürítésével:

```
agi:~> ls -l list
-rw-rw-r--        1 agi    agi       223 Apr 5 17:56 list

agi:~> > list

agi:~> ls -l list
-rw-rw-r--        1 agi    agi         0 Apr 6 12:03 list
```

Ezt a folyamatot hívjuk csonkolásnak.

Ha ugyanezt az átirányítást egy még nem létező fájlra végezzük el, létrejön egy új üres fájl az adott névvel:

```
agi:~> ls -l ujlista
ls: ujlista: No such file or directory

agi:~> > ujlista

agi:~> ls -l ujlista
-rw-rw-r--        1 agi    agi         0 Apr 6 12:07 ujlista
```

A | jellel egy szót megkereshetünk egy szövegben. Ehhez mintákat adunk meg:

```
grep minta1 fajl | grep -v minta2
```

Ha oldalanként szeretnénk megtekinteni egy könyvtár kimenetét, az alábbi parancsot adjuk ki:

```
ls -la | less
```

Ha egy fájlt keresünk egy könyvtárban, az alábbi szintaktikát alkalmazzuk:

```
ls -l | grep a_fajlnev_resze
```

5.2.2. A bemenet átirányítása

Vannak olyan esetek is, amikor azt szeretnénk, hogy egy olyan fájl legyen egy parancs bemenete, mely alapból nem fogad fájlt opcióként, a bemenetet kell átirányítanunk. Erre használható a < (kisebb, mint) operátor.
Küldjünk el egy fájlt valakinek a bemenet-átirányítás felhasználásával!

```
viki:~> mail bazsi@siki-soft.hu < talalkozo
```

Ha a bazsi felhasználó létezik a rendszerben, még a teljes címet sem kell megadnunk. Ha valakit az interneten keresünk, a mail argumentumának értelemszerűen teljes nevet kell beírni.

Ez az olvasás egy kissé bonyolultabb, mint a kezdők által használt cat fajl | mail nev formájú üzenetküldés, de így jobban felhasználhatók a rendelkezésre álló eszközeink.

5.2.3. Átirányítások kombinálása

A kimeneti és bemeneti átirányításokat kombinálhatjuk is. Először vizsgáljuk meg a text.txt angol nyelvű szöveges fájl gépelési hibáit, majd a kimenetet irányítsuk át egy hibanaplóba:

```
aspell < text.txt > hibanaplo.log
```

Egyes rendszereken az alábbi szintaktikát kell használni (a man oldalakon lehet megnézni, nekünk melyik változatra van szükségünk):

```
aspell -H list < text.txt | sort -u
```

Az alábbi parancs kilistázza az összes olyan parancsot, mely a less használatánál egy másik fájl megvizsgálására szolgál:

```
tibi:~> less --help | grep -i examine
  :e [file]           Examine a new file.
  :n                * Examine the (N-th) next file from the command line.
  :p                * Examine the (N-th) previous file from the command line.
  :x                * Examine the first (or N-th) file from the command line.
```

Az egyes karakterek jelentése tehát a következő:

Karakter	Jelentés
e [fajl]	Egy új fájl megvizsgálása.
n	A parancssor (N.) következő fájljának vizsgálata.
p	A parancssor (N.) előző fájljának vizsgálata.
x	A parancssor első vagy N. fájljának vizsgálata.

Ha azt szeretnénk, hogy a keresés ne legyen kisbetű/nagybetű-érzékeny, használjuk az -i opciót.

Ha a parancs kimenetét el szeretnénk menteni jövőbeli felhasználásra, irányítsuk át a fájl kimenetét:

```
mike:~> less --help | grep -i examine > vizsgalando_fajlok
mike:~> cat vizsgalando_fajlok
  :e [file]           Examine a new file.
  :n                * Examine the (N-th) next file from the command line.
  :p                * Examine the (N-th) previous file from the command line.
  :x                * Examine the first (or N-th) file from the command line.
```

Egy parancs kimenete a | jellel más parancsokba elvileg tetszés szerinti alkalommal fűzhető addig, amíg ezek a parancsok problémamentesen olvassák a bemenetet az alapértelmezett bemenetről és ki tudják írni a kimenetet a standard kimenetre.

A kimenet átirányításánál vigyázzunk, nehogy felülírjuk már létező, fontos fájlokat. Sok shell, beleértve a Bash-t is, rendelkezik egy beépített védelemmel ilyen esetekre (noclobber). A Bash-ben a set -o noclobber parancsot kell hozzáadnunk a .bashrc konfigurációs fájlhoz a fájlok véletlenszerű felülírásának elkerülésére.

5.2.4. A >> operátor

A fájladatok felülírása helyett a létező fájlokhoz hozzá is fűzhetünk szöveget két, egymást követő nagyobb, mint karakterrel. Például:

```
zoli:~> date >> ajandek

zoli:~> cat ajandek
mobiltelefon
konyv
Sat Jan 29 11:43:05 CET 2005
```

A date parancs alapesetben az utolsó sort a képernyőre írja, most az ajandek nevű fájlhoz csatoltuk.

5.2.5. Fájlleírók használata

Háromféle I/O létezik, melyek mindegyikének saját azonosítója (fájlleírója) van:

- a standard bemenet: 0
- a standard kimenet: 1
- a standard hiba: 2

Ha a leírókban a fájlleíró számot elhagyva az átirányító operátor első karaktere <, az átirányítás a standard bemenetre hivatkozik (0-ás fájlleíró). Ha az átirányító operátor első karaktere >, az átirányítás a standard kimenetre hivatkozik (1-es fájlleíró). Nézzük egy gyakorlati példát! Az

```
ls > lista 2>&1
```

parancs mind a standard kimenetet, mind a standard hibát a lista fájlba irányítja. Ha csak a standard kimenetet szeretnénk a lista fájlba irányítani, akkor az alábbi parancsra van szükségünk:

```
ls 2>&1 > dirlist
```

Ez a lehetőség különösen hasznos a programozók számára.

5.2.6. Szűrők

A szűrő olyan program, mely a bemenet adataival végez műveleteket, az eredményt pedig az alapértelmezett kimenetre irányítja. A szűrők egyik leggyakoribb alkalmazási területe a kimenet átstruktúrálása. A következőkben a legfontosabb szűrőkkel ismerkedünk meg.

5.2.6.1. Szűrés grep-pel

A grep-pel már korábban megismerkedtünk. Akkor a kimenetet sorról sorra vizsgáltattuk vele és egyezőséget vizsgáltunk. Ez a viselkedés megfordítható a -v opció használatával.

Tegyük fel például, hogy szeretnénk tudni, egy adott könyvtár mely fájljai lettek módosítva december óta:

```
orsi:~> ls -la | grep Dec
```

A grep parancs a legtöbb parancshoz hasonlóan kisbetű/nagybetű-érzékeny. Ha nem szeretnénk különbséget tenni a kisbetűk és a nagybetűk között, használjuk az -i opciót. Számos GNU kiterjesztés is alkalmazható, így például a hosszú sorok böngészésében hasznos --colour vagy az --after-context, ami az utolsó egyező sor utáni sorok számát írja ki. Használhatjuk a grep-et rekurzívan is, ekkor a megszámolt könyvtárak összes alkönyvtárában keres (-r opció). Az opciók természetesen kombinálhatók is.

Reguláris kifejezéseket is használhatunk az összes bementi sorból kiválasztandó pontos karaktertalálatok további részletezéséhez. Ha reguláris kifejezésekkel szeretnénk foglalkozni, érdemes áttanulmányozni a grep dokumentációját, különösen a grep info oldalát.

5.2.6.2. A kimenet szűrése

A sort parancs a sorokat alapesetben abc-sorrendbe rendezi:

```
laci:~> cat baratok | sort
Bazsi
Leslie
Nagyi
Nickg
Nori
Szunci
Viki
```

A sort parancs azonban ennél sokkal több célra is felhasználható. Például egy könyvtár tartalmát képes fájlméret szerint rendezni, a legkisebb méretű fájltól a legnagyobb méretűig:

```
ls -la | sort -nk 5
```

A régebbi sort-szintaktikák ettől eltérhetnek. A fenti parancs hatása elvileg elérhető az ls -la | sort +4n paranccsal is, de ez egy régebbi megadás, mely nincs összhangban a jelenlegi szabványokkal.

A sort parancs másik fontos felhasználása a uniq programmal való kombinált felhasználása (vagy az -u opcióval kiegészített alkalmazása) a kimenet rendezéséhez és a kétszeres előfordulások kiszűrésére.

6. SZÖVEGSZERKESZTŐK

6.1. Szövegszerkesztők

A következőkben megnézzük a szövegszerkesztő programok felhasználhatóságát és fontosságát. A konkrét alkalmazások közül a fejlett vi(m) szerkesztőprogramot nézzük meg részletesebben. Foglalkozunk a fájlok szöveges módban történő megnyitásával és bezárásával, a fájlok szerkesztésével, a szövegben történő kereséssel, elveszett fájlok visszaállításával.

Nagyon fontos, hogy legalább egy konkrét szövegszerkesztő programmal megismerkedjünk, mert ez az első lépés a függetlenedés felé.

Az egész rendszerkörnyezetre hatással levő fájlok jó része szöveges fájl, így ha manuálisan szeretnénk módosítani ezeket a fájlokat, szükségünk lesz egy szövegszerkesztőre. A haladó felhasználók, a profik és a fejlesztők sem boldogulhatnak ezen eszköz nélkül, hiszen a scriptíráshoz, a weblapfejlesztéshez, a programozáshoz, tehát a legkülönfélébb felhasználási területekhez is (és nemcsak az operációs rendszer konfigurálásához) szükséges a szövegszerkesztő program.

6.2. A fontosabb szövegszerkesztők

Az igényeinknek megfelelő szövegszerkesztő programok kiválasztásakor érdemes azokra a programokra koncentrálni, melyek grafikus környezet, illetve terminálablak nélkül is használhatók (a legcsupaszabb Linux rendszeren is). Ha emellett ismerünk egyéb programokat is, az természetesen nem jelent hátrányt.

Az egyszerű szövegszerkesztők további előnye, hogy távoli gépen is használhatók (nem szükséges a teljes grafikus környezetet átvinni a hálózaton). A következőkben a leggyakrabban előforduló szövegszerkesztőkkel ismerkedünk meg.

6.2.1. Ed

Az ed egy sororientált szövegszerkesztő program. Alkalmas szövegfájlok létrehozására, tartalmának megjelenítésére, módosítására. Mind közvetlenül (interaktívan), mind shell scriptekben használható.

Az ed a UNIX rendszerek eredeti szövegszerkesztője (ebből kifolyólag széles körben elérhető). A legtöbb esetben azonban érdemesebb teljes képernyős szövegszerkesztőket (emacs, vi) használni.

6.2.2. GNU Emacs

Az emacs egy bővíthető, testreszabható, valós idejű megjelenítést biztosító szövegszerkesztő program, mely a legtöbb UNIX (és azon alapuló) rendszerben megtalálható.

Az emacs-ben szerkesztett szöveg a képernyőn látható és parancsaink kiadásakor automatikusan frissül. Azért nevezhetjük valós idejű szerkesztőprogramnak, mert a kijelzés nagyon gyakran frissül (általában minden begépelt karakter vagy karakterpár után). Ez a technológia minimalizálja a fejben tartandó szerkesztett információ mennyiségét.

Az emacs egy fejlett szerkesztőprogram, mert képességei túlmutatnak az egyszerű beillesztésen vagy törlésen. Képes alfolyamatok vezérlésére, programok automatikus elkezdésére, egyidejűleg több fájl megjelenítésére. Az emacs-ben nem probléma a formázott szöveg szerkesztése sem. Karakterszintű, szószintű, sorszintű, mondatszintű, bekezdésszintű és oldalszintű műveleteket hajthatunk végre. Még különféle programozási nyelvek kifejezéseit és megjegyzés-sorait is kezelni tudja.

Saját opcióink megtekintéséhez bármely időpillanatban használhatjuk a Ctrl + H billenytűkombinációt. Ezt arra is használhatjuk, hogy megnézzük, valamely parancs mit csinál, esetleg megkereshetjük az egy témához kapcsolódó összes parancsot.

A testreszabhatóság az emacs esetében azt jelenti, hogy egyszerűen módosíthatjuk a szövegszerkesztő program parancs-definícióit. Ha például egy olyan programozási nyelvet használunk, melyben a megjegyzéseket a /* nyitja és a */ karakterek zárják, rábírhatjuk a szövegszerkesztőt ezen jelek használatára. A testreszabás másik formája a parancshalmaz átrendezése (ha például a billentyűzet alapvető kurzormozgató nyílbillentyűit részesítjük előnyben a numerikus klaviatúra nyilaival szemben, ezt tudathatjuk a programmal).

A bővíthetőség az emacs esetében azt jelenti, hogy akár teljesen úgy parancsokat vagy programokat is írhatunk Lisp programozási nyelven, melyet aztán az emacs saját Lisp interpretere fog futtatni. Az emacs online bővíthető rendszer, mivel számos funkcióra van bontva, melyek egymást hívják meg. Ezek bármelyikét akár egy szerkesztés közepén is újradefiniálhatjuk. A program szinte bármely része eltávolítható anélkül, hogy egy különálló másolatot készítenénk a szövegszerkesztő összes részéről. A program legtöbb szerkesztő parancsa Lisp nyelven van megírva (a kivételt azok az egyébként Lisp nyelven is megírható parancsot képezik, melyeket C nyelven írtak a hatékonyság nö-

velése érdekében). Miután egy programozó készített egy fejlesztést a szövegszerkesztőhöz, azt később bárki használhatja.

Ha X Window rendszer alatt futtatjuk az emacs-ot (xemacs-ként indítva), a program menüit és kényelmes felületét egérgombokkal is vezérelhetjük. A program érdekessége, hogy csak szöveges terminálon is nyújtja egy ablakos rendszer számos előnyét. Például egyidejűleg több fájlt is megnyithatunk és szerkeszthetünk, a fájlok között szövegeket mozgathatunk, sőt a shell parancs futtatása alatt is szerkeszthetünk fájlokat.

6.2.3. Vi, vim

A vim program elődjéről, a vi-ról kapta nevét (*Vi improved*, azaz javított vi). A vim egy olyan szövegszerkesztő program, mely a UNIX-os vi szinte összes parancsát tartalmazza, de emellett sok új parancsa is van.

A vi szerkesztőben a parancsokat kizárólag a billenytűzetről adhatjuk ki. Ennek előnye, hogy ujjunkat a billentyűzeten tarthatjuk ahelyett, hogy kezünket állandóan a billentyűzet és az egér között mozgatnánk. Azok számára, akik mégis szeretnék egérrel is vezérelni a programot, lehetőség van egértámogatás igénybevételére is. A GUI verzió egérrel vezérelhető gördítősávokat és menüket tartalmaz.

Bár a vi, illetve a vim használata nem túl egyszerű, a program képességei igencsak figyelemreméltók. Ráadásul azért is érdemes legalább alapszinten megismerkedni a szerkesztővel, mert szinte az összes UNIX rendszeren ez a standard szövegszerkesztő. Gépenként és terminálonként lehetnek ugyan kisebb eltérések, de ha ismerjük a vi programot, az összes UNIX rendszeren boldogulni fogunk a szövegfájlok szerkesztésével. Ettől függetlenül természetesen van lehetőség más szövegszerkesztő programok használatára is.

Eltekintve a vim parancstól, a vim csomag a vim Gnome verzióját, a gvimet is tartalmazza. A kezdő felhasználók ezt egyszerűbben használhatónak találhatják, mert a menük segítenek, ha elfelejtettük vagy nem is ismerjük valamely szerkesztési fogást a standard vim parancsok használatával.

6.2.3.1. A vi két üzemmódja

A vi egy nagyon hatékony szövegszerkesztő program terjedelmes beépített felhasználói kézikönyvvel, melyet a program indulásakor a :help parancs használatával aktiválhatunk (itt a man, illetve az info oldalak nem tartalmaznak ennyi információt).

Ami a kezdő felhasználókat zavarhatja, az az, hogy a vi két üzemmódban tud dolgozni: parancs módban és beszúró módban. A szövegszerkesztő mindig parancs módban indul. A parancsok segítségével mozoghatunk a szövegben, kereshetünk, áthelyezhetünk részeket, blokkokat jelölhetünk ki és egyéb szer-

kesztési feladatokat végezhetünk (egyes műveletek átváltanak beszúró üzemmódba).

A fentiekből következik, hogy a vi programban egy-egy billentyűnek általában nem egy, hanem két jelentése van. Egyrészt jelenthet egy parancsot parancs üzemmódban, másrészt jelenthet egy szövegbe beszúrandó karaktert beszúró üzemmódban.

6.2.3.2. Mozgás a szövegben

A szövegben való mozgás legtöbbször a nyílbillentyűkkel lehetséges. Másik lehetőség az alábbi billentyűk használata:

- a h billentyű balra mozgatja a kurzort
- az l billentyű jobbra mozgatja a kurzort
- a k billentyű felfelé mozgatja a kurzort
- a j billentyű lefelé mozgatja a kurzort

A Shift + G billentyűkombináció a promptot a dokumentum végére helyezi.

6.2.3.3. Alapvető műveletek

Operátor	Jelentés
n dd	az aktuális kurzorpozíciótól töröl n sort
n dw	a kurzor jobb oldalától töröl n szót
x	az aktuális kurzorpozíción levő karakter törlése
:n	a fájl n. sorához ugrik
:w	menti a fájlt
:q	kilépés
:q!	feltétel nélküli kilépés
:wq	mentés és kilépés
:w ujfajl	a szöveget az ujfajl-ba menti
:wq!	felülbírálja a csak olvasható jogot (felülbírálási jog szükséges hozzá, pl. rendszergazdai bejelentkezés)
/string	megkeresi a stringet a fájlban és a kurzort az első találat alatti kurzorpozícióra helyezi
/	folytatja a keresést a fájlban, a kurzort a következő találathoz mozgatja
:1, $s/szo/masik_szo/g	a szo cseréje a masik_szo-ra az egész fájlban
yy	szövegblokk másolása
n p	beillesztés n-szer
:recover	fájl visszaállítása váratlan megszakadás esetén

6.2.3.4. A szerkesztőt beszúró üzemmódba váltó parancsok

Parancs	Jelentés
a	a kurzort egy pozícióval jobbra mozgatja, mielőtt átváltana beszúró üzemmódba
i	beszúrás
o	az aktuális kurzorpozíció alá beszúr egy üres sort és a kurzort erre a sorra mozgatja

Az Esc billentyűvel válthatunk vissza parancs üzemmódba. Ha nem tudjuk biztosan, melyik üzemmódban vagyunk éppen, mert a vi egy olyan régi verzióját használjuk, ami nem jelzi ki az INSERT üzenetet (beszúró üzemmódban), az Esc billentyűvel biztosan a parancssorba tudunk ugrani. Ha már parancs módban voltunk és úgy ütöttünk Esc-et, a rendszer hanghatással vagy egy képernyő-villanással jelezheti ezt.

6.3. A Linux irodai felhasználása

6.3.1. Történeti áttekintés

Az elmúlt évtizedben az irodai programcsomagok piacán a Microsoft cég Office programcsomagja lett az irányadó, a domináns. A Word szövegszerkesztő, az Excel táblázatkezelő, a PowerPoint prezentáció-készítő, az Access adatbázis-kezelő alkalmazások mintegy „ipari szabvánnyá" nőtték ki magukat és nagyon széles körben terjedtek el. A hozzájuk való – legalább kismértékű – igazodás előbb-utóbb elkerülhetetlen az irodai programokat használók számára.

A Microsoft monopolhelyzete igen hátrányosan érintette a Linux operációs rendszer újabb felhasználóit. Német fejlesztők egy csoportja ennek hatására életre keltette a StarOffice projektet, melynek eredeti célja egy Microsoft Office-klón létrehozása volt. A céget, a StarDivison-t az 1990-es évek végén megvásárolta a Sun Microsystems (az 5.2 változat előtt). A Sun ugyan folytatta a fejlesztést, de a forráshoz csak korlátozott hozzáférést kaptak. Az eredeti forrás fejlesztését az Open Office közösség folytatta, mely átnevezte a projektet OpenOffice-ra. Az OpenOffice ma már számos platformra (Linux, Windows, Solaris, MacOS) elérhető.

Közel egyidőben számos más projekt is készült. A Microsoft Office másik jó alternatívája lehet például a KOffice programcsomag, mely a KDE és/vagy SuSE felhasználók körében népszerű. Az eredetihez hasonlóan ez a klón is tartalmaz Microsoft Word és Excel-kompatibilis programokat és számos más alkalmazást is. Példaként nézzük meg a KOffice részeit:

Programcsomag-rész	Rövid leírás
KWord	Keret-alapú szövegszerkesztő professzionális dokumentumok készítéséhez.
KSpread	Egy hatékony táblázatkezelő alkalmazás.
KPresenter	Egy nagytudású prezentációs program.
Kivio	Egy Visio-stílusú program.
Karbon14	Egy vektorgrafikus rajzolóprogram.
Krita	Egy pixelgrafikus képszerkesztő program (a GIMP-hez és a Photoshop-hoz hasonlít).
Kugar	Üzleti jelentések készítésére alkalmas program.
KChart	Egy intergrált grafikon- és diagramrajzoló.
KFormula	Egy hatékony egyenletszerkesztő.
Kexi	Integrált környezet adatok menedzseléséhez.
Szűrők	A KOffice programcsomag programjainak importálásaihoz és exportálásaihoz.

Kisebb projektek keretében készültek egy adott Office termék „kiváltására" programok. Ilyen például az AbiWord vagy az MS WordView, melyek a Microsoft Word dokumentumokkal kompatibilisek, illetve a Gnumeric, ami Excel-kompatibilis táblázatok megtekintésére és létrehozására alkalmas.

6.3.2. Általános használat

Az irodai programcsomagok részeit csak arra használjuk, amire valók: irodai (vagy ilyen jellegű) dokumentumok elkészítésére.

Ha például egy elektronikus levelet szeretnénk valakinek küldeni egy formázott szöveges melléklettel, ezt csak akkor tegyük, ha tényleg szükség van rá. Ha elegendő egy egyszerű szöveg, ne használjunk irodai programok által generált fájlokat. Mivel ezek mérete nagyságrendekkel nagyobb, mint egy egyszerű szövegfájlé, tárolásuk komplikáltabb, a hálózaton nehezebb őket továbbítani. A címzettnek ráadásul rendelkeznie kell egy kompatibilis programmal, mely meg tudja nyitni a fájlt, különben nem értesül a benne foglaltakról (ezzel szemben egy egyszerű szövegfájlt bárki meg tud nézni, legfeljebb a karakterkódolással lehetnek problémák). A formázott szöveget vagy egyéb „irodai" dokumentumot olvasni képes programot a többi alkalmazás mellett pluszban kell futtatni, ami többletet jelent az erőforrás-felhasználásban is.

Nem túl jó ötlet az irodai programmal készített fájlok aláírásként való csatolása sem. Jobb megoldás ilyen esetekben a GPG, a PGP-kompatibilis GNU Privacy Guard vagy az SSL (Secure Socket Layer) tanúsítvány használata.

6.3.3. Rendszer- és felhasználói konfigurációs fájlok

Ha rendszerkörnyezetünket konfigurálni szeretnénk, még véletlenül se használjunk irodai programokat! Mivel ezeknél a programoknál formázott szövegekkel dolgozunk, számos sort vagy (sokszor nem látható) karaktert szúrhatnak a fájlokba, melyek a felhasználó programtól függően sokszor nem a megfelelő módon kerülnének értelmezésre. A konfigurációs fájlokhoz mindig egyszerű szövegszerkesztő programokat használjunk!

6.3.4. Ha mindenképpen grafikus szövegszerkesztőt szeretnénk

Ha ragaszkodunk a grafikus felülethez, próbáljuk ki a gedit-et vagy az xedit-et, melyek egyszerű szöveges fájlok feldolgozására valók. Ha azonban igazán komoly feladatokat szeretnénk elvégezni, erősen ajánlott egy igazi szöveges üzemmódú szövegszerkesztő (vim, emacs) használata.

Egy elfogadható alternatíva a vim Gnome verziója, a gvim. Itt is a vi parancsait használhatjuk, de ha bajban vagyunk, a menük segítségünkre siethetnek.

7. KONFIGURÁLÁS

A következőkben rendszerünk konfigurálásával foglalkozunk részletesebben. Most már tudjuk, hogyan használjunk egy szövegszerkesztőt (persze a használatot gyakorolni is kell), a legkülönfélébb típusú fájlokat módosíthatjuk saját kényelmünk érdekében. Rendszerkörnyezetünk szervezése mellett a shell setup fájlokkal és a shell konfigurációjával is megismerkedünk. Megismerjük a prompt, illetve a grafikus környezet konfigurálását. Foglalkozunk a multimédia-alkalmazásokkal. Megnézünk néhány ablakozó rendszert és megtudhatjuk azt is, hogyan működik az X kliens-szerver rendszer. Foglalkozunk a nyelvi és betűtípus-beállításokkal, új programok telepítésével és meglévő csomagjaink frissítésével is.

7.1. Karbantartás

Az alábbiakban olyan fogásokkal ismerkedünk meg, melyek nemcsak a programozók életét teszik könnyebbé, de bármely felhasználóét is. Ha a rendszer menedzselésével vannak problémáink, akkor is jól hasznosíthatjuk ezeket az ismereteket. Néhány tanács a rendszer használatának könnyebbé tételére:

- programfájljaink és scriptjeink részére hozzunk létre egy bin könyvtárat
- a nem futtatható fájlokat rendezzük külön könyvtárakba és tetszés szerinti mennyiségben hozzunk létre új könyvtárakat a fájlok rendezésére (a képek, a dokumentumok, a munkáink, a letöltött fájlok, a táblázatok, a személyes fájlok stb. mind-mind lehetnek külön könyvtárban)
- könyvtárainkat tegyük priváttá a chmod 700 konyvtarnev szintaktikájú paranccsal
- olyan elnevezéseket alkalmazzunk, melyek utalnak a tartalomra (pl. level_az_igazgatonak_050406, nem pedig level1)

7.2. Helytakarékosság

Rendszerünkben a lemezkvóta-rendszer, illetve a merevlemez fizikai korlátozása kényszerítheti a felhasználókat, hogy időnként szabaduljanak meg a

fölösleges adatoktól. Számos lehetőség mellett tárhely felszabadításához használhatjuk az rm parancsot.

A quota -v parancs segítségével ellenőrizhetjük a szabad terület nagyságát.

7.2.1. Fájlok kiürítése

Néha egy fájl tartalmára már nincs szükségünk, de a fájlnévvel szeretnénk jelezni magunknak, hogy ezen a helyen volt egy ilyen fájl, illetve a jövőben majd írni szeretnénk bele.

A fájl tartalmát a null parancs kimenetének átirányításával üríthetjük:

```
imi:~> cat ajandek > helyfenntarto

imi:~> ls -la helyfenntarto
-rw-rw-r--        1 imi imi         118 Feb 14 11:19 helyfenntarto

imi:~> > helyfenntarto

imi:~> ls -la helyfenntarto
-rw-rw-r--        1 imi imi           0 Feb 14 11:20 helyfenntarto
```

Egy létező fájl tartalmának törlése és ugyanolyan néven (0 bájt hosszúsággal) történő meghagyásának folyamatát csonkolásnak hívjuk.

Egy új üres fájl létrehozásához a touch parancsra van szükség. Ha a fájl már létezik, a touch parancs csak a fájl idő-adatait frissíti.

Egy fájl részleges ürítéséhez használjuk a tail parancsot. Ha ajándéklistánk túl hosszú lett, mert új elemekkel bővítettük, de a már ajándékozott tárgyakat soha nem töröltük, lehetőség van például az utolsó 5 sor megtartására és az azt megelőző sorok törlésére is:

```
imi:~> tail -5 ajandeklista > ujlista

imi:~> cat ujlista > ajandeklista

imi:~> rm ujlista
```

7.2.2. A naplófájlokról részletesebben

Egyes Linux programok mindenfajta kimenetet egy log fájlba írnak. A legtöbbször opciókkal megadhatjuk, hogy csak a hibák kerüljenek naplózásra vagy csak minimális információk kerüljenek log fájlokba (pl. a program hibakeresési szintjének beállítása).

Sokszor egyáltalán nincs szükség a naplófájlokra, esetleg korlátozásokkal érdemes lenne a méretüket minimalizálni:

- Próbáljuk meg eltávolítani a log fájlt, mikor nem fut a program, ha biztosak vagyunk benne, hogy arra többé nincs szükség. Egyes programok indulásukkor ellenőrzik, hogy van-e naplófájl és ha nem talál, nem naplózik.

- Ha eltávolítottuk a log fájlt, de a program újra létrehozta azt, olvassuk el az adott program dokumentációját és keressük meg azokat az opciókat, melyekkel elkerülhető a log fájlok létrehozása.
- Próbáljuk meg minimalizálni a naplófájlok méretét és csak azokat az információkat tartsuk meg, melyre feltétlenül szükség van.
- Próbáljuk meg helyettesíteni a fájlt egy /dev/null szimbolikus hivatkozással. Ha szerencsénk van, a program nem fog hibát jelezni. Azon programok log fájljaival, melyek rendszerbetöltéskor vagy a cron-ból futnak, ne tegyük meg ezt a lépést! (Ezen programok a szimbolikus hivatkozást kicserélhetik egy kisméretű fájlra, mely újra elkezd nőni.)

7.2.3. A levelezés ismérvei

Rendszeresen ürítsük postafiókunkat! Készítsünk alkönyvtárakat és alkalmazzunk automatikus átirányításokat a procmail használatával vagy kedvenc levelezőprogramunk szűrőivel. Ha használunk kukát, azt rendszeresen ürítsük!

A levelek átirányításához használjuk home könyvtárunk .forward fájlját. A Linux levelezési szolgáltatása mindig ezt a fájlt nézi meg, ha helyi levelet kell kézbesítenie. A fájl tartalma definiálja, mit kell tennie a levelezőrendszernek levelünkkel. Előfordulhat, hogy csak egyetlen sora van, mely egy e-mail cím teljes megadását tartalmazza. Ebben az esetben a rendszer összes levelünket erre a címre küldi. Ha például egy webhelynek szeretnénk helyet megtakarítani, a webmesternek címzett leveleket továbbíthatjuk saját azonosítónkhoz. Ekkor a webmester .forward fájlja valahogy így néz ki:

```
webmester@disz ~/> cat .forward
bazsi@siki-soft.hu
```

A levéltovábbítás azért is hasznos, mert segítségével nem kell több különböző postafiókot ellenőrizgetnünk. Az összes címet egy központi, könnyen hozzáférhető címre továbbíthatjuk.

A levéltovábbítást a rendszergazda állíthatja be a helyi levél-aliasz fájlban.

7.2.4. Helymegtakarítás hivatkozásokkal

Ha több felhasználónak kell hozzáférést biztosítani ugyanahhoz a fájlhoz vagy programhoz, mikor az eredeti fájlnév túl hosszú vagy nehezen megjegyezhető, használjunk egy szimbolikus hivatkozást ahelyett, hogy minden felhasználónak (minden célra) külön másolatot készítenénk.

Az összetett szimbolikus hivatkozásoknak bonyolult nevei is lehetnek. Ezek a (különböző nevű) hivatkozások ugyanarra a fájlra akár ugyanabban a könyvtárban is lehetnek. Ez gyakran előfordul például a /lib könyvtárban. Nézzük is meg:

```
ls -l /lib
```

Számos olyan hivatkozást láthatunk, melyek ugyanarra a fájlra mutatnak. Ennek az az oka, hogy ugyanazt a fájlt egyes programok más néven keresik (és meg is találják anélkül, hogy több példányban tárolnánk ugyanazt a fájlt más néven).

7.2.5. Fájlméret-korlátozás

A shell egy beépített parancsot tartalmaz a fájlméretek korlátozására. Ez az ulimit, mellyel meg is jeleníthetők a rendszer erőforrásainak korlátozásai:

```
erno:~> ulimit -a
core file size (blocks)         0
data seg size (kbytes)          unlimited
file size (blocks)              unlimited
max locked memory (kbytes)      unlimited
max memory size (kbytes)        unlimited
open files                      1024
pipe size (512 bytes)           8
stack size (kbytes)             8192
cpu time (seconds)              unlimited
max user processes              512
virtual memory (kbytes)         unlimited
```

Az unlimited jelzésű tételek nem korlátozottak.

7.2.6. Fájlok tömörítése

A tömörített fájlok előnye, hogy kevesebb tárhelyet igényelnek. A kisebb méret miatt a hálózaton is könnyebb átküldeni ezeket az állományokat. Rendszerünkön számos fájl (így például a man oldalak is) tömörített formátumban vannak tárolva. A technológia hátránya, hogy ezen információk kicsomagolása, majd használat utáni visszatömörítése időigényes. Egy man oldalt például egy parancs-opció leírásának elolvasása után nem szeretnénk tömöríteni. A legtöbb ember valószínűleg elfelejtené betömöríteni a fájlokat, miután megtalálta az általa keresett információkat.

A probléma megoldására készültek azok az eszközök, melyek képesek tömörített fájlokkal dolgozni úgy, hogy csak a memóriába tömörítik ki azok tartalmát. A tömörített fájl foglal csak helyet a merevlemezen. A legtöbb rendszer támogatja az egrep, zgrep, zcat, zmore és hasonló eszközöket a szükségtelen kicsomagolási és betömörítési műveletek megelőzésére.

7.3. A szöveges környezet

7.3.1. Környezeti változók

A korábbiakban már több környezeti változóval is foglalkoztunk (PATH, HOME). Eddig a változók egy meghatározott céllal szolgálták a shell-t. Van azonban számos más Linux kellék is, melyeknek a hatékony munkához információra van szükségük rólunk. Felvetődhet a kérdés: milyen információkra van szüksége a programoknak az útvonalakon és a home könyvtárakon kívül?

Sok program szeretné tudni az általunk használt terminál típusát, mely a TERM változóban van tárolva. Szöveges módban a linux terminál emulációt használhatjuk, grafikus módban pedig inkább az xterm-et.

Számos alkalmazásnak szüksége lenne arra az információra is, melyik a kedvenc szövegszerkesztő programunk (ekkor el kell indítaniuk egy szerkesztőt egy alfolyamatban). Az általunk használt shell a SHELL környezeti változóban van tárolva. Az operációs rendszer típusát az OS változó tartalmazza. Aktuálisan definiált változóink listáját a printenv paranccsal kérhetjük le.

A környezeti változókat a shell menedzseli. A közönséges shell változókkal ellentétben a környezeti változókat az összes elindított program örökli, beleértve egy másik shell-t is. Az új folyamatok ezen változók egy másolatát kapják meg, melyet olvashatnak, módosíthatnak és átadhatnak saját gyerekfolyamataiknak.

A környezeti változók neve nem különleges. A beépített környezeti változók neve csupa nagybetűs, így lehet őket megkülönböztetni saját változóinktól, melyeket kisbetűvel célszerű létrehozni. Az elnevezésben nincs semmilyen megkötés, csupán a rendszer standard változóitól való eltérést kell biztosítani (a PATH vagy a HOME annyira fontos környzeti változók, hogy minden Linux rendszerben megtalálhatók).

7.3.2. Változók exportálása

Egy egyedi változó tartalma az echo parancs használatával jeleníthető meg:

```
tomi:~> echo $PATH
/usr/bin:/usr/sbin:/bin:/sbin:/usr/X11R6/bin:/usr/local/bin

tomi:~> echo $MANPATH
/usr/man:/usr/share/man/:/usr/local/man:/usr/X11R6/man
```

Ha úgy szeretnénk megváltoztatni egy változó tartalmát, hogy azt más programok is használni tudják, környezetünkből exportálni kell az új értéket abba a környezetbe, mely ezeket a programokat futtatja. Gyakori példa erre a PATH változó exportálása. Ha egy olyan szoftvert szeretnénk futtatni, mely az /opt/foldrajz/bin könyvtárban van, az alábbi szintaktikára van szükség:

```
tomi:~> PATH=$PATH:/opt/foldrajz/bin
```

A parancs arra utasítja a shell-t, hogy ne csak az aktuális útvonalakon ($PATH) keresse a programokat, hanem az /opt/foldrajz/bin könyvtárban is.

Amíg a PATH változó új értékét a környezet nem ismeri, a dolog nem működik:

```
tomi:~> terkep
bash: terkep: command not found
```

A változók exportálása a shell beépített export parancsával lehetséges:

```
tomi:~> export PATH

tomi:~> terkep
- A megyei terkepprogram betoltese folyamatban. Kerem, varjon... -
```

A Bash-ben ezt egyetlen lépésben elvégezhetjük:

```
export VALTOZO=ertek
```

Ugyanez a technika alkalmazható a MANPATH változó esetében is, ami közli a man paranccsal, hol találhatók a tömörített man oldalak. Ha új programot adunk a rendszer új vagy nem használt könyvtáraihoz, valószínűleg a dokumentációk is egy nem használt könyvtárba kerülnek. Ha az új szoftver man oldalait szeretnénk nézegetni, bővítsük a MANPATH változót:

```
tomi:~> export MANPATH=$MANPATH:/opt/foldrajz/man

tomi:~> echo $MANPATH
/usr/man:/usr/share/man:/usr/local/man:/usr/X11R6/man:/opt/foldrajz/man
```

Ezt a parancsot nem kell minden újonnan nyitott ablakba begépelnünk, ha valamelyik shell setup-fájlunkhoz adjuk.

7.3.3. Fenntartott változók

Az alábbiakban összefoglaljuk a legfontosabb beépített változókat:

Változó	Tárolt információ
DISPLAY	az X Window rendszer használja a kijelző-szerver azonosítására
DOMAIN	domain-név
EDITOR	kedvenc szövegszerkesztőnket tárolja
HISTSIZE	a shell kiadott parancsokat tároló fájljának mérete a sorok számával kifejezve
HOME	saját home könyvtárunk elérési útvonala
HOSTNAME	helyi hosztnév

Változó	Tárolt információ
INPUTRC	bemeneti eszközök (pl. billentyűzet) definíciós fájljának elérése
LANG	elsődleges nyelv
LD_KONYVTAR_UTVONAL	útvonalak könyvtár-keresésekhez
LOGNAME	bejelentkezési név
MAIL	a bejövő leveleinket tartalmazó könyvtár elérése
MANPATH	útvonalak man-oldalak kereséséhez
OS	az operációs rendszert leíró string
OSTYPE	több információ az operációs rendszerről (pl. verzió)
PAGER	az olyan programok használják, melyeknek tudniuk kell, mi a teendő, ha a kimenet hosszabb egy terminálablaknál (pl. man).
PATH	elérési útvonalak parancsokhoz
PS1	elsődleges prompt
PS2	másodlagos prompt
PWD	jelenlegi munkakönyvtár
SHELL	az aktuális shell
TERM	a terminál típusa
UID	felhasználó-azonosító
USER(NAME)	felhasználói név
VISUAL	kedvenc teljes képernyős szövegszerkesztő programunk
XENVIRONMENT	grafikus környezetünk személyes beállításainak helye
XFILESEARCHPATH	grafikus könyvtárak elérési útvonalai

Számos változó nemcsak hogy beépített, de értéke előre be van állítva (konfigurációs fájlok használatával).

7.3.4. A shell setup fájljai

Ha saját home könyvtárunkban kiadunk egy ls -al parancsot az összes fájl (a . karakterrel kezdődőket is beleértve) hosszú listázására, legalább egy . karakterrel kezdődő és rc végződésű fájlt látunk. Ha bash-t használunk, ez a fájl a .bashrc, ami nem más, mint a rendszerszerte használt /etc/bashrc konfigurációs fájl másolata.

Ha egy inaktív bejelentkező shell-be jelentkezünk be, a login végzi az azonosítást, beállítja a környezetet és elindítja a shell-ünket. A bash esetében a

következő lépés az általános profil, vagyis a profile beolvasása az /etc könyvtárból (ha ez a fájl létezik). A bash ezután a ~/.bash_profile, ~/.bash_login és a ~/.profile fájlokat keresi pontosan ebben a sorrendben. Az első létező és olvasható fájlból kiolvassa és futtatja a parancsokat. Ha egyik sem létezik, az /ect/bashrc lesz érvényben.

Ha létezik egy login shell, a bash beolvassa és futtatja a ~/.bash_logout parancsait, ha ez a fájl létezik.

A folyamatot a login és a bash man oldalai részletesen tárgyalják.

7.3.5. A tipikus beállítások

Vessünk egy pillantást a legfontosabb konfigurációs fájlokra! Kezdjük az /etc/profile fájllal, ahol olyan fontos változók kerülnek beálíltásra, mint a PATH, a USER és a HOSTNAME:

```
tomi:~> cat /etc/profile
# /etc/profile

# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc


# Path manipulation
if [ `id -u` = 0 ] && ! echo $PATH | /bin/grep -q "/sbin" ; then
        PATH=/sbin:$PATH
fi

if [ `id -u` = 0 ] && ! echo $PATH | /bin/grep -q "/usr/sbin" ; then
        PATH=/usr/sbin:$PATH

fi

if [ `id -u` = 0 ] && ! echo $PATH | /bin/grep -q "/usr/local/sbin"
        then
        PATH=/usr/local/sbin:$PATH
fi

if ! echo $PATH | /bin/grep -q "/usr/X11R6/bin" ; then
        PATH="$PATH:/usr/X11R6/bin"
fi
```

Ezek a sorok állapítják meg a beállított útvonalakat. Ha a rendszergazda megnyit egy shell-t (a felhasználó-azonosító 0), megállapításra kerül, hogy az /sbin, az /usr/sbin és az /usr/local/sbin szerepelnek az útvonalban. Ha nem így van, hozzáadódnak. Ez minden felhasználónál megállapításra kerül, akinek az útvonalában szerepel az /usr/X11R6/bin.

```
# No core files by default
ulimit -S -c 0 > /dev/null 2>&1
```

Az összes „hulladék" a /dev/null-ba kerül, hacsak a felhasználó át nem állítja ezt a beállítást.

```
USER=`id -un`
LOGNAME=$USER
MAIL="/var/spool/mail/$USER"
HOSTNAME=`/bin/hostname`
HISTSIZE=1000
```

Az általános változók megkapják saját értékeiket.

```
if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ]; then
        INPUTRC=/etc/inputrc
fi
```

Ha az inputrc változó nincs beállítva és nincs .inputrc a felhasználó home könyvtárában, az alpértelmezett bemeneti vezérlőfájl kerül betöltésre.

```
export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE INPUTRC
```

Az összes változó exportálásra kerül, így más, a környezetünkről információkat igénylő programok számára is elérhetővé válnak.

```
for i in /etc/profile.d/*.sh ; do
        if [ -r $i ]; then
                . $i
        fi
done
unset i
```

Az /etc/profile.d könyvtár összes olvasható shell scriptje beolvasásra és futtatásra kerül. A színes ls engedélyezése, a vi aliasszá állítása vim-re stb. mind itt kerülnek beállításra. Az ideiglenes i változó beállítását végül elvetjük, így elkerülhető, hogy a későbbiekben a shell viselkedését zavarja.
A bash a felhasználó home könyvtárában keresi a .bash_profile-t:

```
tomi:~> cat .bash_profile
#############################################################
#                                                           #
#           .bash_profile file                              #
#                                                           #
#           Executed from the bash shell when you log in.   #
#                                                           #
#############################################################

source ~/.bashrc
source ~/.bash_login
```

Az előző fájl utasítja a shell-t, hogy először a ~/.bashrc-t, majd a ~/.bash_login-t olvassa.

A beépített shell paranccsal, a source-szal gyakran találkozunk, ha egy shell környezetben dolgozunk, hiszen segítségével alkalmazhatjuk a megváltoztatott konfigurációkat az aktuális környezetre.

A ~/.bash_login egy alapvető fájlvédelmet definiál az umask érték beállításával.

A ~/.bashrc fájl felhasználó-specifikus aliaszok, funkciók és személyes környezeti változók beállítására szolgál. Először az /etc/bashrc-t olvassa, ami leírja az alapértelmezett promptot (PS1) és az alapértelmezett umask értéket. Ezután adhatjuk meg saját beállításainkat. Ha nem létezik a ~/.bashrc, alapból a /etc/bashrc kerül beolvasásra:

```
tomi:~> cat /etc/bashrc
# /etc/bashrc

# System wide functions and aliases
# Environment stuff goes in /etc/profile

# by default, we want this to get set.
# Even for non-interactive, non-login shells.
if [ `id -gn` = `id -un` -a `id -u` -gt 99 ]; then
        umask 002
else
        umask 022
fi

# are we an interactive shell?
if [ "$PS1" ]; then
        if [ -x /usr/bin/tput ]; then
if [ "x`tput kbs`" != "x" ]; then
# We can't do this with "dumb" terminal
                stty erase `tput kbs`
        elif [ -x /usr/bin/wc ]; then
                if [ "`tput kbs|wc -c `" -gt 0 ]; then
# We can't do this with "dumb" terminal
                  stty erase `tput kbs`
                fi
        fi
fi
case $TERM in
        xterm*)
        if [ -e /etc/sysconfig/bash-prompt-xterm ]; then
                PROMPT_COMMAND=/etc/sysconfig/bash-prompt-xterm
        else
 PROMPT_COMMAND='echo -ne "\033]0;${USER}@${HOSTNAME%%.*}:\
${PWD/$HOME/~}\007"'
        fi
```

```
    ;;
        *)
[ -e /etc/sysconfig/bash-prompt-default ] && PROMPT_COMMAND=\
/etc/sysconfig/bash-prompt-default
                ;;
  esac
  [ "$PS1" = "\\s-\\v\\$ " ] && PS1="[\u@\h \W]\\$ "

  if [ "x$SHLVL" != "x1" ]; then # We're not a login shell
        for i in /etc/profile.d/*.sh; do
                if [ -x $i ]; then
                        . $i
                fi
        done
    fi
fi
```

A kijelentkezés alatt a ~/.bash_logout-ban levő parancsok hajtódnak végre (például terminál törlése):

```
tomi:~> cat .bash_logout
# ~/.bash_logout

clear
```

A következőkben azt nézzük meg, hogyan is működnek ezek a scriptek.

7.3.6. A Bash prompt

A Bash prompt sokkal többre is képes, mint egyszerű információk megjelenítése (felhasználói név, gépünk neve, az aktuális könyvtár). Olyan információkat is megjeleníthetünk vele, mint az aktuális dátum és idő vagy a kapcsolódott felhasználók száma.

Mielőtt nekiállnánk módosítani, elmentjük aktuális promptunkat egy másik környezeti változóba:

```
[andris@disz andris]$ SAJATPROMPT=$PS1

[andris@disz andris]$ echo $SAJATPROMPT
[\u@\h \W]\$

[andris@disz andris]$
```

Most már nyugodtan megváltoztathatjuk a promptot, például a PS1="->" parancs kiadása után mindig visszakaphatjuk az eredeti promptot a PS1=$SAJATPROMPT paranccsal. Természetesen akkor is visszakapjuk, ha újra csatlakozunk a rendszerhez.

7.3.6.1. Példák a prompt beállítására

Az aktuális idő és a futó feladatok számának kiírása:

```
export PS1="[\t \j] "
```

A dátum, a felhasználói név, a hosztnév és az aktuális munkakönyvtár megjelenítése:

```
export PS1="[\d][\u@\h \w] : "
```

Megjegyzendő, hogy a \W csak az aktuális munkakönyvtár alapvető neveit jeleníti meg.

Minden parancs kiadási számát jeleníti meg az

```
export PS1="{\!} "
```

A felhasznalo@hoszt rózsaszínben jelenik meg:

```
export PS1="\[\033[1;35m\]\u@\h\[\033[0m\] "
```

A felhasználói név rózsaszínnel, az aktuális munkakönyvtár kékkel jelenik meg:

```
export PS1="\[\033[1;35m\]\u\[\033[0m\] \[\033[1;34m\]\w\[\033[0m\] "
```

Egy azok számára hasznos prompt, akik nehezen tudják elkülöníteni magát a promptot az általuk gépeltektől:

```
export PS1="\[\033[1;44m\]$USER is in \w\[\033[0m\] "
```

Aláhúzottan megjelenő prompt:

```
export PS1="\[\033[4;34m\]\u@\h \w \[\033[0m\]"
```

Ha fehér karakterekkel szeretnénk gépelni kék háttéren:

```
export PS1="\[\033[7;34m\]\u@\h \w \[\033[0m\] "
```

A változók exportálásával a későbbiekben futtatott parancsok is ismerni fogják a környezetet. Az általunk igényelt prompt-konfigurációs fájl legjobb helye a shell konfigurációs fájlunkban, a ~/.bashrc-ben van.

Ha szeretnénk, promptunk akár shell scripteket is képes futtatni, így különböző feltételek hatására másképpen viselkedik. Még arra is lehetőség van, hogy minden parancs kiadása után a prompt egy hangeffektust produkáljon, bár az ilyen és ehhez hasonló megoldások hamar unalmasak lehetnek.

7.3.7. Shell scriptek

Mint a korábbiakban azt a shell-konfigurációs példáknál már láttuk, a shell script nem más, mint egy olyan szöveges fájl, mely shell parancsokat tartalmaz. Ha egy ilyen fájlt használunk az első nem opciós argumentumként a Bash meghívásakor és sem a -c, sem az -s opciót nem használjuk, a Bash

elolvassa és futtatja a fájlban levő parancsokat, majd kilép. Ez a működési mód egy nem interaktív shell-t eredményez. Amikor a Bash egy shell scriptet futtat, a 0 speciális karaktert állítja be fájlnévnek (a shell neve helyett). Az itteni paraméterek a maradék argumentumokra lesznek beállítva, ha van ilyen megadva. Ha nincsenek további argumentumok, ezek a paraméterek törlődnek.

Egy shell script lehet futtatható is (a chmod paranccsal bekapcsolhatjuk a futtat bitet). Ha a Bash egy ilyen fájlt talál, amikor egy parancs miatt végignézi a PATH-t, egy alshell-t hoz létre a futtatásához. Másképpen fogalmazva a

fajlnev ARGUMENTUMOK

futtatása egyenértékű a

bash fajlnev ARGUMENTUMOK

futtatásával, ha a fajlnev nevű fájl egy futtatható shell script. Ez az alshell újrainicializálja magát, így a gyerek megakadályozza, hogy egy új shell legyen meghívva a script értelmezésére (kivételt jelent az, ha a szülő megjegyzi a parancsok helyét).

A legtöbb UNIX változat az imént felsoroltakat az operációs rendszer parancsfuttatási mechanizmusának részeként kezeli. Ha egy script első sora a #! karakterpárossal kezdődik, a sor hátralevő része egy értelmezőt határoz meg a programhoz (ez lehet a bash, az awk, a perl stb.). A scriptfájl nagy része az adott nyelven van megírva.

Az interpreter argumentumai egy egyszerű opcionális argumentumból, az azt követő interpreternévből (a scriptfájl első sorában), a scriptfájl nevéből és a további argumentumokból áll. Azokon az operációs rendszereken, melyek maguk nem kezelik le ezt a műveletet, a Bash hajtja végre.

A Bash scriptek gyakran kezdődnek a #!/bin/bash stringgel (ha a Bash a /bin-be van telepítve), ezzel biztosíthatjuk, hogy mindenképpen a Bash értelmezze a scriptet, még akkor is, ha az egy másik shell alatt van futtatva.

7.3.7.1. Példák shell scriptekre

Nézzünk egy egyszerű, csupán egyetlen parancsot tartalmazó scriptet, ami üdvözli az őt futtató felhasználót:

```
[kriszti@disz ~] cat szia.sh
#!/bin/bash
echo "Szia, $USER"
```

A script egyetlen parancsa az echo, mely a USER környezeti változó értékét ($USER) használja fel, hogy a parancsot kiadó felhasználót saját nevével köszönthesse.

Egy másik jól használható egysoros script a kapcsolódott felhasználókat jeleníti meg:

```
#!/bin/bash
who | cut -d " " -f 1 | sort -u
```

Ideje megnézni egy összetettebb, többsoros shell scriptet is! A kedvenc zenénket tartalmazó WAV fájlokat MP3-ba szeretnénk tömöríteni helytakarékosság végett a lame paranccsal. Ehhez a script először készít egy listát az aktuális könyvtár összes WAV fájljáról, majd azt a LISTA változóba teszi. Ezután beállítja az új fájlnevet az összes fájlhoz, majd elkezdi a konverziót. Végül eltünteti az óriási WAV fájlokat, hiszen helyettük már ott vannak a tömörített változataik:

```
tille:~> cat /usr/local/bin/zene

#!/bin/bash
# .WAV konvertalas .MP3-ba
LISTA=$(ls *.wav)
for i in $LIST; do
        EREDETI=$i
        TOMOR=$(ls $i | cut -d "." -f 1).mp3
        lame -h $EREDETI $TOMOR
        echo "a $i konvertalasa kesz, az eredeti fajl eltavolitasa folyamatban.."
        rm $i
done
```

A fájlnevet a fájlvégződéstől a cut paranccsal választjuk le (az mv *.wav *.mp3 szintaktika nem működik). A műveletekről a felhasználót az echo parancs értesíti. (Az echo scriptek hibakeresésénél is kitűnő szolgálatot tehet.)

Végül nézzünk meg egy komolyabb scriptet az /etc/rc.d/init.d könyvtár példascriptjei közül!

```
#!/bin/sh
# description: ICanSeeYou allows you to see networked people

# process name: ICanSeeYou
# pidfile: /var/run/ICanSeeYou/ICanSeeYou.pid
# config: /etc/ICanSeeYou.cfg

# Source function library.
. /etc/rc.d/init.d/functions

# See how (with which arguments) we were called.
case "$1" in
        start)
                echo -n "Starting ICanSeeYou: "
                daemon ICanSeeYou
                echo
                touch /var/lock/subsys/ICanSeeYou
```

```
                ;;
        stop)
                echo -n "Shutting down ICanSeeYou: "
                killproc ICanSeeYou
                echo
                rm -f /var/lock/subsys/ICanSeeYou
                rm -f /var/run/ICanSeeYou/ICanSeeYou.pid
                ;;
        status)
                status ICanSeeYou
                ;;
        restart)
                $0 stop
                $0 start
                ;;
        *)
                echo "Usage: $0 {start|stop|restart|status}"
                exit 1
esac
exit 0
```

A script egy képzeletbeli ICanSeeYou (látlak) nevű szervert vezérel. Az első sorban a pont (.) parancs hatására a legtöbb shell scriptben előforduló shell függvények gyűjteményét betölti az /etc/rc.d/init.d könyvtárból. Ezután egy case paranccsal egy esetszétválasztást ír le a script, ami négy különböző módon is futhat (indítás, leállítás, állapot, újraindítás). Azt, hogy melyik változat fut le, a script (első) argumentuma dönti el (a $1 kifejezés). Ha a script nem kap megfelelő bemenetet, a csillag (*) karakterrel jelzett alapértelmezés fut le, vagyis egy hibaüzenetet kapunk. A case ág végét az esac utasítás zárja (a case visszafelé írva, hasonlóan az if fi párjához). A start ágban a szerverprogram egy démonként indul el. Egy folyamatazonosító és a lokkolás is beállításra kerül. A stop ágban a szerverfolyamat leáll, a folyamatazonosító és a lokkolás eltávolításra kerül. Az opciók (mint a daemon opció) és a függvények (mint pl. a killproc) az /etc/rc.d/init.d/functions fájlban vannak definiálva. (Ez a beállítás disztribúciófüggő!) Rendszerünk inicializáló scriptjei más könyvtárakban definiált más függvényeket is használhatnak.

Ha sikeres volt, a script egy 0 kilépő kódot ad vissza a szülőjének.

Egy ilyen scriptben már jól látható, hogyan lehet egy scriptet könnyebben olvashatóvá tenni. Érdekesség, hogy a script az sh-t használja a bash helyett (ennek oka, hogy szélesebb körben is felhasználható legyen). (Linux-ban a bash meghívása sh-ként POSIX-nak megfelelő shell futtatását jelenti.)

7.4. A grafikus környezet

Egy átlagos felhasználó gyakran nem sokat foglalkozik login beállításaival. A Linux számos asztal-menedzsert kínál a grafikus környezet, az X alá. Az ablakozók és asztalok használata és konfigurációja egyértelmű és a UNIX CDE, a Machintosh és a Microsoft Windows felületére is hasonlít. Sok felhasználónak a csillogó-villogó felület és a felhasználóbarátság a legfontosabb szempont a grafikus felületnél.

7.4.1. Az óra beállítása

Ha már nagy felbontású grafikus felületen dolgozunk, miért ne ketyeghetne egy óra a képernyő egyik sarkában? Az igen jó szolgálatot tevő óra többféleképpen is megjelenhet, az óra mellett akár a dátumot is kijelzhetjük (korábban már volt róla szó). Ahhoz, hogy igazán használható legyen egy ilyen lehetőség, szükség van arra is, hogy be tudjuk állítani a megfelelő értékeket (bár ma már lehetőség van arra is, hogy egy internetes alkalmazás időről időre beállítsa az óránkat valamely atomórához). Az ilyen és ehhez hasonló manőverekhez rendszergazdai jogosultságra is szükségünk lehet. Ha ez nem lenne meg, sokszor csak a lehetőségek leszürkített, ki nem választható változatait látjuk.

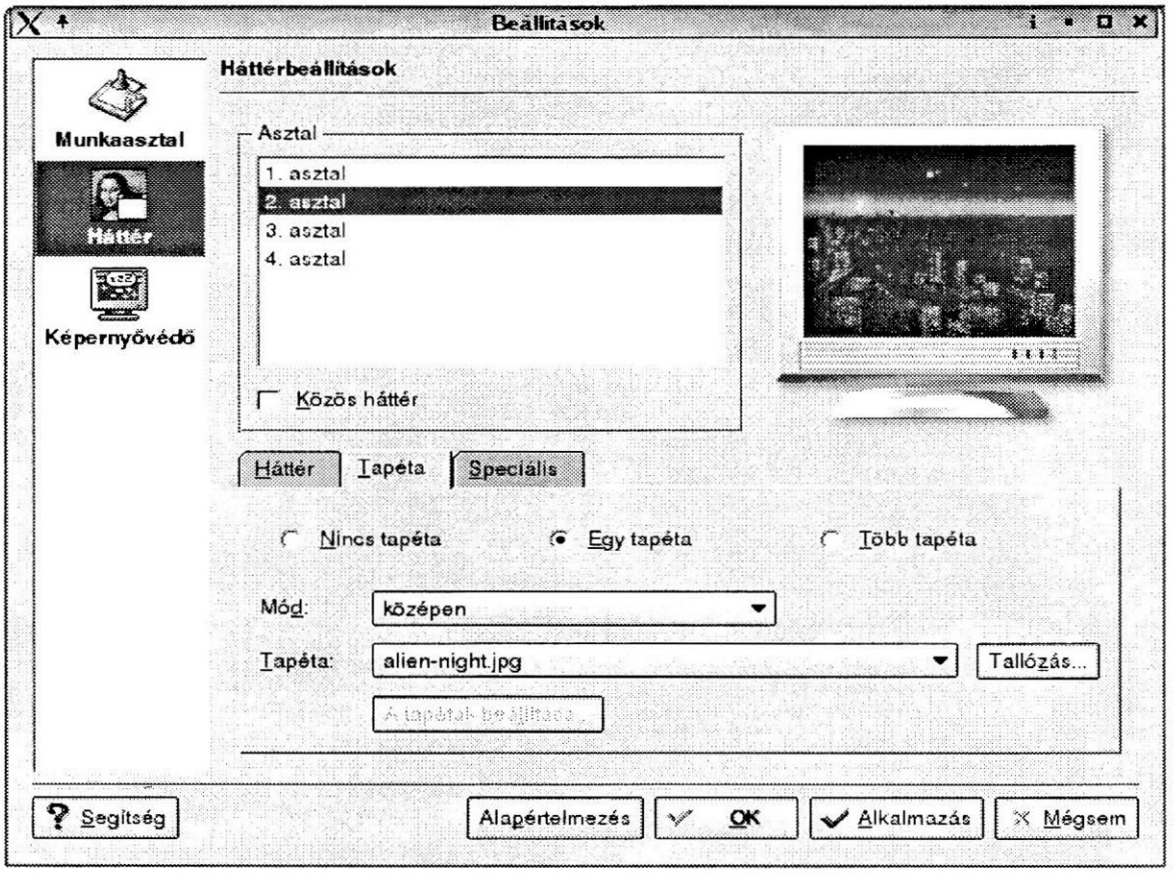
A dátum, illetve idő beállítása a KDE beállítómodul segítségével történhet, melyet akár a K menü Rendszer menüjéből, akár magára az órára jobbegérrel kattintva a helyi menüből is meghívhatunk:

A megfelelő idő beállításához el kell döntenünk, hogy a BIOS-ban a helyi vagy a greenwich-i középidőt (GMT vagy UTC) akarjuk tárolni. Ha a Linux az egyetlen operációs rendszer a gépünkön, a második megoldás a célszerűbb. Ekkor nem kell évente kétszer állítgatni az órát a nyári-téli időszámítás miatt. Ha Windows rendszerünk is van, akkor mindkét időre szükségünk lesz, hogy mindkét rendszeren a megfelelő idő jelenjen meg.

Egy SuSE karakteres felületen az ennek megfelelő beállításhoz az /etc/sysconfig/clock fájlban kell megnézni a GMT változó értékét, illetve a TIMEZONE változót, ami jó esetben Europe/Budapest értékű. Az /etc/localtime fájl az /usr/share/zoneinfo/Europe/Budapest másolata. Ha a BIOS-ban állítjuk az időt, a GMT esetén télen 1, nyáron 2 órát kell kivonni a magyarországi időből. Ha minden beállítás korrekt, a date parancs a helyi időt télen CET, nyáron CEST időzóna-megnevezéssel írja ki. A greenwich-i középidőt a date parancs -u opciós változata mutatja meg.

7.4.2. A képernyő beállítása

KDE alatt a K menü Rendszer menüjének Beállítások részében érhető el a KDE Vezérlőközpont. Itt a Perifériák részben a Képernyő hivatkozással nyithatjuk meg a képernyő beállításait tartalmazó ablakot. Itt tehetjük meg az úgynevezett XFree86 beállítást, a K X Konfigurációt. Ez magában foglalja a videókártya, a monitor, a billentyűzet, a mutatóeszköz, a képernyő és az X általános jellemzőinek finomhangolását.



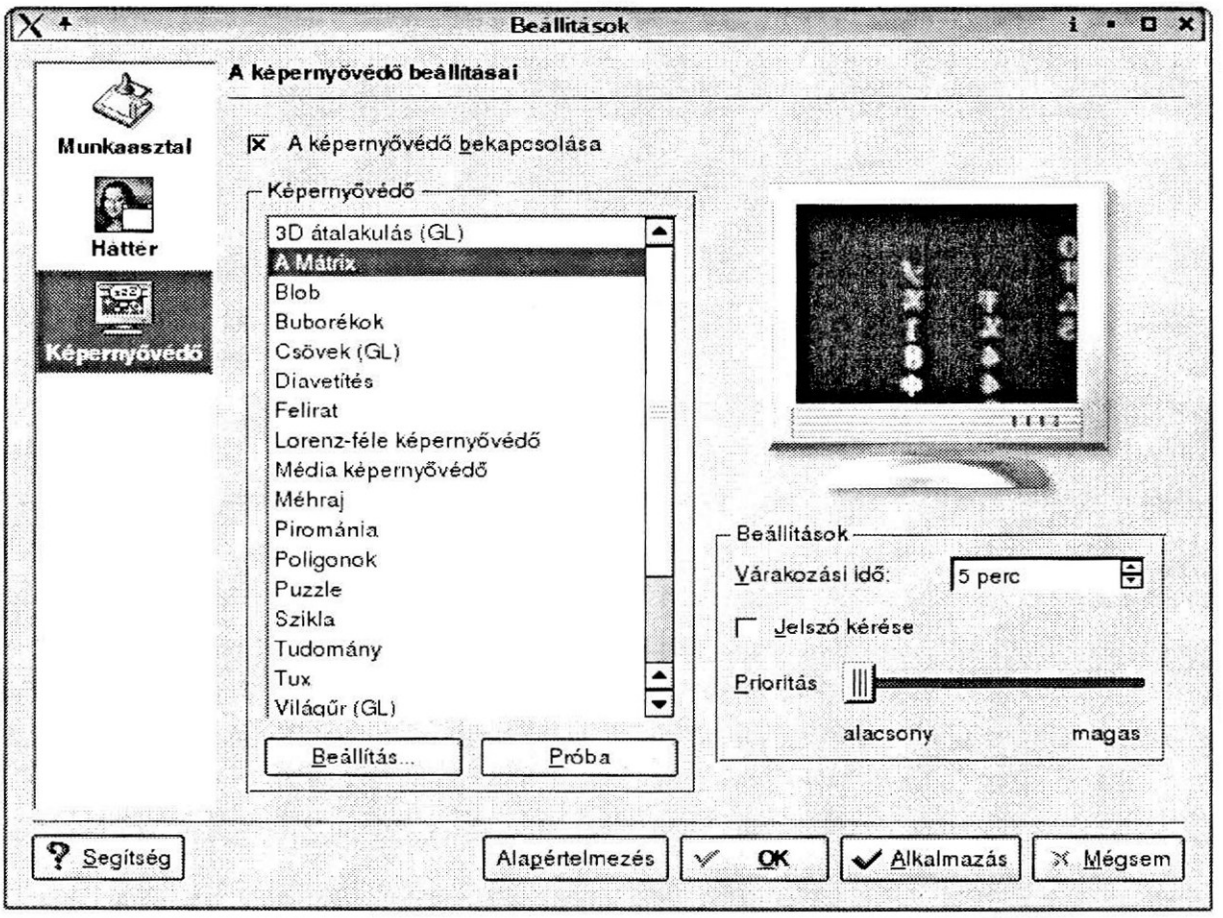
Ha a háttéren a jobb egérgombbal kattintunk, úgy A munkaasztal beállítása menüponttal előhívhatjuk az asztal konfigurálására szolgáló panelt. A panel bal szélén tudunk váltani a beállítandó funkciók között. Ha itt kattintunk a háttérre, úgy lehetőségünk van a képernyő hátterének tulajdonságait módosítani.

Mivel linuxban több munkaasztalt is használhatunk, ezért elsőként a módosítandó asztalt kell kattintással kiválasztani.

Ha nem akarunk háttérképet, csupán egyszerű mintát, úgy a panel alsó részében lévő háttér fülnél tudjuk a háttér tulajdonságait módosítani. Háttérkép esetén kattintsunk a Tapéta fülre, válasszuk ki a kívánt fájlt, illetve állítsuk be a megjelenési módokat.

A beállított hátteret a jobb felső sarokban kis méretben ellenőrizhetjük.

Amennyiben azt szeretnénk, hogy egy bizonyos ideig tartó tétlenség esetén bekapcsoljon a képernyővédő is, úgy a panel bal oldalán kattintsunk a képernyővédő ikonra.



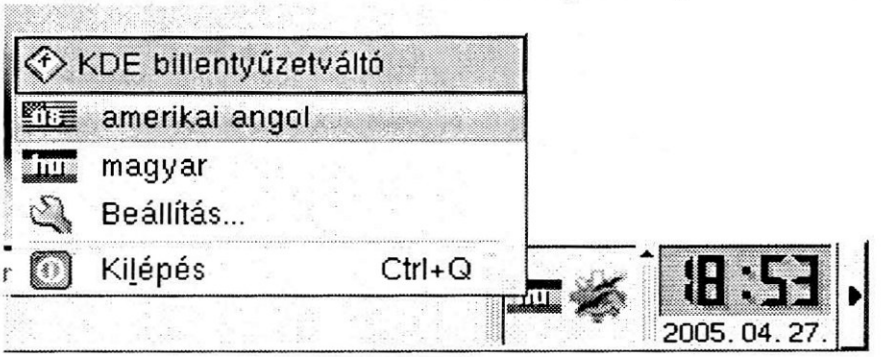
Most már ki tudjuk választani a kívánt képernyőkimélő funkciót a panel közepén lévő listából. A bekapcsoláshoz szükséges időt, illetve a visszakapcsoláskor történő jelszókérést a panel jobb oldalán állíthatjuk be.

A beállítás gomb az adott képernyővédő részletes finomhangolására, a Próba gomb pedig annak megtekintésére nyújt lehetőséget.

A grafikus képernyő alkalmzott felbontásértékének növeléséhez, illetve csökkentéséhez használhatjuk a Ctrl Alt + illetve a Ctrl Alt - billentyűkombinációkat.

7.4.3. A billentyűzet beállítása

A billentyűzet különféle jellemzőinek beállításait több menüből is elérhetjük. Az egyik legfontosabb tulajdonság, a billentyűzet kiosztása a panel jobb szélén látható zászló ikonról jobb egérgombbal előhívott helyi menüből az alternatív kiosztás(ok)ra egyetlen kattintással átválthatunk. Érdemes alapkiosztásra a magyar billentyűzetet állítani (ha magyar a billentyűzetünk), másodlagosra pedig az amerikai angolt (speciális karaktereknél jól jöhet). Ha különlegesebb kiosztásra lenne szükségünk, a beállítást ugyanebből a menüből is elvégezhetjük. Másik lehetőségünk, hogy a K menüből előhívható Vezérlőközponton keresztül végezzük el a kívánt módosításokat.



7.4.4. Az X Window rendszer

Az X Window egy hálózaton keresztül is használható ablakos rendszer. Az X Window rendszer-szerverek bitképes kijelzést alkalmazó számítógépeken futnak. Az X szerver számos különböző folyamatközi kommunikációs csatornán keresztül fogadja a felhasználói bemeneteket és kezeli a különféle kliens programok kimenet-kéréseit. Bár a legtöbb esetben a kliens programok ugyanazon a gépen futnak, mint a szerver, a kliensek más gépek számára is láthatóan működhetnek (a más architektúrájú és operációs rendszerű gépeket is beleértve).

Az X rendszer támogatja az átlapolásos hierarchikus alablakokat, a szöveges és grafikus műveleteket (nemcsak színes, de egyszínű kijelzőkön is). Az X szerver által használt X kliens programok száma meglehetősen nagy. Az alapvető X konzorciumos disztribúció néhány fontosabb programja:

- xterm: egy terminál emulátor
- twm: egy minimalizált ablakozó
- xdm: egy megjelenítő menedzser
- xconsole: egy konzolátirányító program
- bitmap: egy bitképes szerkesztőprogram
- xauth, xhost, iceauth: hozzáférés-vezérlők
- xset, xmodmap: beállító programok
- xclock: óra-alkalmazás

- xlsfonts: font- és fontinformáció-megjelenítő
- xfs: betűtípus-szerver

...

7.4.4.1. Kijelzőnevek

A felhasználó szemszögéből minden X szerver rendelkezik egy kijelzőnévvel:

hosztnev:kijelzoszam.kepernyoszam

Ezt az információt használja az alkalmazás annak megállapítására, hogyan tud kapcsolódni az X szerverhez és melyik képernyőt szeretnénk alapértelmezésben használni (többképernyős rendszereken):

- hosztnev: a hosztnév határozza meg annak a kliens gépnek a nevét, melyhez a kijelző fizikailag csatlakozik. Ha a hosztnevet nem adjuk meg, a leghatékonyabb kommunikációs mód lesz érvényben.
- kijelzoszam: a kijelző ebben az esetben egy olyan monitor-gyűjteményre vonatkozik, melyek egy közös billentyűzet vagy pozicionáló eszköz (pl. egér) reakcióit mutatja. A legtöbb munkaállomásnak természetesen csak egyetlen billentyűzete és egyetlen kijelzője van. Nagyobb, többfelhasználós rendszereken azonban gyakran alkalmaznak több kijelzőt, így egyszerre egynél többen is végezhetnek grafikus munkát. A kijelzők összekeverésének elkerülésére minden készülék kap egy kijelzoszam-ot (0-tól kezdve), amikor a kijelző X szervere elindul. A kijelzoszam mindig egy kijelzőnévben van megadva.
- kepernyoszam: néhány kijelző egyetlen billentyűzetet és pozicionálót oszt meg kettő vagy még több monitor között. Mivel minden monitornak megvan a saját ablak-csoportja, minden kijelző kap egy kepernyoszam-ot (0-tól kezdve), amikor a kijelző X szervere elindul. Ha nem adunk meg képernyőszámot, a 0-ás képernyő lesz használva.

A POSIX rendszereken az alapértelmezett képernyő neve a DISPLAY környezeti változóban van tárolva. A változó tartalmát az xterm terminál emulátor automatikusan állítja be. Ha egy másik hálózat gépére jelentkezünk be, szükségünk lehet a DISPLAY kézi beállítására a képernyőnk meghatározásához.

7.4.4.2. Ablak- és asztalmenedzserek

Az ablakok képernyőn történő megjelenését speciális programok, az úgynevezett ablakmenedzserek végzik. Bár számos ablakmenedzser elfogadja az adott geometria-specifikációkat, mások le is tilthatják azokat (például akkor, mikor a felhasználó explicit módon akar rajzolni az ablak területén a pozicionálóval).

Mivel az ablakmenedzserek szokványos (de komplex) kliens programok, számos különböző felhasználói interfész készült. Az X konzorcium disztribúciója például egy twm nevű ablakmenedzsert használ. A legtöbb felhasználó azonban valami más, a felhasználó hangulatához és stílusához jobban illeszkedő ablakozót részesít előnyben, ha a rendszererőforrások azt lehetővé teszik (pl. Sawfish, Enlightment stb.).

Az asztalmenedzser valamely ablakmenedszer használatával teszi kényelmessé grafikus felületünk használatát menükkel, üzenetekkel, órával, egy programmenedzserrel, egy fájlmenedzserrel és hasonlóan fontos programokkal. A legnépszerűbb ablakmenedszerek a Gnome és a KDE. Mindkettő szinte az összes Linux disztribúción (és számos más UNIX rendszeren) futtatható.

A KDE alkalmazások futtathatók Gnome alatt is. Ha tehát KDE alkalmazásokat szeretnénk futtatni, nem kell feltétlenül a KDE-felületet használni. Ha telepítettük a KDE-könyvtárakat (a kdelibs csomagot), ezen alkalmazásokat a Gnome menükből vagy a Gnome terminálról is futtathatjuk.

Fordítva kissé komplikáltabb a helyzet, mert Gnome-ban nincs egyszerű alapkönyvtár-halmaz. Ha ilyen esetben kiegészítő csomagokat kell telepítenünk, erről az alkalmazás futtatásakor (vagy már a telepítésekor) tudomást szerzünk.

7.4.5. Az X szerver konfigurációja

A Linux X disztribúciója, az XFree86 az XF86Config konfigurációs fájlt használja az inicializációs beállításokhoz. Ez a fájl konfigurálja videókártyánkat és több helyen is keresi azt (bár a legtöbbször az /etc/X11-ben van). A fájl számos részből áll, melyek bármilyen sorrendben lehetnek:

- Device: a grafikus eszköz leírása
- DRI: 3-D beállítások
- Files: fájl-elérési utak
- InputDevice: beviteli eszköz-leírás
- Modes: videómódok leírása
- Module: dinamikus modul-betöltés
- Monitor: a monitor leírása
- Screen: a képernyő konfigurációja
- ServerFlags: szerveropciók
- ServerLayout: megjelenés
- VideoAdapter: videóadapter leírása

Teljes leírást az XF86Config Info és man oldalai adnak. Felhasználóként valójában nem nagyon kell foglalkoznunk a fájl tartalmával, mivel a telepítéskor minden beállítás el lett végezve. Ha valamilyen okból kifolyólag mégis szükség lenne a grafikus szerver-beállítások megváltoztatására, futtathatjuk a konfigurációs eszközöket vagy szerkeszthetjük is azokat a konfigurációs fájlokat, me-

lyek az XFree86 szerver használatához a beállításokat elvégzik. Az egyes disztribúciók saját eszközeiket kínálják az ilyen feladatok elvégzéséhez (érdemes megnézni a man oldalakat). Mivel a helytelen konfigurálás olvashatatlan adatokhoz vezethet grafikus módban, feltétlenül készíteni kell egy biztonsági mentést az XF86Config fájlról annak módosítása előtt.

7.5. Multimédia

7.5.1. Hangkártya-konfigurálás

A legtöbb Linux disztribúció kínál hangkártya-tesztelő eszközöket. A legtöbb jelenlegi Plug & Play hangkártyát a rendszer automatikusan felismeri. Ha a mintahangokat halljuk, nincs más dolgunk, mint az OK-ra kattintani és minden beállítás megtörténik.

Ha hangkártyánkat a rendszer nem ismeri fel automatikusan, kapunk egy hangkártya-listát és/vagy hangkártya-tulajdonságokat, melyekből választhatunk. Ezután meg kell adnunk a megfelelő I/O portot, az IRQ és a DMA beállításokat. A beállításokról hangkártyánk leírása adhat információkat. Ha a Linux mellett Windows operációs rendszert is használunk, a Windows Vezérlőpultban is megtalálhatjuk a megfelelő beállításokat.

7.5.2. Zenelejátszás

7.5.2.1. CD-lejátszás

A cdp csomagot a legtöbb disztribúció tartalmazza, mely többek között egy szöveg-alapú CD-lejátszó programot foglal magába (cdp vagy cdplay). Az ablakozók általában egy grafikus eszközt is kínálnak erre a célra (pl. a gnome-cd lejátszó Gnome alatt). Ezek a menüből egyszerűen indíthatók.

Érdemes megjegyezni, hogy Linux-ban óriási különbség van az Audio CD és az adatlemezek használata között. Az adatlemezt be kell csatolni a fájlrendszerbe, hogy használni tudjuk a rajta levő adatokat. Az Audio CD-kkel más a helyzet. Mivel ezek tartalma nem olyan, mint a Linux fájlrendszer adatai, ezeket nem kell becsatolni a fájlrendszerbe, hogy hallgathassuk a zenét. Az Audio CD-t közvetlenül érjük el és küldjük az adatokat a kimeneti csatornára, egy CD-lejátszó program segítségével. Ha hangfájlokat tartalmazó lemezünk van, azt értelemszerűen be kell csatolni a fájlrendszerbe a használat előtt.

A csomag cdparanoia programja a zenét adatként, közvetlenül a CD-ről olvassa be, analóg konverziók nélkül. Az adatokat egy fájlba írja (különféle formátumok fordulnak elő, de a legnépszerűbb a .wav). Számos más formátumba is konvertálhatunk más eszközökkel, például .mp3-ba (a legtöbb disztribúció

tartalmazza vagy külön csomagban letölthető). A GNU pojekt számos CD-lejátszó, rippelő és kódoló eszközt kínál.

Az Audio CD létrehozása az idők során leegyszerűsödött. Ma már sok más eszköz mellett a KDE kaudiocreator programjával egyszerűen készíthetünk Audio CD-ket (ha a kdemultimedia csomag telepítve van a rendszeren).

7.5.2.2. MP3-lejátszás

A népszerű .mp3 formátumot a Linux gépeken is széles körben támogatják. A legtöbb disztribúció több MP3-lejátszót is tartalmaz. Az egyik legszélesebb körben elterjedt lejátszó az xmms.

Egyes disztribúciókon a konfiguráció módosítása nélkül nem lehet .mp3 fájlokat lejátszani (az MP3-programok licensze miatt). Az xmms lejátszóhoz az xmms-mp3 beépülő modulra is szükség van. Ha a hibaüzenetből látszik, hogy licensz-probléma áll fenn, az alábbi folyamatot kövessük:

- töltsük le és telepítsük az apt-get-et (pl. a www.freshrpms.net-ről)

  rpm -Uvh /var/tmp/xmms-mp3-1.2.7-21.p.fr1

- futtassuk az alábbi programot a rendszer apt csomagkezelővel való használatára történő felkészítéséhez:

  apt-get update

  apt-get upgrade

- Telepítsük az xmms-mp3 beépülő modult:

  apt-get install xmms-mp3

- Jelentkezzünk ki rendszergazdaként, hiszen a rendszer módosításait befejeztük. Készen is állunk kedvenc zenénk lejátszására:

  xmms /zene/konnyu/angyal.mp3

Az xmms akár menüből, akár parancssorból indítható. Utóbbi esetben az alábbi szintaktikát kell alkalmazni:

```
moni:~/mp3> xmms zene.mp3
```

Ha az adott könyvtárban található összes .mp3 fájlt le szeretnénk játszani, használjunk dzsóker-karaktert:

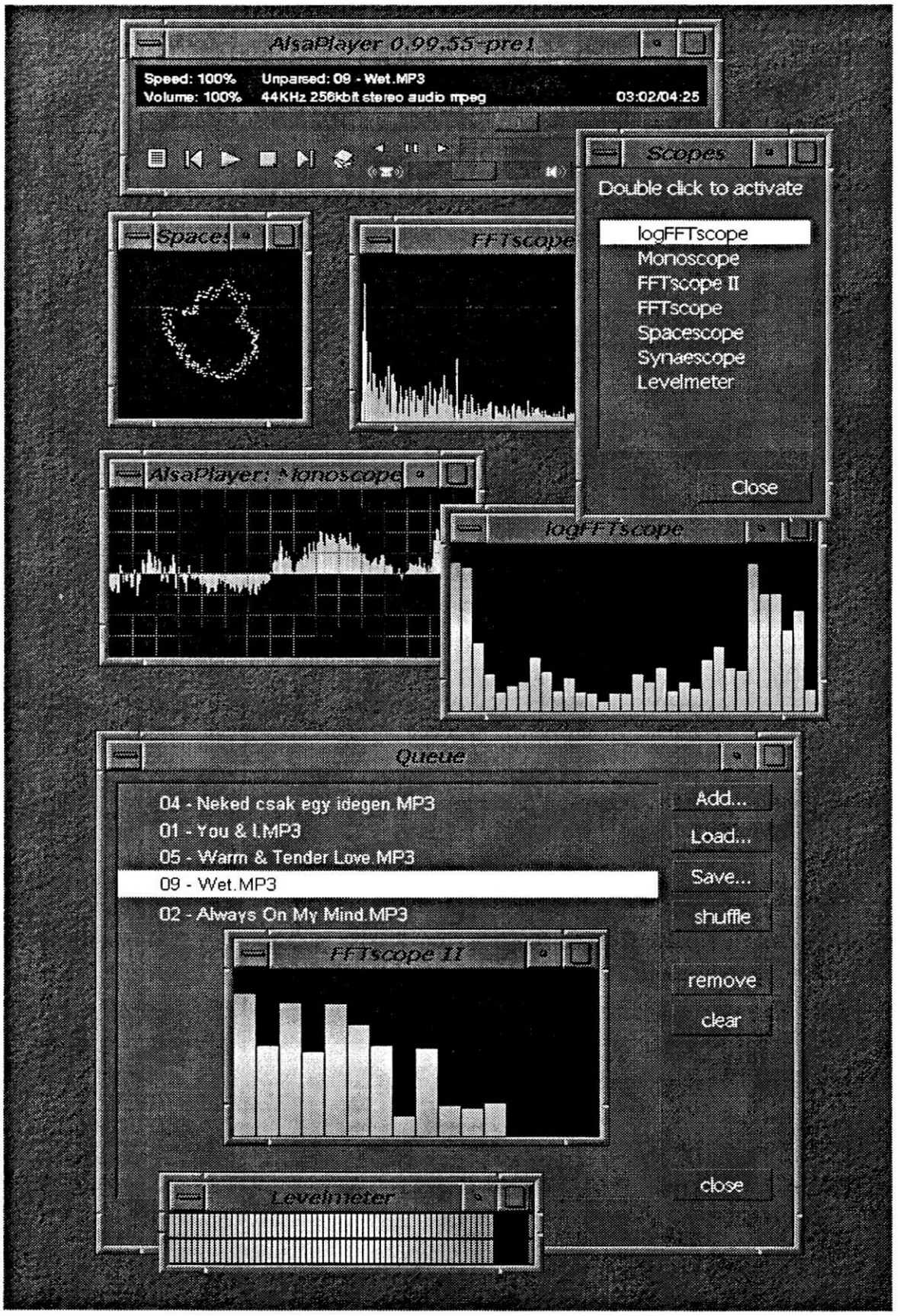
```
moni:~/mp3> xmms *.mp3
```

Ha jobban megnézzük a menüt, bizonyára észrevesszük, hogy a rendszer több grafikus MP3-lejátszót tartalmaz (akár Gnome-ban, akár KDE-ben dolgozunk).

7.5.2.3. Más formátumú hangfájlok lejátszása

Linux platformon az alábbi zenelejátszó és -manipuláló programok a legelterjedtebbek:

- Alsaplayer: az Advanced Linux Sound Architecture projekt lejátszója



- KMediaPlayer: a KDE multimédia-eszközök alapja
- mplayer: a legkülönfélébb fájlokat is lejátszó program
- Ogg Vorbis: az ingyenes audio-formátum, az .ogg fájlok lejátszására
- Playmidi: egy MIDI-lejátszó
- realplay: real audio és real video fájlok lejátszásához
- SoX (Sound eXchange): a play programmal előforduló hangkonverter. A .wav, .ogg és egyéb formátumokú fájlok mellett a nyers bináris formátumokat is lejátssza.

7.5.2.4. Hangerő-szabályozás

Az aumix vagy az alsamixer olyan szöveges eszközök, melyekkel szabályozhatjuk a hangerőt. Ha menüből indítjuk, az alsamixer-nek van grafikus interfésze is (illetve a parancssorban gnome-alsamixer-ként indítva). A kmix eszköz hasonlóan viselkedik KDE-n.

7.5.3. Videólejátszás

A zenelejátszáshoz hasonlóan a videólejátszáshoz is számos program közül választhatunk. A legfontosabbak:

- xine: ingyenes videólejátszó
- ogle: DVD-lejátszó
- mplayer: filmlejátszó
- gstreamer: audio- és videófelvétel, szerkesztés és lejátszás
- totem: audio- és videófájlok lejátszására (Audio CD, VCD, DVD)
- realplay: a RealNetworks terméke

A legtöbb esetben ezen programok valamelyike megtalálható a grafikus menüben.

7.6. Területi beállítások

7.6.1. A billentyűzet beállítása

A billentyűzet karakterkiosztását a loadkeys paranccsal állíthatjuk szöveges konzolokon. Grafikus módban a helyi X konfigurációs eszközt használhatjuk vagy manuálisan is módosíthatjuk az XF86Config billentyűzetre vonatkozó (Keyboard) részét. Az egyik beállítandó az XkbdLayout, melynek alapértelmezése legtöbb esetben az amerikai beállítás:

```
XkbLayout "us"
```

Helyi beállításainknak megfelelően szükséges módosítani az idézőjelek közötti részt (a megadható érték a /lib/kdb/keymaps alkönyvtárak valamelyikében

felsorolt név lehet, a .map.gz végződés nélkül). Ha nem találjuk a billentyűzetkiosztásokat, próbálkozzunk meg a
locate .map.gz

paranccsal. A billentyűzet beállításai kombinálhatók is, például:

Xkblayout "hu, us"

Az /etc/X11/XF86Config fájl szerkesztése előtt készítsünk róla biztonsági másolatot! (Értelemszerűen rendszergazdai jogosultságokra is szükség van.)

Jelentkezzünk ki, majd jelentkezzünk újra be az X beállítások betöltéséhez.

A Gnome, illetve a KDE felületén különleges jogok nélkül bárki könnyendén, valós időben átválthat a billentyűkiosztások között.

Ha Windows-os billentyűzetünk van, egyes billentyűk (pl. ⊞) egyáltalán nem vagy nem megfelelően működnek.

7.6.2. Betűtípusok

Szöveges módban a setfont segítségével tölthetünk be fontokat. A legtöbb rendszer tartalmazza a standard inputrc fájlt, mely lehetővé teszi a karakterek kombinálását (metakarakterek). A rendszergazda ehhez az

export INPUTRC="/etc/inputrc"

sort adja az /etc/bashrc fájlhoz.

7.6.3. Időzóna

Az idővel kapcsolatos beállításokat legtöbbször a telepítéskor kell megtenni. Ezután egy NTP (Network Time Protocol) kliens segítségével frissül. A legtöbb Linux rendszer az ntdp-t alapból futtatja:

```
lali:~> ps -ef | grep ntpd
ntp 25463 1 0 2005 ? 00:00:55 ntpd -U ntp
```

A legtöbb ablakozó különféle eszközökkel segíti az idő beállítását, feltéve, hogy rendszergazdai jogosultságokkal rendelkezünk.

7.6.4. Nyelvi beállítások

Ha a rendszer üzeneteit angolul vagy éppen dánul kapjuk meg, a LANG és a LANGUAGE környezeti változók tartalmát kell megváltoztatnunk. Így élvezhetjük az adott nyelv helyi támogatottságát, sőt a nyelv karakter-konvencióinak megfelelő betűtípusokat is.

A legtöbb grafikus beléptető rendszeren (gdm, kdm) arra is lehetőség van, hogy ezeket a nyelvi beállításokat még a bejelentkezés előtt elvégezzük.

Megjegyzendő, hogy a legtöbb rendszeren napjainkban az alapértelmezett beállítás az en_US.UTF-8. Ezeken a rendszereken az összes program támogatja az UTF-8 kódolást, így akár a vi, akár a cat programot használjuk, nem lehetnek gondjaink. Más a helyzet a magyar rendszerekkel, de nem szabad elfelejteni, hogy a legtöbb disztribúció nem kis hazánkban születik.

A régebbi rendszerekkel is lehetnek problémák, hiszen azok nem mindegyike ismeri az UTF-8 kódolást. Egyes régi rendszerek csak 1 bájtos karakterfontokat támogatnak. A recode program jól jöhet, ha egy másik karakterkészletre szeretnénk konvertálni egy fájlt. Másik lehetőségünk, hogy ideiglenesen egy másik kódolásra váltunk, melyet a LANG környezeti paraméterben állítunk be:

```
robi:~> acroread /var/tmp/preface.pdf
Warning: charset "UTF-8" not supported, using "ISO8859-1".
Aborted

robi:~> set | grep UTF
LANG=en_US.UTF-8

robi:~> export LANG=en_US

robi:~> acroread /var/tmp/preface.pdf
<--uj ablak nyilik-->
```

7.7. Új szoftver telepítése

A legtöbben meglepődnek, miután a Linux operációs rendszert feltelepítve számítógépükre egy működő, jól használható készülékhez jutnak. A disztribúciók többsége számos hardvereszköz (videókártya, hálózati kártya stb.) meghajtóprogramját tartalmazza. Emellett irodai programokat, böngészőket e-mail klienseket is magában foglal egy-egy ilyen telepítés. Ennek ellenére természetesen nem kell megelégednünk ezekkel a szoftverekkel.

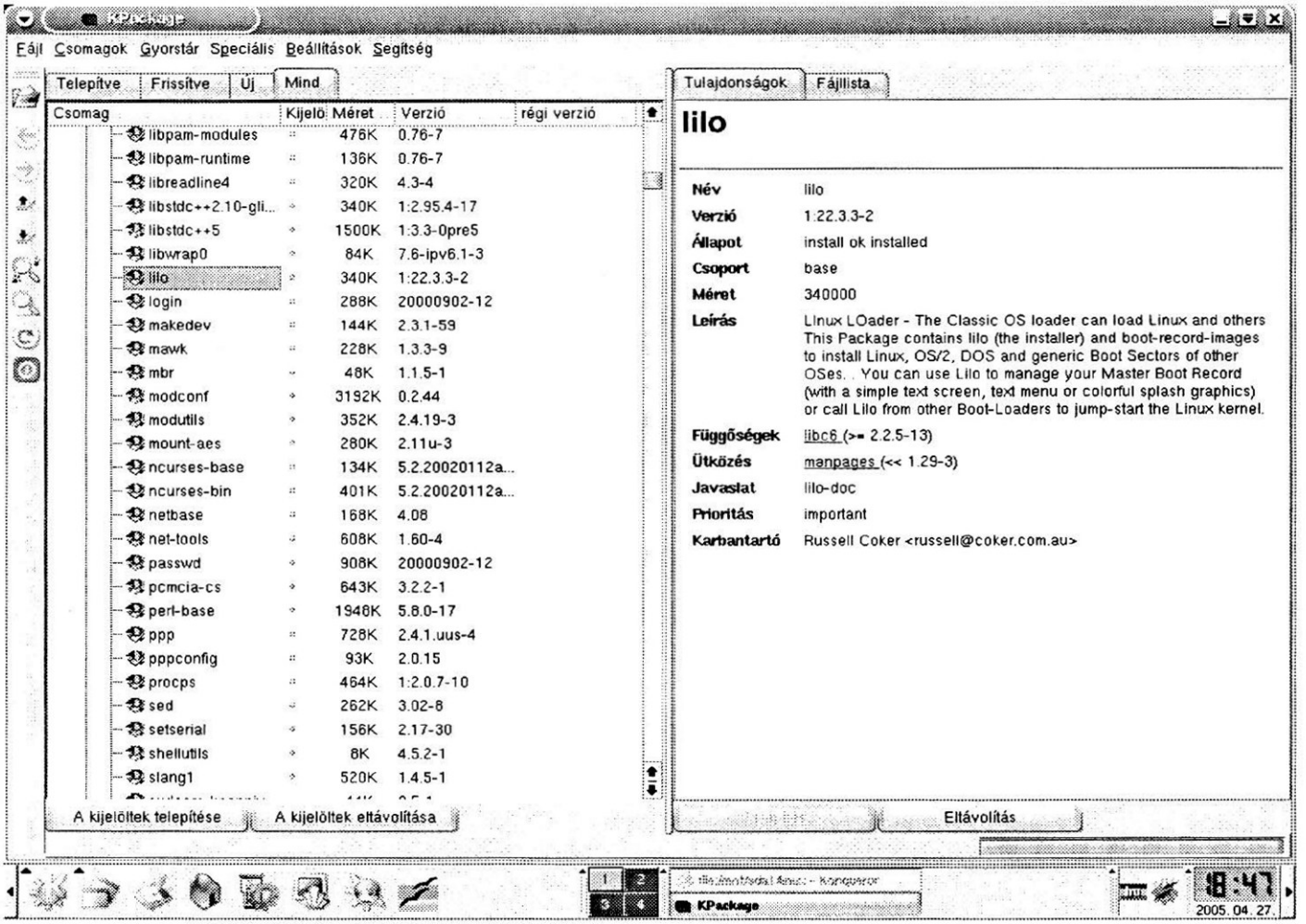
Ha nem találjuk meg, amit keresünk, lehet, hogy nincs telepítve rendszerünkre. Az is előfordulhat, hogy rendelkezünk a programmal, csak az nem úgy működik, ahogy azt szeretnénk. Mivel a Linux programok gyorsan fejlődnek, érdemes ilyenkor utánanézni, nem létezik-e a programnak újabb verziója. Fölösleges időközben már megoldott problémákkal vesződnünk.

A rendszert bármikor frissíthetjük és tetszés szerinti időpontban telepíthetünk új programokat. Számos program csomagokban jelenik meg. Egyes disztribúciók több CD-s változatai (pl. a SuSE Linux 8.0 Professional verziója 7 CD-s) további programokat is tartalmaznak (a telepítéskor esetleg csak a CD-k felét használtuk fel). Egy másik programforrás az internet. Érdemes felkeresni saját Linux disztribúciónk weblapját. Mindig olvassuk el a program dokumentációját! Sokszor találkozhatunk telepítési útmutatókkal is. Ezekre is

érdemes egy pillantást vetni. A programokhoz szokott lenni egy README (olvassel) fájl is, melynek elolvasása erősen ajánlott.

7.7.1. Telepítés grafikus felületen

KDE alatt a KPackage programmal egyszerűen adminisztrálhatjuk már meglévő, illetve telepítendő csomagainkat. A csomag fizikai mérete és rövid leírása mellett az esetleges függőségeket és ütközéseket is megtudhatjuk. Ha szerencsénk van, még a karbantartóval is felvehetjük a kapcsolatot.



7.7.2. Csomagformátumok

7.7.2.1. Az rpm csomagok

Az RPM (RedHat Package Manager) egy hatékony csomagkezelő, mely csomagok telepítéséhez, frissítéséhez és eltávolításához egyaránt felhasználható. Segítségével kereshetünk is csomagokat. Egy beépített biztonsági rendszernek köszönhetően az internetről letöltött csomagok eredetisége ellenőrizhető. A haladó felhasználók akár saját csomagokat is készíthetnek az RPM-mel.

Az RPM csomag egy fájlarchívumot és az archív fájlok telepítéséhez, illetve törléséhez szükséges metaadatokat tartalmaz. A metaadat súgóscripteket, fájlattribútumokat és egy csomagleírást foglal magába. A csomagoknak két formája létezik:

- bináris csomagok: a telepítendő programot tartalmazza
- forráscsomagok: a bináris csomagok előállításához szükséges forráskód és információk

Nevével ellentétben az RPM-et nemcsak a Red Hat disztribúció támogatja, hanem számos más változat is (SuSE, Mandrake stb.). Célszerű elolvasni disztribúciónk specifikus információit a man rpm paranccsal.

A legtöbb csomag egyszerűen a frissítő (upgrade) opcióval (-U) telepítődik, függetlenül attól, hogy a csomag telepítve volt-e korábban avagy sem. Az RPM csomag a program egy teljes verzióját tartalmazza, amely felülírja a létező változatot, illetve egy új csomagként kerül telepítésre. A tipikus használat az alábbi:

```
rpm -Uvh /utvonal/rpm-csomag(ok)
```

Ha részletesebb információkat szeretnénk, használjuk a -v opciót. A -h opció egy folyamatdiagramot is megjelenít:

```
[root@disz tmp]# rpm -Uvh siraly-0.56.6-1.hu.i386.rpm
Preparing...        ########################################### [100%]
1:siraly            ########################################### [100%]
[root@disz tmp]#
```

Az új kernel verziók telepítése a fentiektől kissé eltér, hiszen az -i opcióval a csomag létező változata(i) nem íródik (íródnak) felül. Ennek magyarázata, hogy abban az esetben, ha az új változat nem működik, a régivel a rendszer még mindig el tud indulni.

Az rpm segítségével ellenőrizhetjük, hogy egy csomag fent van-e már a rendszerünkön:

```
[goliat@disz ~] rpm -qa | grep vim
vim-minimal-6.1-29
vim-X11-6.1-29
vim-enhanced-6.1-29
vim-common-6.1-29
```

Még azt is megtudhatjuk, hogy melyik csomag tartalmaz egy adott fájlt vagy programot:

```
[goliat@disz ~] rpm -qf /etc/profile
setup-2.5.25-1

[goliat@disz ~] which cat
cat is /bin/cat

[goliat@disz ~] rpm -qf /bin/cat
coreutils-4.5.3-19
```

Fontos megjegyezni, hogy az RPM adatbázis lekéréséhez nincs szükség rendszergazdai jogosultságokra. Erre csak akkor van szükség, ha hozzáadni, módosítani vagy törölni szeretnénk csomagokat.

Végül nézzük meg egy csomag telepítését az rpm használatával:

```
[root@diszel root]# rpm -e siraly
[root@diszel root]#
```

Ha eltávolítjuk valamelyik csomagot, ne lepődjünk meg, hogy látszólag nem történik semmi. Az rpm -qa használatával ellenőrizhetjük, hogy a csomagot ténylegesen sikerült-e eltávolítani.

7.7.2.2. A deb csomagok

A deb a Debian Linux-disztribúciók alapértelmezett csomagformátuma. Itt a dselect segítségével kezelhetjük a csomagokat. Segítségével kijelölthetjük a telepítendő vagy frissítendő csomagokat. Egy Debian rendszer telepítése során is találkozunk vele. A dselect segít a használandó elérési módok definiálásában, az elérhető csomagok kilistázásában és a csomagok konfigurálásában.

7.7.2.3. A forráscsomagok

A Linux programok nagy része ingyenes és nyílt forráskódú szoftver, így ezen programokhoz forráscsomagok is elérhetők. A forrásfájlok segítségével egy saját program-verziót fordíthatunk. A kereskedelmi szoftverek tartalmazzák a forráskódot, de az adott program weblapjáról is letölthetjük a nekünk megfelelő forrást nyers formátumban (program-verzio.tar.gz vagy ehhez hasonló) vagy csomagban (rpm, deb).

A program különleges igényeit, megkötéseit, telepítési tudnivalóit a legtöbbször a README (olvassel) fájlban találjuk meg. A fordításhoz egy C fordítóra (pl. gcc) is szükségünk lesz. A GNU C fordító szinte az összes Linux alapszolgáltatásához tartozik.

7.7.3. A csomagkezelés automatizálása

Az első dolog, amit egy új rendszer telepítésekor teszünk, a frissítések alkalmazása.

A legtöbb Linux rendszer frissítése disztribúciónk egy viszonylag közeli oldalán található. A frissítéseket célszerű lenne minél gyakrabban (akár naponta) elvégezni. Minden csomagnál érdemes a legújabb változatot is kipróbálni. Már a korábbiakban is említettük, hogy nemcsak új tulajdonságokat és lehetőségeket tartalmazhat egy újabb programváltozat, de sokszor hibajavításokon is átesik a program és ekkor akár fontos biztonsági kérdésekben is fejlődhetett a szoftver. A Linux programok gyors fejlődése az örökké elégedetlen felhaszná-

lóknak és a kitartó programozóknak köszönhető. A rendszer igazi kihasználása azonban némi rugalmasságot kíván.

A frissítéseket Linux alatt nem úgy kell elképzelni, hogy naponta több tíz csomagot frissítünk manuálisan. A legtöbb disztribúció lehetővé teszi a frissítések automatizálását, ideértve még a forráscsomagok rendszeres frissítéseit is.

7.7.3.1. APT

Az APT az Advanced Package Tool (Fejlett csomagos eszköz) kifejezés rövidítése. A csomagok kezelésének parancssori eszköze az apt-get (egy nagyon jó man oldallal rendelkezik, ami a csomagok telepítését, frissítését, sőt még az egész disztribúció frissítését is leírja). Az APT gyökerei a Debian GNU/Linux-ig vezethetők vissza, ahol ez a Debian csomagok alapértelmezett kezelője. Az APT az RPM csomagokkal is jól működik. Az APT nagy előnye, hogy ingyenes és rugalmasan használható.

Az apt-get első használatakor általában szükségünk van az elérhető csomagok indexére. Ezt az

```
apt-get update
```

szintaktikájú paranccsal készíthetjük el. Ezután az apt-get használatával frissíthetjük a rendszert:

```
apt-get upgrade
```

Ha ezt gyakran elvégezzük, rendszerünk egyszerűen frissíthető lesz.

Az általános használat mellett az apt-get önálló csomagok telepítésénél is nagyon gyors. Szintaktikája:

```
[david@jupiter ~] su - -c "apt-get install xsnow"
Password:
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  xsnow
0 packages upgraded, 1 newly installed, 0 removed and 3 not upgraded.
Need to get 33.6kB of archives.
After unpacking 104kB of additional disk space will be used.
Get:1 http://ayo.freshrpms.net redhat/9/i386/os xsnow 1.42-10 [33.6kB]
Fetched 33.6kB in 0s (112kB/s)
Executing RPM (-Uvh)...
Preparing... ########################################### [100%]
1:xsnow ########################################### [100%]
```

Érdemes ismerni az su parancs -c opcióját, ami jelzi a root shell-nek, hogy csak ezt a parancsot futtassa, majd térjen vissza a felhasználói környezethez.

Ha más csomagoktól való függőség is fennáll, az apt-get letölti és telepíti ezeket.

7.7.3.2. Az RPM csomagokat támogató rendszerek

Az Update Agent (Frissítő), ami eredetileg csak a RedHat RPM csomagokat támogatta, mára már nagyobb támogatást kínál. Az eszköz egy teljes rendszert biztosít az RPM csomagok frissítésére. A parancssorban az up2date paranccsal frissíthetjük a rendszert. Grafikus felületen alapból egy kis ikon kerül aktiválásra, mely jelzi, hogy van-e elérhető frissítés rendszerünk számára.

Egy másik népszerű, Fedora rendszereken használt program a yum (Yellowdog's Updater Monitor). Interaktív, de ugyanakkor automatizált frissítő program RPM csomagok telepítéséhez, frissítéséhez, illetve eltávolításához.

A SuSE Linux-on mindent a YaST (Yet another Setup Tool) végez. Számos rendszeradminisztrációs programot kínál, melyek között az RPM csomagok frissítésére szolgáló eszköz is megtalálható. A 7.1-es verziótól kezdve SuSE Linux-ban egy webes interfész és a YOU (Yast Online Update) használatával is frissíthetünk.

A Mandrake Linux-on az úgynevezett URPMI eszközökkel találkozunk, melyek megkönnyítik az új programok telepítését a felhasználónak. Ezek a programok az RPMDrake és a MandrakeUpdate kombinálásával minden telepítéssel kapcsolatos kérdést megoldanak. A MandrakeOnline egyre bővülő szolgáltatási kört nyújt. Rendszergazdánkat automatikusan értesíthetjük, ha Mandrake rendszerünkhöz új frissítések vannak.

A KDE és a Gnome felületek is rendelkeznek saját (grafikus) csomagkezelőkkel (GnoRPM, KPackage).

A grafikus Linux programok egyik fejlesztője, a Ximian készítette a RedCarpet nevű grafikus eszközt. Segítségével különböző frissítési „csatornákra" iratkozhatunk fel. A különféle Linux disztribúciókhoz és verziókhoz különböző frissítési „csatornák" léteznek. A RedCarpet jelenleg a RedHat, a SuSE, a Debian, a Mandrake, a Solaris és a YellowDog rendszereket támogatja.

7.7.4. A kernelünk frissítése

Mivel a Linux disztribúciók komoly fejlesztéseken esnek át, érdemes időről időre frissíteni őket. A frissítési folyamat egy új kernelt telepít (ha szükséges) és az összes szükséges változtatást elvégzi a rendszeren. Csak le kell fordítanunk vagy manuálisan kell telepítenünk egy új kernelt, ha olyan kernel tulajdonságokra van szükségünk, melyeket Linux disztribúciónk alapértelmezett kernele nem támogat.

Akár egy saját magunknak optimatizált kernelt fordítunk, akár egy előfordított kernelcsomagot használunk, úgy telepítsük, hogy a régi kernel is működjön, amíg meg nem győződünk róla, hogy az új változat tökéletes.

Készítsünk egy két rendszeres indítási rendszert, hogy választhassunk a kernelek közül. Ehhez a rendszerindítási konfigurációs fájlt (grub.conf) kell átírnunk. Nézzünk egy példát:

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making config changes.
# NOTICE: You have a /boot partition. This means that
#         all kernel and initrd paths are relative to /boot/, e.g.
#         root (hd0,0)
#         kernel /vmlinuz-version ro root=/dev/hde8
#         initrd /initrd-version.img
#boot=/dev/hde
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Linux new (2.4.9-31)
        root (hd0,0)
        kernel /vmlinuz-2.4.9-31 ro root=/dev/hde8
        initrd /initrd-2.4.9-31.img
title old-kernel
        root (hd0,0)
        kernel /vmlinuz-2.4.9-21 ro root=/dev/hde8
        initrd /initrd-2.4.9-21.img
```

Érdemes pár napig tesztelni az új változatot, s ha az megfelelően működik, távolítsuk el a sorokat a fájlból és ekkor már a régi kernelre sincs szükség.

7.7.5. További csomagok telepítése a telepítő CD-kről

A további csomagok telepítése a telepítő CD-kről hasonlóan zajlik, mint a csomagok manuális telepítése. A legfontosabb eltérés, hogy a CD fájlrendszerét be kell kapcsolnunk (mount-olnunk) a gépünk fájlrendszerébe, hogy használni tudjuk a CD tartalmát. A legtöbb rendszeren ez egy CD behelyezésével automatikusan megtörténik, mivel az automount démon a rendszerbetöltés idején elindul. Ha azonban CD-nk automatikusan nem érhető el, az alábbihoz hasonló sorra lesz szükségünk:

```
mount /dev/cdrom /mnt/cdrom
```

Egyes rendszereken kizárólag a rendszergazda használhat eltávolítható lemezeket, ezt a konfiguráció határozza meg.

Automatizálási célokból a CD-meghajtó általában az /etc/fstab-ban elérhető, ami a fájlrendszereket és azok becsatolási pontjait tartalmazza.

```
[balazs@szhely ~] grep cdrom /etc/fstab
/dev/cdrom      /mnt/cdrom      iso9660         noauto,owner,ro 0 0
```

Ez jelzi, hogy a rendszer megérti a mount /mnt/cdrom parancsot. A noauto a rendszernek azt jelenti, hogy a CD-k becsatolásáról nekünk kell gondoskodnunk.

Ha fájlkezelőnk nem teszi meg nekünk, a CD becsatolásával grafikus felületen a CD ikonra történő jobbegeres kattintással is megpróbálkozhatunk. Ellenőrzéshez adjuk ki a mount parancsot argumentumok nélkül:

```
[balazs@szhely ~] mount | grep cdrom
/dev/cdrom on /mnt/cdrom type iso9660 (ro,nosuid,nodev)
```

Miután becsatoltuk a CD fájlszerkezetét, könyvtárat válthatunk. A becsatolási pont általában az /mnt/cdrom, ahonnan elérhető a CD-ROM tartalma. Ekkor a korábbiakban már megismert rpm használatával nekiállhatunk az új csomagok telepítésének.

A CD-t használata után nem távolíthatjuk el a meghajtó eject gombjával. Mivel a becsatolási pont alkönyvtárai a fájlrendszer részét képezik, a CD tartalmát először le kell csatolnunk. Ehhez először kiadhatjuk a cd parancsot argumentumok nélkül, így a saját home könyvtárunkba kerülünk. Ezután az

```
umount /mnt/cdrom
```

vagy az

```
eject cdrom
```

paranccsal csatolhatjuk le a CD-t.

A meghajtót sohase erőltessük (normál esetben ne használjuk a vészesetre készült nyílást a CD-meghajtón)! Akik megszokták egy másik operációs rendszer használatát, azoknak igencsak furcsa lehet a becsatolás és lecsatolás szükségessége (blokkolt meghajtók). Ennek megértéséhez a Linux filozófiáját kell jobban megismernünk. Ha drasztikus módon távolítanánk el a CD-t, a rendszer úgy kezelné, mintha még mindig a meghajtóban lenne. Ekkor a legjobb a rendszert újraindítani.

Ha a meghajtó foglaltságát jelző üzeneteket kapunk, ellenőrizzük, hogy egyetlen (grafikus) program sem használja az időközben eltávolított CD tartalmát. Ha bizonytalanok vagyunk, használjuk az lsof programot.

8. NYOMTATÁS

8.1. Egyszerű nyomtatás

A RedHat része az LPRng, ami a UNIX nyomtatási rendszer továbbfejlesztett változata. Ha a nyomtató be van állítva, az egyetlen dolgunk az lpr használatának megtanulása, s máris küldhetjük fájljainkat a nyomtatóra. A szintaktika igen egyszerű:

lpr fajl(ok)

Az lpr az lpd démont használja a nyomtatáshoz, ha az erőforrások azt lehetővé teszik. Ha nem adunk meg fájlneve(ke)t, a standard bement lesz érvényben (ha például egy parancs kimenetét szeretnénk a nyomtatóra küldeni). Az lpr parancsnak számos opciója van, melyekről a --help opció ad részletesebb felvilágosítást (a teljes leírást az Info oldalak tartalmazzák). Rendszerünkön megtalálhatjuk az lp parancsot is, ennek kompatibilitási okai vannak. Egyébiránt az lp egy szimbolikus hivatkozás az lpr-re:

```
peti:~> ls -l /usr/bin/lp*
lrwxrwxrwx    1 root     root            3 Aug 29  11:04 /usr/bin/lp -> lpr
-rwxr-xr-x    1 lp       lp         395192 Jun 12  2004 /usr/bin/lpq
-rwxr-xr-x    1 lp       lp         408536 Jun 12  2004 /usr/bin/lpr
-rwxr-xr-x    1 lp       lp         392984 Jun 12  2004 /usr/bin/lprm
-rwxr-xr-x    1 root     root         4651 Aug 20  21:58 /usr/bin/lprsetup.sh
-rwxr-xr-x    1 lp       lp         398488 Jun 12  2004 /usr/bin/lpstat
peti:~> ps -ef | grep lpd
lp 1003 1 0 Feb13 ? 00:00:00 lpd Waiting
```

Ha egyszer egy fájlt elküldtünk a nyomtatási sorba, a nyomtatási feladat kap egy azonosítót:

```
peti:~> lp /etc/profile
request id is peti@kapuvar+255
```

A nyomtatási sor megtekintéséhez használhatjuk az lpq parancsot. Ha argumentumok nélkül hívjuk meg, az alapértelmezett nyomtatási sor tartalmát jeleníti meg:

```
peti:~> lpq
Printer: lp@kapuvar
Queue: no printable jobs in queue
Status: job '0502disz.kapuvar.hu' removed at 11:13:55.123
```

A nyomtatási feladatokat törölhetjük is az lprm paranccsal. Ha csak egyetlen nyomtatást szeretnénk visszavonni, a nyomtatási feladat számát kell megadni az lprm argumentumaként.

Nagyobb környezetekben a többszörös nyomtatók kezeléséhez használhatjuk az lpc parancsot is.

Számos grafikus program nyomtatási funkciója az lpr használatával működik.

8.2. Formázott dokumentumok nyomtatása

Ha egy valamirevaló dokumentumot szeretnénk nyomtatni, azt először formázni kell.

A legtöbb jelölő nyelv inkább a nagy és ismétlődő feladatokat támogatja, míg az egyszerű szövegek megjelenéséért a számítógép a felelős (például scriptek futtatásakor). Ha formázott szövegeket szeretnénk nyomtatni, akkor sem maradunk azonban magunkra, hiszen számos program és rendszer közül választhatunk. A legfontosabbak a következők:

- A groff a UNIX-os roff parancs GNU változata. Alapból a troff parancsot futtatja és a kiválasztott eszközhöz utófeldolgozást végez. PostScript fájlok létrehozását is engedélyezi.
- A TeX a LaTeX kistestvére, mely az egyik legszélesebb körben használt jelölőnyelv a UNIX rendszereken. Meghívása általában a tex paranccsal történik. A dokumentum egy eszközfüggetlen változatát adja. A műszaki szövegeket gyakran írják LaTeX-ben, mivel támogatja a matematikai formulákat is, bár a webes ajánlásokat kiadó W3C Konzorcium más technológiákkal hasonló lehetőségeket kínál.
- SGML, XML: Linux alá ingyenes elemzők is beszerezhetők.
- pdf2ps, fax2ps, a2ps: parancssori eszközök
- StarOffice, OpenOffice, AbiWord, KWord, Applix, WordPerfect: grafikus eszközök

Ha man oldalakat szeretnénk nyomtatni (melyek előformázott troff adatokat tartalmaznak), azokat is formázni kell. A nyomtatás a -t opcióval lehetséges:

```
man -t parancs>man-parancs.ps
```

Ekkor a PostScript fájlt nyomtatjuk ki. Ha rendszerünkhöz vagy bejelentkezésünkhöz van megadva alapértelmezett nyomtatási cél, elegendő a man -t parancs kiadása is (ekkor a formázott szöveg közvetlenül a nyomtatóra megy).

8.2.1. Formázott szövegek nyomtatási előképe

Mindent, amit a nyomtatóra tudunk küldeni, a képernyőre is küldhetjük. A fájlformátumtól függően az alábbi lehetőségeink vannak:

- PDF fájlok: acroread vagy xpdf
- PostScript fájlok: gv (GhostView)
- TeX dvi fájlok: xdvi

8.3. Nyomtatás szerver oldalon

Néhány éve a szerver-oldali nyomtatás Linux-ban úgy nézett ki, hogy mindenki a régi öreg LPD-t futtatta. Már számos rendszer közül választhatunk. Egyes disztribúciók az LPRng-t tartalmazzák, ami az eredeti UNIX Line Printer Daemon-ra (LPD) vezethető vissza. Az LPD azonban a hálózati nyomtatási protokoll neve is. A hálózati protokollt nemcsak maga az LPD ismeri, de lényegében minden hálózati nyomtatószerver és hálózati nyomtató is.

Az LPRng sokkal jobb megvalósítás, mint az LPD. Könnyebben megvalósíthatók az egyes műveletek, másrészt jól dokumentált is a rendszer.

Az LPRng a nagyobb környezetekben jobban boldogul (egynél több nyomtatóval, különleges nem lpd-s hálózati nyomtatóval stb.), forráskódja pedig biztonságosabb, mint az LPD-é. Nincsenek SUID bináris adatai, ráadásul a hitelesítést is támogatja (PGP, Kerberos).

Egy érdekes új projekt a Common UNIX Print System (Közös UNIX Nyomtatási Rendszer), az Internet Printing Protocol (IPP) egyik implementációja, egy HTTP-szerű RFC standard csereprotokoll az LPD-protokoll helyett. A CUPS GNU GPL licensz alatt szerezhető be. A CUPS a MacOS X alapértelmezett nyomtatási rendszere is.

8.3.1. Grafikus nyomtató-konfigurálás

A legtöbb disztribúció tartalmaz egy grafikus felületet a helyi (párhuzamos) és hálózati nyomtatók konfigurálásához. Kiválaszthatjuk a nyomtatónk típusát egy felsorolásból és egyszerű teszteket végezhetünk. Nem kell vesződnünk a konfigurációs fájlok helyével, tartalmának szintaxisával. A nyomtató telepítése előtt érdemes megnézni a rendszer dokumentációját.

8.3.2. Nyomtató vásárlása Linux-hoz

A Linux kernel lehetővé teszi, hogy bármely, a soros, párhuzamos vagy USB portra csatlakoztatott nyomtatóval, illetve a hálózati nyomtatókkal kommunikáljunk. Ez azonban önmagában kevés. Ahhoz, hogy nyomtatni tudjunk a printerrel, olyan adatokat kell küldenünk neki, melyet az megért.

A Linux szinte az összes IBM- és/vagy HP-kompatibilis nyomtatót támogatja. Általában a Windows NT vagy a UNIX alapú rendszereken működő

nyomtatók Linux alatt is működnek. Ha csak Win9x meghajtóprogrammal kiadott nyomtatónk van, problémáink lehetnek Linux alatt a nyomtató használatával.

Legjobb választásunk egy PostScript-támogatású nyomtató lenne, hiszen az összes Linux program nyomtatható kimenetet produkál PostScript formátumban. Mivel a PostScript technológia eszközfüggetlen, használatával nem lehetnek nyomtatási gondjaink.

8.4. Nyomtatási problémák

A következőkben azt nézzük meg, mit tehetünk felhasználóként, ha valami nem megfelelően működik. A nyomtatási szolgáltatások démon-részével itt nem foglalkozunk, az egyébként is a rendszergazdákra tartozik.

8.4.1. Hibás fájl

Ha hibás fájlt nyomtatunk, a nyomtatási feladat az lprm azonosito parancscsal állítható le (az azonosito a nyomtatonev-nyomtatasi_feladat_ szama formájú megadást jelenti, melyet az lpq információi alapján tudunk meg). Ez a megoldás akkor működik, ha más feladatok is várnak a nyomtatási sorban. Nagyon gyorsnak kell viszont lennünk, ha egyedül mi használjuk a nyomtatót, mivel a nyomtatási feladatok másodpercek alatt a nyomtatón lehetnek. Ha a nyomtatási feladatok egyszer elértek a nyomtatóhoz, Linux eszközökkel már nem lehet eltávolítani azokat.

8.4.2. Ha nyomtatványunk nem készült el

Az lpq parancs használatával megnézhetjük nyomtatási feladatunkat:

```
csabi:~> lpq
Printer: lp@tapolca
Queue: 2 printable jobs
Server: pid 19873 active
Unspooler: pid 19874 active
Status: waiting for subserver to exit at 14:52:20.699
Rank    Owner/ID              Class  Job  Files       Size  Time
1       csabi@tapolca+997     A      997  (STDIN)      129  14:51:54
2       csabi@tapolca+22      A       22  /etc/profile 917  14:52:20
```

Napjainkban számos nyomtatónak van webes interfésze is, mely nyomtatónkról állapot-információkkal szolgál. Ennek lekérése nyomtatónk IP-címének webböngészőben történő megadását jelenti:

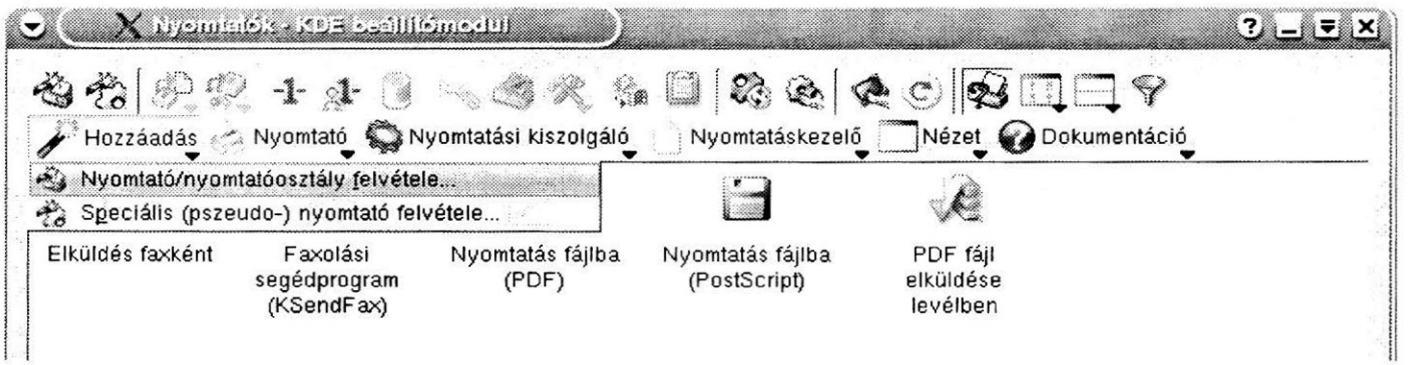
Ha nyomtatási feladatunkat nem találjuk és az nem a nyomtatón van, lépjünk kapcsolatba a rendszergazdával. Ha feladatunk a kimenetben szerepel, ellenőrizzük, hogy a nyomtató jelenleg dolgozik-e. Ha igen, csak várnunk kell nyomtatási feladataink elvégzésére.

Ha a nyomtató nem nyomtat, ellenőrizzük, hogy nem fogyott-e ki a papír, a fizikai érintkezés tökéletes-e és a hálózat is rendben van. Ha minden jónak tűnik, esetleg újra kell indítani a nyomtatót.

Hálózati nyomtatóknál próbáljunk egy másik gépről nyomtatni. Ha a nyomtató saját gépünkről elérhető, poróbáljuk meg rákűldeni a formázott fájlt (mint a fajl.ps a PostScript nyomtató esetében) FTP-klienssel. Ha ez működik, nyomtatási rendszerünk konfigurálása helytelen. Ha nem működik, elképzelhető, hogy a nyomtató nem tudja értelmezni az általunk megadott formátumot.

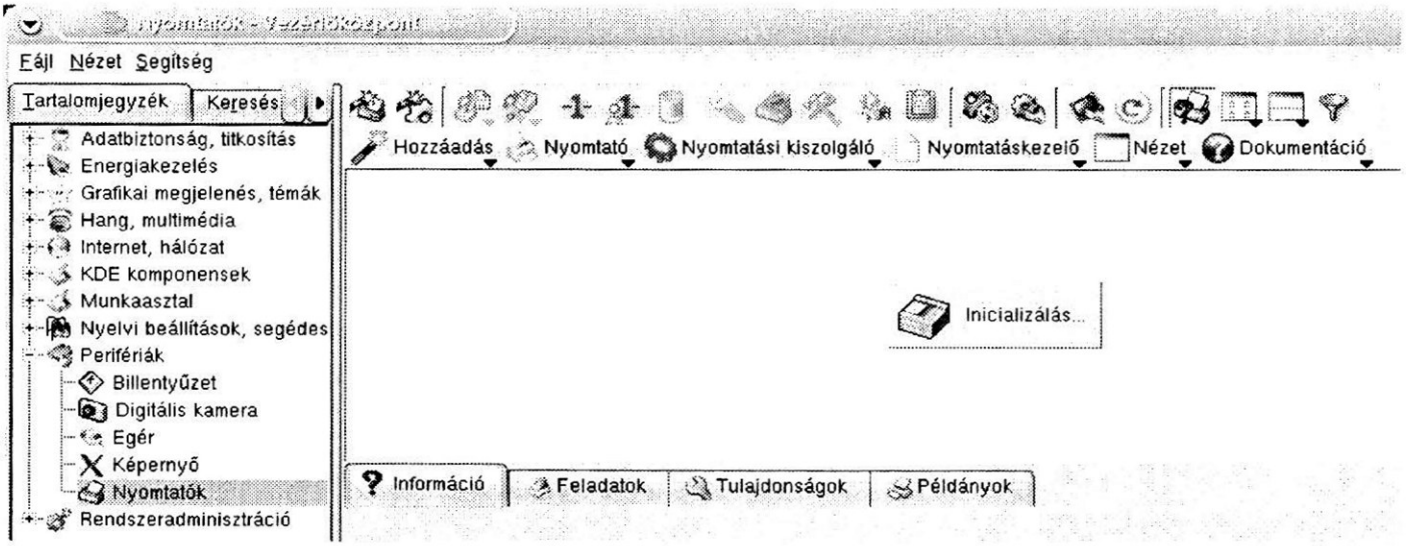
8.5. Nyomtatás grafikus felületről

A nyomtatókkal kapcsolatos műveleteket (új nyomtató hozzáadása, az alapértelmezés szerinti nyomtató megváltoztatása más telepített nyomtatóra stb.) a Nyomtatók KDE beállítómodul segítségével végezhetjük el.



Az itt található Nyomtatáskezelő segítségével szüneteltethetjük, újrakezdhetjük vagy törölhetjük nyomtatási feladatainkat.

A nyomtatókkal kapcsolatos beállításokat a K menü **Rendszer** részéből elérhető **Beállítások** menüben a **Nyomtatók** részben is elvégezhetjük.



9. ARCHÍVÁLÁS

Ha rendszeresen használunk egy számítógépet, előbb-utóbb adatveszteségek léphetnének fel, ha nem készítenénk adatainkról biztonsági másolatot. A következőkben azt nézzük meg, hogyan lehet adatainkat biztonságos helyen, floppy lemezeken, CD-ken vagy streamer-eken tárolni. Megismerkedünk a fájlarchívumok készítésével, lekérdezésével, kicsomagolásával. Megnézzük, hogyan lehet a floppy lemezeket kezelni és hogyan készíthetünk rendszerindító lemezet. A sorból természetesen a CD-írás sem maradhat ki.

Bár a Linux az egyik legbiztonságosabb operációs rendszer, melyet folyamatos futásra terveztek, adataink néha elveszhetnek. Ezt okozhatja felhasználói hiba vagy rendszerhiba (pl. áramkimaradás), így érdemes legalább a fontos adatainkról biztonsági másolatot készítenünk.

9.1. Adatok előkészítése

9.1.1. Archíválás tar-ral

A legtöbb esetben először összegyűjtjük az összes archíválandó fájlt egyetlen archívum-fájlba, melyet később le is tömörítünk. Az archíválás folyamata az összes szükséges fájl összefűzését és a fölösleges részek kihagyását is magában hordozza. A Linux alatt ezt a folyamatot többnyire a tar paranccsal végezzük el. Bár a tar-t eredetileg szalagra történő mentésekhez tervezték, más adathordozókra is menthetünk vele. A tar parancs számos opcióval rendelkezik. A legfontosabbak az alábbiak:

- -c: archívum létrehozása
- -f archiveszkoz: az archiveszkoz eszközt használja az archíválás forrásául vagy céljául. Az eszköz alapértelmezése az első szalagos meghajtó (általában /dev/st0 vagy ehhez hasonló)
- -j: szűrő a bzip2-höz
- -t: tesztelés, egy archívum tartalmát jeleníti meg
- -v: leírást kapunk a folyamatról
- -x: archívum kibontása

A tar opcióinál a kötőjel karaktereket általában el szokták hagyni.

Ha egy rendszer saját tar verzióját használjuk archíváláshoz, előfordulhat, hogy inkompatibilitási problémáink lesznek egy másik rendszeren. Ez különösen akkor nagy probléma, ha egy olyan rendszert kell visszaállítani, mely már nem létezik. A fentiek elkerülésére használjuk a GNU tar-t az összes UNIX rendszeren (a Linux mindig a GNU tar-t használja). Ha más UNIX gépen dolgozunk, a tar --help segítségével ellenőrizzük, melyik verziót használjuk.
Készítsünk egy archívumot, majd bontsuk ki:

```
judit:~> ls kepek/
kutya.jpg macska.jpg

judit:~> tar cvf kepek.tar kepek/
kepek/
kepek/macska.jpg
kepek/kutya.jpg

judit:~> cd kepek

judit:~/images> tar cvf fotok.tar *.jpg
kutya.jpg
macska.jpg

judit:~/kepek> cd
judit:~> ls */*.tar
kepek/fotok.tar

judit:~> ls *.tar
kepek.tar

judit:~> tar xvf kepek.tar
kepek/
kepek/macska.jpg
kepek/kutya.jpg

judit:~> tar tvf kepek/fotok.tar
-rw-r--r-- judit/judit 43645 2004-07-25 08:38:15 kutya.jpg
-rw-r--r-- judit/judit 35464 2005-02-21 12:12:24 macska.jpg

judit:~> tar xvf kepek/fotok.tar
kutya.jpg
macska.jpg

judit:~> ls *.jpg
kutya.jpg macska.jpg
```

A példában jól megfigyelhető a különbség egy tar-ral archivált könyvtár és egy tar-ral archivált fájlcsoport között. Ajánlatos könyvtárakat archíválni, így nem kell foglalkozni az egyes fájlokkal és kibontáskor az eredeti könyvtárat kapjuk vissza.

Ha rendszerünk fel van szerelve szalagos egységgel és a rendszergazda azt megfelelően konfigurálta is, a .tar végződésű fájlnevek lecserélődnek a szalagos egység nevével:

```
tar cvf /dev/tape mentes/
```

A mentes könyvtár és annak összes fájlja egyetlen fájlba kerül, mely közvetlenül kiíródik a szalagra. Mivel használtuk a -v opciót, a folyamatról információkat is kapunk (látjuk a tartalmat).

9.1.2. Bővíthető archívumok tar-ral

A tar támogatja növekvő archívumok létrehozását is az -N opcióval. Ekkor megadhatunk egy dátumot, a tar pedig ellenőrzi az összes megadott fájl módosítási dátumát és összeveti a megadottal. Ha olyan fájlokat talál, melyek később lettek módosítva, mint a megadott dátum, a fájlok bekerülnek az archívumba.

A következő példában egy korábbi archívum létrehozási idejét használjuk dátum-értékként. Először a kezdeti archívumot hozzuk létre, majd megtekintjük a kezdeti mentés létrehozási idejét. Végül létrehozunk egy új fájlt és az új biztonsági mentés csak ezt az új fájlt tartalmazza:

```
mari:~> tar cvpf /var/tmp/progik.tar java/*.java
java/bejon.java
java/bejovo.java
java/eszkoz.java
java/fo.java
java/hiba.java
java/honlap.java
java/input.java
java/jatek.java
java/modosit01.java
java/piramis.java
java/slw.java
java/test3.java
java/teszt1.java
java/teszt2.java
java/udv.java

mari:~> ls -l /var/tmp/progik.tar
-rw-rw-r-- 1 mari mari    36572 Feb 15 19:10 /var/tmp/progik.tar

mari:~> touch java/ujprog.java

mari:~> tar -N /var/tmp/progik.tar \
-cvp /var/tmp/bovit-progik.tar java/*.java 2> /dev/null
java/ujprog.java
```

```
mari:~> cd /var/tmp/

mari:~> tar xvf bovit-progik.tar
java/ujprog.java
```

Az alapvető hibákat a /dev/null-ba irányítjuk. Ha nem így járunk el, a tar minden egyes változatlan fájlról üzenetet ad.

Ennek a mentési módnak az a hátránya, hogy a fájlok idő-adataira épít. Tételezzük fel, hogy letöltünk egy archívumot abba a könyvtárba, ahol a biztonsági mentéseink vannak. Az archívumban levő adatokat 5 éve hoztuk létre. Ha ezen fájlok idő-adatait ellenőrzi a tar a kezdeti archívumhoz képest, az új fájlok réginek tűnnek számára és az -N opcióval készített archívumba ezek a fájlok nem kerülnek bele.

Egy másik lehetőség a -g opció használata, mely egy listát készít az archíválandó fájlokról. Ha bővíthető mentéseket készítünk, a fájlok ezen lista alapján kerülnek ellenőrzésre:

```
anita:~> tar cvpf melo-20040825.tar -g aktual-20040825 munka/
munka/
munka/feladat1
munka/feladat2
munka/feladat3

anita:~> file aktual-20040825
aktual-20040825: ASCII text
```

A következő napon anita dolgozott a feladat3 fájlon és létrehozta feladat4-et. A nap végén új biztonsági mentést végez:

```
anita:~> tar cvpf melo-20040825.tar -g aktual-20040825 munka/
munka/
munka/feladat3
munka/feladat4
```

A biztonsági mentés parancsait természetesen ütemezhetjük is heti vagy napi rendszerességgel, igényeinknek megfelelően. Ha egy-egy aktuális állapotról is készítünk mentést, teljes biztonsági mentések készítésekor ezeket természetesen törölhetjük.

9.1.3. Tömörítés, kicsomagolás

A különféle adatok, beleértve a tar-ral készített archívumokat is, tömöríthetők. A gzip parancs a fájlnév után a .gz végződést teszi, az eredeti fájlt pedig eltünteti.

```
magdi:~> ls -la | grep tar
-rw-rw-r-- 1 magdi magdi 71655 Jul 5 14:34 kepek.tar
```

```
magdi:~> gzip images-without-dir.tar

magdi:~> ls -la images-without-dir.tar.gz
-rw-rw-r-- 1 magdi magdi 60777 Jul 5 14:34 kepek.tar.gz
```

A gzip használatával tömörített fájlokat a -d opcióval csomagolhatjuk ki.

A bzip2 teljesen hasonlóan működik, de egy fejlettebb tömörítési algoritmust használ, így kisebb fájlokat készít.

A különféle Linux szoftvercsomagok sokszor gzip-pel vannak tömörítve. Érdemes az ilyen csomagok kitömörítése után megkeresni és elolvasni a README (olvassel) fájlt, ami a csomag telepítésének általános leírását tartalmazza.

A GNU tar program képes a gzip fájlok kezelésére. A

```
tar zxvf fajl.tar.gz
```

parancs használatával kitömöríthetjük és kibonthatjuk a .tar.gz (vagy rövidített formában .tgz) fájlokat.

Ha olyan tar archívumokat szeretnénk kicsomagolni, melyek bzip2-vel lettek tömörítve, használjuk az alábbi parancsot:

```
tar jxvf fajl.tar.bz2
```

9.1.4. Java archívumok

A GNU projekt a jar eszközt kínálja Java archívumok létrehozására. A jar egy olyan Java alkalmazás, mely több fájlt egyetlen JAR archívumba egyesít. Mivel a jar a ZIP és a ZLIB formátumokon alapuló általános célú archíváló és tömörítő program, kifejezetten alkalmas Java kódok, applet-ek és alkalmazások egyetlen fájlba archíválására. Ha egy Java alkalmazás komponenseit egyetlen archívumba egyesítjük, azok sokkal gyorsabban tölthetők le. A tar-ral ellentétben a jar alapból tömörít, függetlenül más programoktól (alapjában véve a zip Java verziójáról van szó). A fájl szerzője megjelölheti a fájlt, így annak eredete könnyen hitelesíthető.

A jar szintaktikája hasonlít a tar-hoz, a különbségeket az info jar kiadásával ismerhetjük meg legkönnyebben.

Érdekes megjegyzés a szimbolikus hivatkozások archíválása. Ha a jar szimbolikus hivatkozásba ütközik, azt az adatot is menti, melyre a hivatkozás mutat. A tar alapesetben ilyenkor csak a hivatkozást menti el, de ez a -h opcióval felülbírálható.

9.2. Adatok kiírása

9.2.1. Mentés floppy-ra

9.2.1.1. Floppy-formázás

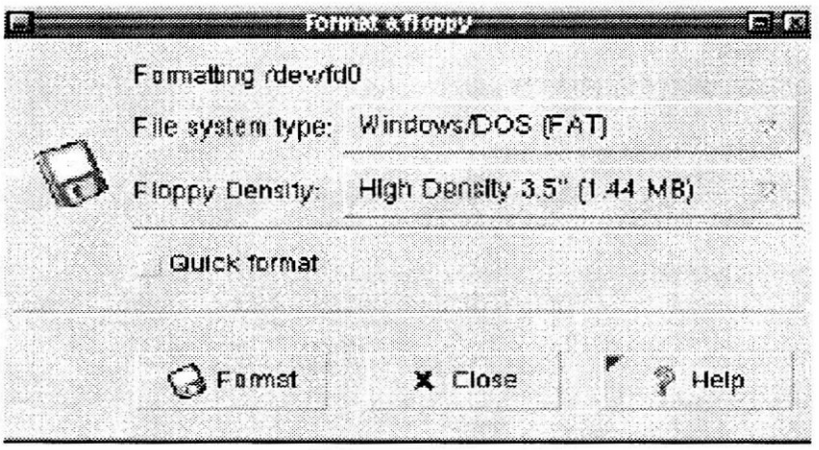
A legtöbb Linux rendszeren a felhasználók elérik a floppy-meghajtót. Az eszköz neve a floppy-meghajtók méretétől és számától függően eltérő. Gyakran a /dev/floppy hivatkozás mutat a megfelelő meghajtóra (ez /dev/fd0 automatikusan detektált meghajtók esetén vagy /dev/fd0H1440 1,44 MB-os meghajtóknál).

Alacsony szintű formázást az fdformat paranccsal végezhetünk. Opciója a floppy-lemez eszközneve. Ha a floppy írásvédő csúszkája el van húzva (a lemez írásvédett), az fdformat hibát ad.

```
moni:~> fdformat /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... done

moni:~>
```

Az mtools csomag mformat parancsa DOS-kompatibilis lemezeket tud létrehozni, mely aztán az mcopy, mdir és egyéb m-parancsokkal elérhető (az m-parancsokkal DOS-os szintaktikájú parancsokat adhatunk ki).
Természetesen grafikus programot is választhatunk.



Miután a lemezt megformáztuk, becsatolhatjuk a fájlrendszerbe. Ettől kezdve egy szokásos (kis méretű) könyvtárként érhetjük el tartalmát, általában az /mnt/floppy-n keresztül.

Szinte az összes Linux disztribúció tartalmazza az mkbootdisk programot, mely egy rendszerindító lemezt készít (az aktuális rendszerhez).

9.2.1.2. A dd parancs

A dd a GNU coreutils csomag része. Segítségével adatainkat a floppy-ra írhatjuk van onnan eltávolíthatjuk (a megadott be- és kimeneti eszköztől függően):

```
ibi:~> dd if=kepek.tar.gz of=/dev/fd0H1440
98+1 records in
98+1 records out

ibi:~> dd if=/dev/fd0H1440 of=/var/tmp/fotok.tar.gz
2880+0 records in
2880+0 records out

ibi:~> ls /var/tmp/fotok*
/var/tmp/fotok.tar.gz
```

Megjegyzendő, hogy a műveletek egy be nem csatolt eszközön lettek végrehajtva. Az így létrehozott floppy-k nem csatolhatók be a fájlrendszerbe!
A dd parancs egy teljes merevlemez nyers mentéséhez is használható.

9.2.2. Mentés CD-re

Egyes rendszereken a felhasználók is használhatják a CD-írót (valahol csak a rendszergazda). A CD-re történő mentéshez először rendezni kell adatainkat egy könyvtárban. Ehhez használhatjuk az mkisofs parancsot. A df segítségével ellenőrizhető, van-e elegendő szabad hely (az új fájl majdnem ugyanakkora, mint a teljes szóban forgó könyvtár):

```
[alex@veszprem mentendo] df -h .
Filesystem                    Size  Used  Avail  Use%  Mounted on
/dev/hde4                     19G   15G   3.2G    82%  /home

[alex@veszprem mentendo] du -h -s .
325M    .

[alex@veszprem mentendo] mkisofs -J -r -o cd.iso .
<--snap-->
making a lot of conversions
<--/snap-->
98.95% done, estimate finish Wed Apr 6 10:35:18 2005
Total translation table size: 0
Total rockridge attributes bytes: 35971
Total directory bytes: 94208
Path table size(bytes): 452
Max brk space used 37e84
166768 extents written (325 Mb)
```

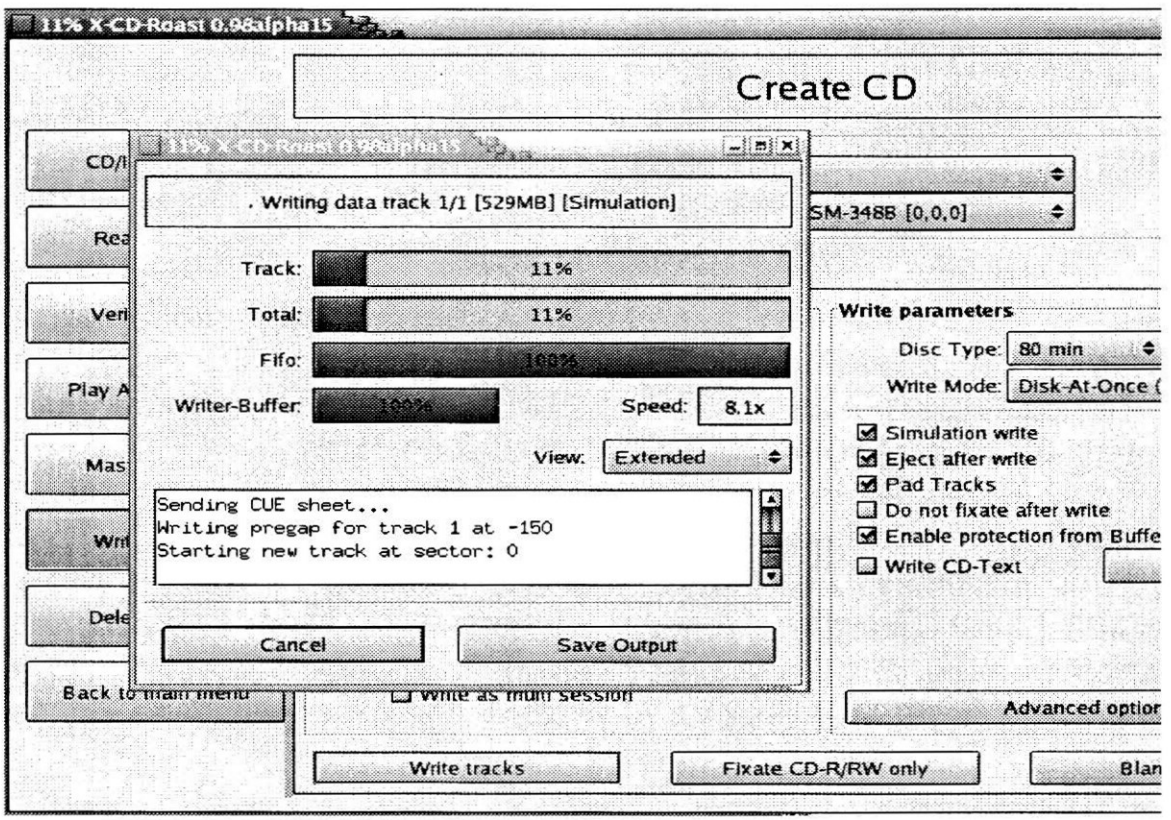
A J és az r opciókkal a CD-ROM különböző rendszerek alá is becsatolható. Ezután a CD a cdrecord és a megfelelő opciók használatával hozható létre:

```
[timi@diszel mentendo] cdrecord -dev 0,0,0 -speed=8 cd.iso
Cdrecord 1.10 (i686-pc-linux-gnu) (C) 1995-2001 Joerg Schilling
scsidev: '0,0,0'
scsibus: 0 target: 0 lun: 0
Linux sg driver version: 3.1.20
Using libscg version 'schily-0.5'
Device type    : Removable CD-ROM
Version        : 0
Response Format: 1
Vendor_info    : 'HP '
Identification : 'CD-Writer+ 8100 '
Revision       : '1.0g'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags   : SWABAUDIO
Starting to write CD/DVD at speed 4 in write mode for single session.
Last chance to quit, starting real write in 0 seconds.
Operation starts.
```

A folyamat végeztével egy üzenetet kapunk:

```
Track 01: Total bytes read/written: 341540864/341540864
          (166768 sectors).
```

Linux alatt természetesen grafikus eszközökkel is írhatunk CD-ket. Az egyik legnépszerűbb az xcdroast (a legtöbb rendszerben a GNU könyvtárban megtalálható).



A különféle ablakozók (Gnome, KDE) saját eszközeikkel segítik egyéni CD-ink elkészítését.

9.2.3. Mentés USB-eszközökre

Az USB-eszközöket (pl. Flash drive) általában be kell csatolni a fájlrendszerbe. A becsatolás után tartalmukat közönséges könyvtárakként érhetjük el, azaz a közönséges parancsokkal végezhetünk fájlműveleteket.

Másoljuk át USB-s fényképezőgépünk képeit a merevlemezre*:

```
olga:~> mount /mnt/camera

olga:~> mount | grep camera
/dev/sda1 on /mnt/camera type vfat (rw,nosuid,nodev)

olga:~> cp -R /mnt/camera/* kepek/

olga:~> umount /mnt/camera
```

A fenti műveletek elvégzéséhez bővíthetjük az /etc/modules.conf és az /etc/fstab fájlokat is.

9.2.4. Mentés szalagra

A szalagos egységre történő mentéshez használható a már megismert tar. Az mt program segítségével kontrollálhatjuk a mágnesszalagos egységet, például a /dev/st0 eszközt. Ha adatbázisokat mentünk, tartsuk szem előtt, hogy egyéb mentési módszerekre is szükség lehet az adatszerkezet miatt.

A megfelelő mentési parancsok általában valamelyik cron könyvtárban vannak, mivel gyakran futtatjuk őket. A nagyobb környezetekben a szabadon elérhető Amanda mentési csomag vagy valamely kereskedelmi termék segítségével biztosíthatjuk az összetett adatok mentését.

9.2.5. Programok archíváláshoz

A legtöbb disztribúció saját archíváló programokkal rendelkezik. A leggyakrabban az alábbiakkal találkozunk:

- File Roller: archívumok és tömörített archívumok vizuális megjelenítésére alkalmas program RedHat disztribúciókon
- X-CD-Roast: a Mandrake Linux archíválója
- YaST: fejlett archíválási rendszerrel is rendelkező eszköz SuSE Linux-on

A legtöbb disztribúció tartalmazza a BSD-s dump és restore programokat ext2 és ext3 fájlrendszerek biztonsági mentéseihez. Több eszközre írhatjuk az adatokat bitről bitre. A dd-hez hasonlóan ilyen módon különleges típusú fájlokat is menthetünk (pl. amilyenek a /dev könyvtárban vannak).

---

* A témával foglalkozik a Szerző Digitális fényképezés a mindennapokban c. könyve is

10. HÁLÓZATMENEDZSMENT

Ha komolyan akarunk hálózatokkal dolgozni, a Linux egy kiváló választás, mivel a rendszer hálózati operációs rendszernek készült. Számos ingyenes hálózatkezelő alkalmazás érhető el hozzá. Robosztusságát és stabilitását pedig a fejlődését biztosító nyílt forráskódú programokkal foglalkozó projekt több éves tesztelései garantálják.

10.1. Hálózati protokollok

Egy hálózati protokoll tulajdonképpen a kommunikáció szabályainak halmaza. A Linux rendszerek számos különféle hálózati protokollt támogatnak. A következőkben a legfontosabb protokollokkal ismerkedünk meg.

10.1.1. TCP/IP

A TCP/IP a Transport Control Protocol és az Internet Protocol rövidítése. Ez a két protokoll a legfontosabb kommunikációs mód az interneten. Számos alkalmazás (böngészők, levelezőprogramok) ezekre a protokollokra épít.

Az IP nagyon leegyszerűsítve egy megoldást kínál információk egyik gépről a másik gépre történő továbbítására. A TCP biztosítja, hogy a csomagok adatfolyamokba legyenek rendezve, így a különféle alkalmazások csomagjai nem keveredhetnek össze, továbbá a csomagok megfelelő sorban kerülnek elküldésre és fogadásra.

Az internet néhány évtizedes múltra tekint vissza. Eredetileg különböző számítógépek összekötésére tervezték. A TCP/IP kifejlesztésének másik oka egy nem túl megbízható hálózaton megbízható adatátvitelt megvalósító rendszer létrehozásának igénye volt.

A Linux már a kezdet kezdetén támogatta a TCP/IP protokollt. Az operációs rendszer sikerének egyik alappillére a robosztusságát, gyorsaságát, megbízhatóságát hálózaton garantáló TCP/IP protokoll volt. A Linux és a hálózat olyannyira eggyé vált, hogy ha önállóan futtatunk egy Linux rendszert, az lassabban indulhat és más problémáink is lehetnek. Ha nem is csatlakozunk hálózatba kötött számítógépekhez, a hálózati protokollokat a belső rendszer és

az alkalmazások akkor is használják. A fentieken nem kell meglepődni, tudván, hogy a Linux egy igazi hálózati operációs rendszer.

10.1.2. TCP/IPv6

Senki sem számított rá, hogy az internet ilyen hatalmasra nő. Az IP protokollnak hátrányos tulajdonságai jelentkeztek igen nagyszámú számítógépek összekötése esetén. A legnagyobb problémát a megosztott gépek egyedi címeinek elérése jelentette. Ezért fejlesztették ki az IP 6-os verzióját, mely napjaink internetes igényeit is kielégíti.

A problémát jelenleg az jelenti, hogy még nem minden alkalmazás támogatja az IPv6-ot. A régi alkalmazások egy része használja az eredeti protokollt is. Ha tehát megváltoztatjuk a hálózat konfigurációját, a fentiekből adódóan problémák jelentkezhetnek.

10.1.3. PPP, SLIP, PLIP, PPPOE

A Linux kernel beépített támogatást tartalmaz a PPP-hez (Point-to-Point-Protocol), a SLIP-hez (Serial Line IP) és a PLIP-hez (Parallel Line IP).

A PPP az egyéni felhasználók körében a saját ISP (Internet Service Provider) elérésének legnépszerűbb módja.

A sűrűn lakott területeken a PPP helyett gyakran alkalmazzák a PPPOE-t (PPP Over Ethernet), a kábelmodemes kapcsolatokhoz használatos protokollt.

A legtöbb Linux-disztribúció egyszerűen használható eszközöket kínál az internet-kapcsolat beállítására. Amire feltétlenül szükség van, az egy felhasználói név és egy jelszó az Internet Service Provider-hez való csatlakozáshoz (PPP esetén egy telefonszámra is). Ezeket az adatokat egy grafikus konfiguráló felületen adhatjuk meg, ahol lehetőség van a szolgáltatónkhoz való csatlakozásra és a kapcsolat bontására is.

10.1.4. ISDN

A Linux kernel tartalmazza az ISDN-támogatást is. Az isdn4linux vezérli az ISDN-kártyákat és képes Hayes-szabványú modem ("AT" parancsok) emulálására is. A lehetőségek széles tárháza egy egyszerű terminálprogram használatától az internetre történő teljes kapcsolódásig mindent magába foglal.

10.1.5. AppleTalk

Az AppleTalk az Apple hálózati protokollja. Pont-pont hálózati modellt támogat, ami olyan alapvető funkcionalitásokat biztosít, mint a fájl- vagy a nyomtatómegosztás. Minden gép egyidejűleg működhet kliensként és szerverként is. A szükséges hardver- és szoftvereszközökkel az összes Apple számítógépet felszerelték.

A Linux netatalk az AppleTalk protokoll-gyűjtemény kernel-szintű implementációja (eredetileg BSD-rendszerekhez készült). Támogatja az AppleTalk útvonal-választást, az AppleShare használatával pedig kiszolgálja a UNIX és AFS fájlrendszereket. Kínálja a UNIX nyomtatók használatát és az AppleTalk nyomtatók elérését is megvalósítja.

10.1.6. SMB/NMB

A Microsoft Windows operációs rendszerekkel való kompatibilitás megteremtése érdekében az NMB és SMB protokollokat támogató Samba csomag minden UNIX-alapú rendszerre telepíthető. Az SMB (Server Message Block vagy Session Message Block, más néven NetBIOS vagy LanManager protokoll) fájl- és nyomtatómegosztáshoz használható Microsoft Windows 3.11, NT, 95/98, 2000 és XP alatt.

A Samba alapvető funkciója a Linux meghajtók megosztása Windows-os gépekkel, SMB megosztások elérése Linux gépekről, Linux-os nyomtatók megosztása Windows-os gépekkel, illetve Windows-os nyomtatók megosztása Linux-os gépekkel.

A legtöbb Linux disztribúció tartalmazza a samba csomagot, ami elvégzi a szerver-beállítás nagy részét és alapból elindítja az smbd-t (vagyis a Samba szervert) és az nmbd-t (a netbios name server-t) a rendszerbetöltés idején. A Samba grafikusan konfigurálható webes interfészen vagy a parancssoron keresztül (szöveges konfigurációs fájlokkal). A démonok segítségével egy Linux gép Microsoft Windows alatt egy Windows-os hosztként jelenik meg. Windows-környezetben egy Linux gép megosztása más megosztásoktól megkülönböztethetetlen.

10.1.7. Különleges protokollok

A Linux operációs rendszer a korábbiakban tárgyaltakon kívül speciális protokollokat is támogat. Ilyen például az Amateur Radio, az ATM, a Frame Relay, az X25 vagy az infravörös és egyéb vezeték nélküli kapcsolatok protokolljai. Ezek a protokollok általában speciális hardvert is igényelnek, így részletesebben nem foglalkozunk velük.

10.2. Hálózat-konfigurálás

10.2.1. Helyi hálózati interfészek konfigurációja

Az összes nagyméretű, felhasználóbarát Linux-disztribúció tartalmaz különféle grafikus eszközöket, melyek megkönnyítik a számítógép helyi hálózatos telepítését, illetve egy internet-szolgáltatóhoz való kapcsolódását. Ezek a programok a parancssorból vagy a menüből indíthatók:

- a Mandrake tartalmaz egy hálózat- és internet-konfigurációs varázslót, melyet a legjobb a Mandrake vezérlő-központjából indítani
- a RedHat Linux sajátja a redhat-config-network, melynek mind szöveges, mind grafikus interfésze létezik
- a SuSE YaST, illetve YaST2 minden az egyben típusú konfigurációs eszköze hálózatok konfigurálására is alkalmas

A rendszerdokumentációnkat érdemes megnézni saját hálózat-konfigurációs eszközeinkről.
Az alábbi információkra van szükség:

- helyi hálózathoz történő csatlakozáshoz: hosztnév, domain név és IP cím.
- az internetre történő csatlakozáshoz: felhasználónév és jelszó az ISP-nkhez, modem használatánál egy telefonszám. Az ISP-nk általában automatikusan adja nekünk a hosztnevet, az IP címet és az összes olyan adatot, ami az internetes alkalmazások működéséhez szükséges.

10.2.2. Hálózat-konfigurációs fájlok

A grafikus segédeszközök a hálózati konfigurációs fájlok egy meghatározott halmazát szerkesztik néhány alapvető parancs használatával. Ezen konfigurációs fájlok elnevezése és helye a fájlrendszerben nagymértékben függ a használt Linux-disztribúciótól és verziótól. A következőkben a UNIX-alapú rendszerek közös hálózati konfigurációs fájljaira vetünk egy pillantást:

- /etc/hosts: mindig a localhost IP címet tartalmazza (127.0.0.1), ami a folyamatok közötti komminukáció során használatos. Soha ne távolítsuk el ezt a sort! A fájl néha olyan egyéb hosztok címét is tartalmazza, melyek esetében a kapcsolathoz nem szükséges egy külső név-szolgáltatás (mint például a DNS).
  Egy kis otthoni hálózat hosts fájlja körülbelül a következőképpen néz ki:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       localhost.localdomain   localhost
192.168.51.10   diszel.siki-soft.hu     diszel
192.168.51.11   winxp.siki-soft.hu      winxp
```

- /etc/resolv.conf: egy DNS-szerverhez való hozzáférést konfigurál. Ez a fájl a domain nevünket és azon szerver(ek) nevét (neveit), mely(ek)hez csatlakozni kívánunk:

```
search siki-soft.hu
nameserver 193.134.21.4
```

- /etc/nsswitch.conf: a különféle névszerverekhez való csatlakozás sorrendjét definiálja. Az internetes használathoz fontos, hogy a dns szerepeljen a hosts sorban:

  ```
  [laci@diszel ~] grep hosts /etc/nsswitch.conf
  hosts: fajlok dns
  ```

  Ezzel arra utasítjuk a számítógépet, hogy a hosztneveket és IP-címeket először az /etc/hosts fájlban keresse és kapcsolódjon a DNS-szerverhez, ha egy adott hoszt nem fordul elő a helyi hosts fájlban. A DNS-en kívül más névszolgáltatások is előfordulhatnak, például az LDAP, a NIS vagy a NIS+.

10.2.3. Hálózat-konfigurációs parancsok

A disztribúció-függő scriptek és grafikus eszközök az ip eszközt (a régebbi rendszereken az ifconfig-ot és a route-ot) használják a kernel hálózati konfigurációjának beállítására.

Az ip parancs IP-címek interfészhez történő megadására, útvonalak beállítására az internethez és más hálózatokhoz, TCP/IP konfigurációk megjelenítésére és más hálózat-konfigurációs feladatokra alkalmas.
A következő parancs mutat IP-cím- és útvonal-információkat:

```
tomi@pecs tomi> ip addr show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 127.255.255.255 scope host lo
    inet6 ::1/128 scope host
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:50:bf:7e:54:9a brd ff:ff:ff:ff:ff:ff
    inet 192.168.42.15/24 brd 192.168.42.255 scope global eth0
    inet6 fe80::250:bfff:fe7e:549a/10 scope link

tomi@pecs tomi> ip route show
192.168.42.0/24 dev eth0 scope link
127.0.0.0/8 dev lo scope link
default via 192.168.42.1 dev eth0
```

Ha olyan laptop-unk van, melyet legtöbbször a vállalati hálózathoz csatlakoztatunk Ethernet kapcsolattal, de egyszer otthon vagy egy szállodában kell konfigurálnunk, előfordulhat, hogy a PCMCIA kártyát aktiválnunk kell. Erre alkalmas a cardctl vezérlőprogram. Egy valamirevaló disztribúció egyébként támogatja a PCMCIA-t a hálózati konfigurációs eszközeivel így a felhasználóknak legtöbbször nem kell manuálisan futtatniuk a PCMCIA parancsokat.

10.2.4. Hálózati interfész-nevek

Egy Linux gépen az lo vagy local loop eszköznév a belső 127.0.0.1 címre hivatkozik. Számítógépünk nagyon nehezen dolgozik alkalmazásainkkal, ha ez az eszköz nem létezik. Erre az eszközre mindig szükség van, még akkor is, ha számítógépünk nincs hálózatba kapcsolva.

Az első ethernet-eszköz, az eth0 egy standard hálózati interfész-kártya esetén a helyi LAN IP címére mutat. Egy egyszerű kliens gépnek egyetlen hálózati interfész-kártyája van. A hálózatokat összekötő router-eknek minden általuk kiszolgált hálózathoz van egy hálózati eszközük.

Ha modem vagy kábelmodem használatával kapcsolódunk az internethez, hálózati eszközünk valószínűleg a ppp0 nevet kapja.

10.2.5. Saját hosztunk konfigurációja

A hálózati konfigurációt megjelenítő ip parancs mellett használhatjuk a netstat parancsot is. Számos opciója létezik és általában minden UNIX-alapú rendszeren használható. A hálózati interfészekről az -i opcióval ad információkat:

```
mari:~> netstat -i
Kernel Interface table
Iface   MTU MetRXOK RXERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0   1500 058459      0      0      0  63865      0      0      0 MBRU
lo     6436 024060      0      0      0  24060      0      0      0 LRU
```

Útvonal-információkhoz jutunk, ha a netstat parancsnak használjuk az -nr opcióját:

```
moni:~> netstat -nr
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.42.0    0.0.0.0         255.255.255.0   U        40 0          0 eth0
127.0.0.0       0.0.0.0         255.0.0.0       U        40 0          0 lo
0.0.0.0         192.168.42.1    0.0.0.0         UG       40 0          0 eth0
```

Ez egy tipikus kliens gép egy IP-hálózaton. Itt csak egyetlen hálózati eszköz van, az eth0.

10.2.6. Más hosztok

Mivel a Linux egyik nagy erőssége a hálózatkezelés és a távoli adminisztráció, számos program áll rendelkezésünkre konfigurálási feladataink megkönnyítésére. A következőkben a leggyakrabban előforduló szöveges eszközökkel ismerkedünk meg.

Hosztok vagy domain-ek információinak megjelenítéséhez használhatjuk a host parancsot:

```
[nori@komarom nori]$ host www.siki-soft.hu
www.siki-soft.hu. has address 193.74.208.177

[nori@komarom nori]$ host -t any siki-soft.hu
siki-soft.hu. SOA dns.siki-soft.hu. hostmaster.Hungary.SW.com.
2005050500 28800 7200 604800 86400
siki-soft.hu. mail is handled by 50 pophost. siki-soft.hu.
siki-soft.hu. name server ns.SW.com.
siki-soft.hu. name server dns. siki-soft.hu.
```

Hasonló információkat jelenít meg a dig parancs is, melynek segítségével még azt is megtudhatjuk, hogy a rekordok hogyan vannak tárolva a névszerveren.

Egy hoszt működésének ellenőrzésére szolgál a ping parancs. Ha rendszerünk úgy van konfigurálva, hogy egynél több csomagot küldjön, a ping futása a Ctrl + C billentyűkombinációval szakítható meg:

```
[nori@komarom nori]$ ping egy.hoszt.hu
PING egy.hoszt.hu (1.2.8.3) from 80.20.84.26: 56(84) bytes of data.
64 bytes from egy.hoszt.hu(1.2.8.3):icmp_seq=0 ttl=244 time=99.977msec
--- egy.hoszt.hu ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max/mdev = 99.977/99.977/99.977/0.000 ms
```

Annak az útvonalnak az ellenőrzéséhez, melyen a csomagok egy hálózati hoszthoz haladnak, használható a traceroute parancs:

```
[attila@velence attila]$ /usr/sbin/traceroute www.eunet.be
traceroute to www.eunet.be(193.74.208.177),30 hops max,38b packets
1 blob (10.0.0.1)
0.297ms 0.257ms 0.174ms
2 adsl-65.myprovider.be (217.136.111.1)
12.120ms 13.058ms 13.009ms
3 194.78.255.177 (194.78.255.177)
13.845ms 14.308ms 12.756ms
4 gigabitethernet2-2.intl2.gam.brussels.skynet.be (195.238.2.226)
13.123ms 13.164ms 12.527ms
5 pecbru2.car.belbone.be (194.78.255.118)
16.336ms 13.889ms 13.028ms
6 ser-2-1-110-ias-be-vil-ar01.kpnbelgium.be (194.119.224.9)
14.602ms 15.546ms 15.959ms
7 unknown-195-207-939.eunet.be (195.207.93.49)
16.514ms 17.661ms 18.889ms
8 S0-1-0.Leuven.Belgium.EU.net (195.207.129.1)
22.714ms 19.193ms 18.432ms
9 dukat.Belgium.EU.net (193.74.208.178) 22.758ms * 25.263ms
```

A domain név specifikus információi a whois paranccsal kérhetők le. Nézzük meg egy whois szerver információit!

```
[laci@papa laci]$ whois cnn.com
[whois.crsnic.net]

Whois Server Version 1.3

        $<--snap server message-->

    Domain Name: CNN.COM
    Registrar: NETWORK SOLUTIONS, INC.
    Whois Server: whois.networksolutions.com
    Referral URL: http://www.networksolutions.com
    Name Server: TWDNS-01.NS.AOL.COM
    Name Server: TWDNS-02.NS.AOL.COM
    Name Server: TWDNS-03.NS.AOL.COM
    Name Server: TWDNS-04.NS.AOL.COM
    Updated Date: 12-mar-2002
>>> Last update of whois database: Fri, 5 Apr 2002 05:04:55 EST <<<

The Registry database contains ONLY .COM, .NET, .ORG, .EDU domains
and Registrars.

[whois.networksolutions.com]

        $<--snap server message-->

Registrant:
Turner Broadcasting (CNN-DOM)
    1 CNN Center
    Atlanta, GA 30303
    Domain Name: CNN.COM
    Administrative Contact:
    $<--snap contactinfo-->
    Technical Contact:
    $<--snap contactinfo-->
    Billing Contact:
    $<--snap contactinfo-->
    Record last updated on 12-Mar-2002.
    Record expires on 23-Sep-2009.
    Record created on 22-Sep-1993.
    Database last updated on 4-Apr-2002 20:10:00 EST.
    Domain servers in listed order:
    TWDNS-01.NS.AOL.COM 149.174.213.151
    TWDNS-02.NS.AOL.COM 152.163.239.216
    TWDNS-03.NS.AOL.COM 205.188.146.88
    TWDNS-04.NS.AOL.COM 64.12.147.120
```

A .com-on, a .net-en, az .org-on és az .edu-n kívüli domain neveknél meg kell határozni a whois szervert. Egy német oldalnál például az alábbi szintaktikát követjük:

```
whois domain.de@whois.dns.de
```

10.3. Internet/Intranet alkalmazások

A Linux operációs rendszer nemcsak hálózati szolgáltatások eléréséhez, de nyújtásához is tökéletes platform. A következőkben a gyakori hálózati szervereket és alkalmazásokat tekintjük át.

10.3.1. Szervertípusok

10.3.1.1. Standalone szerver

Szolgáltatás nyújtása felhasználók részére kétféle módon történhet. Egy démon vagy szolgáltatás futhat önálló (standalone) módban vagy egy másik szolgáltatástól is függhet.

A gyakran és/vagy folyamatosan használt hálózati szolgáltatások általában önálló módban futnak. Ez azt jelenti, hogy független program-démonok, melyek mindig futnak. Leginkább a rendszer indítása során indulnak és meghatározott kapcsolódási pontokon vagy portokon várnak a kérésekre. Ha érkezik kérés, az feldolgozásra kerül, majd a következő kérésig egy várakozási időtartam következik. Egy tipikus példa a webszerver: azt szeretnénk, hogy napi 24 órában elérhető legyen, ha pedig túlságosan le van terhelve, több várakozási ciklust iktat be, hogy az egyidejű kéréseket ki tudja szolgálni. Másik példa lehet egy olyan nagy szoftver-archívum (Sourceforge, Tucows stb.), melynek napi több ezer FTP-kérést kell kezelnie.

Egy példa az önálló hálózati szolgáltatásra otthoni számítógépünkön a named, ami egy gyorsító névszerver. Az önálló szolgáltatások saját folyamatokat futtatnak, amit bármikor ellenőrizhetünk a ps használatával:

```
viki:~> ps auxw | grep named
named 908  0.0  1.0  14876 5108 ?  S Apr15 10:49 named -u named
```

A legtöbb otthoni gépünkön fellelhető szolgáltatás (pl. az FTP szolgáltatás) egy démon futtatása nélkül is működik:

```
andris:~> ps auxw | grep ftp
andris  738    690    0  16:17 pts/6          00:00:00 grep ftp

andris:~> ncftp localhost
NcFTP 3.1.3 (Mar 27, 2002) by Mike Gleason (ncftp@ncftp.com).
Connecting to localhost(127.0.0.1)...
```

```
diszel.siki-soft.hu FTP server (Version wu-2.6.2-8) ready.
Logging in...
Guest login ok, access restrictions apply.
Logged in to localhost.
ncftp / >
```

10.3.1.2. inetd, xinetd

Kisebb (iskolai, otthoni) környezetben általában egy viszonylag kis hálózattal dolgozunk. Időnként egyik gépről a másikra viszünk át pár fájlt az FTP vagy a Samba (Linux és Windows közötti kommunikáció esetén) használatával. Az ilyen és ehhez hasonló esetekben a szükséges szolgáltatásokat alkalomadtán kell csak elindítani. Ha mindig futnának a szolgáltatások, pazarolnánk az erőforrásokat. A kisebb telepítéseknél a szükséges démonok egy központi programtól függnek, ami a szolgáltatások összes olyan portját figyeli, melyekért ő a felelős.

Ez a különleges szerver az internet-szolgáltatás démon (Internet services daemon), ami a rendszer inicializálása során indul el. Két gyakori implementációjával találkozhatunk. Az egyik az inetd, a másik az xinetd (utóbbi az extended, vagyis bővített változat). Vagy az egyik, vagy a másik általában fut a Linux rendszerünkön:

```
balu:~> ps -ef | grep inet
root     926     1  0  May11 ?                00:00:00 xinetd-ipv6 -stayalive -reuse \
-pidfile /var/run/xinetd.pid
```

Azon szolgáltatások, melyekért az internet démon felelős, annak konfigurációs fájljában, az /etc/inetd.conf (inetd esetén), illetve az /etc/xinetd.d (xinetd esetén) fájlban vannak felsorolva. Gyakran itt menedzselt szolgáltatások a fájlmegosztási és nyomtatási szolgáltatások, az SSH, az FTP, a Samba konfigurációs démon, időszolgáltatások stb.

Mihelyst egy kérés érkezik, a központi szerver elindítja a szükséges szervert. Ha például a balu nevű felhasználónak FTP-re van szüksége, egy FTP démon fog futni, amíg arra igény van:

```
balu:~> ps auxw | grep ftp
balu     793   0.1 0.2  3960 1076 pts/6    S 16:44   0:00 ncftp localhost
ftp      794   0.7 0.5  5588 2608 ?        SN 16:44   0:00 ftpd:
localhost.localdomain: anonymous/balu@siki-soft.hu: IDLE
```

Természetesen ugyanez történik akkor is, ha távoli hosztokhoz kapcsolódunk. Vagy a démon válaszol közvetlenül vagy egy távol (x)inetd indítja el azt a szolgáltatást, amire szükségünk van. Mikor kilépünk, a szolgáltatást leállítja.

10.3.2. Levelezés

10.3.2.1. Szerverek

Az alapértelmezett levelező szerverprogram a Sendmail, illetve a Mail Transport Agent (UNIX rendszerek esetén). Robosztus, skálázható és ha megfelelően van konfigurálva adott hardverhez, problémamentesen kezel akár több ezer felhasználót is.

Egy másik levelezőszerver a Qmail. Népszerűségét annak köszönheti, hogy sokkal biztonságosabb társánál, a Sendmail-nél. Ennek magyarázata, hogy míg a Sendmail egy monolitikus program, addig a Qmail kisebb interaktív programokból áll.

Ezek a szerverek képesek a levelezési listák kezelésére, szűrésre, víruskeresésre stb. Ezekből a szoftverekből ingyenes és kereskedelmi változatok egyaránt elérhetők. Levelezőlista-program például az EZmlm, a ListServ, a Mailman, a Majordomo. A vírusirtó programokból is rengeteg típus létezik. Az egyes programoknak gyakran elérhető a Linux-os változata is.

10.3.2.2. Távoli levelezőszerverek

A legnépszerűbb protokollok a távoli levél-eléréshez a POP3 és az IMAP4. Mindkettő támogatja az offline működést, új levél távoli elérését és egyik sem tud SMTP szerver nélkül levelet küldeni.

Míg a POP egy egyszerű protokoll, mely egyszerűen implementálható és szinte valamennyi levelezőkliens támogatja, az IMAP használata célszerűbb, mert:

- az oda-vissza küldött levélüzeneteket is tudja tárolni
- eléri a Usenet üzeneteket és egyéb dokumentumokat is
- folyamatos üzenet-állapot flag-ekkel manipulálható
- mind online, mind offline üzemmódban működik
- online működésre tervezték, kisebb sávszélességnél is jól működik
- összetett postafiókok elérésére és menedzselésére is képes
- támogatja a frissítéseket és az osztott postafiókokat

10.3.2.3. Levelezőprogramok

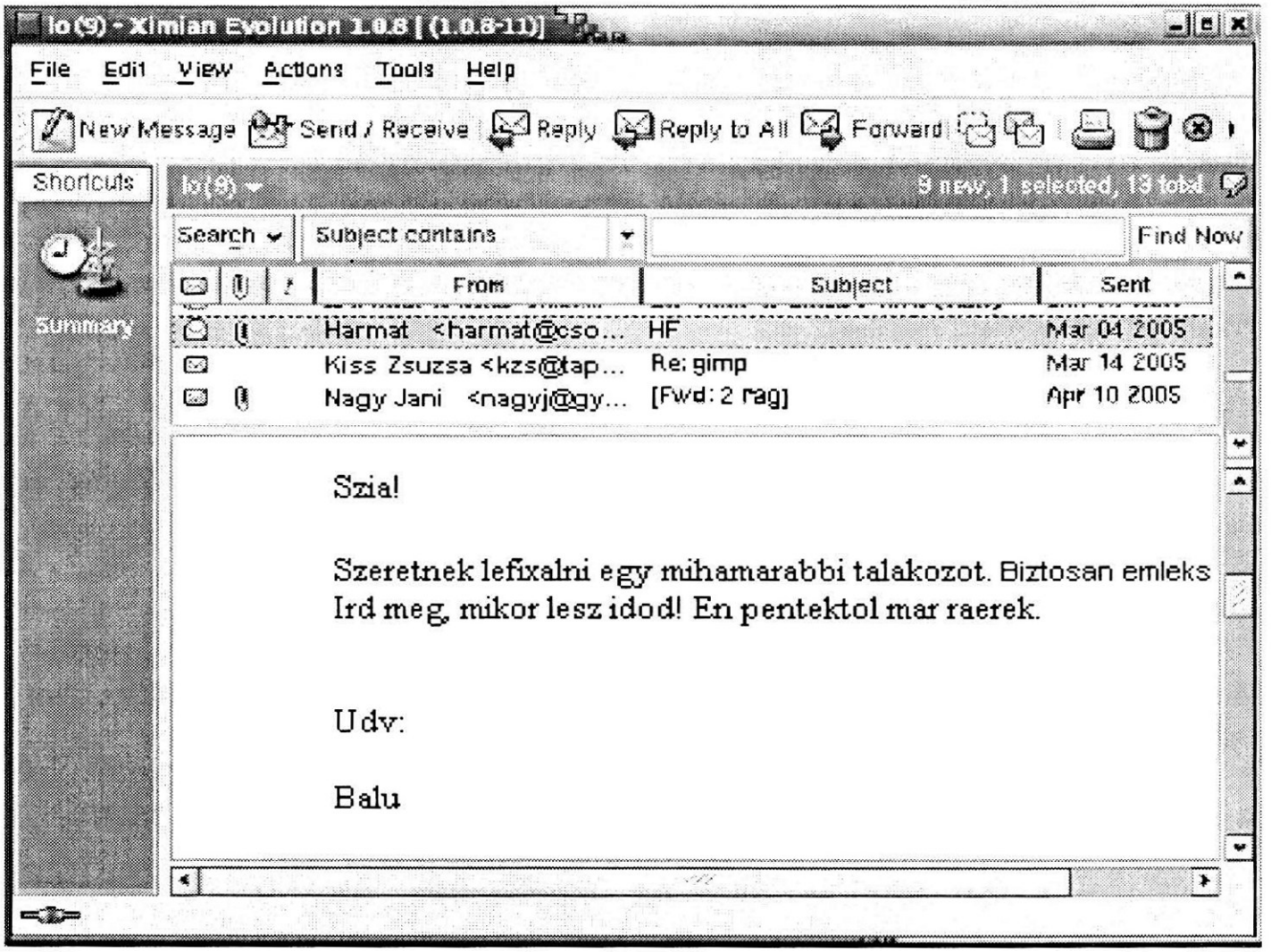
Számos szöveges és grafikus e-mail kliens létezik, melyek közül mindenki tetszés szerint választhat. A következőkben a legnépszerűbb levelezőkből szemezgetünk.

A UNIX-os mail parancs már a hálózatok létezése előtt is elérhető volt. Egy egyszerű interfészt kínál más felhasználók számára történő levélküldéshez. A fogadott levél elmenthető, szerkeszthető, válaszolni lehet rá.

Bár kliensként már nem gyakran használják, a mail program még most is hasznos lehet, ha például egy parancs kimenetét szeretnénk elküldeni valakinek.

Az elm és a pine (Pine Is Not ELM) levélolvasók már a mail fejlettebb változatai. A mutt egy még modernebb program.

A grafikus felületet előnyben részesítők is rengeteg program közül választhatnak. A legnépszerűbbek a Mozilla Mail és a Ximian Microsoft Exchange klónja, az Evolution.



A legtöbb Linux disztribúció tartalmazza a fetchmail levélfogadó és -továbbító programot. Távoli levelezőszerverekről fogadja a leveleket (POP, IMAP) és továbbítja azt helyi rendszerünknek. Ezután a fogadott leveleket már egyszerű levélkliensekkel is kezelhetjük. Démon módban is futhat a megadott időközönkénti ismétlésekhez.

A procmail szűrő alkalmas a bejövő levelek szűrésére, levelezési listák létrehozására, levelek előfeldolgozására, válogatott levéltovábbításra stb. A velejáró formail program egyebek mellett automatikus válasz-generálásra is képes.

10.3.3. Web

10.3.3.1. Az Apache webszerver

Az Apache napjainkban messze a legnépszerűbb webszerver: az internetes webszerverek több, mint fele Apache. A legtöbb Linux disztribúció tartalmazza az Apache-ot. Egyik erőssége a moduláris felépítés, az SSL-támogatottság, a

stabilitás és a sebesség. Ha megfelelő hardver és konfiguráció van biztosítva számára, a legnagyobb terheléseket is állja.

A Linux rendszereken a szerverkonfiguráció általában az /etc/httpd könyvtárban van. A legfontosabb konfigurációs fájl a httpd.conf.

10.3.3.2. Webböngészők

Linux alá számos webböngésző létezik kereskedelmi és ingyenes változatban egyaránt. A Netscape Navigator hosszú ideig az egyetlen választás volt a Linux-felhasználók számára, de a Mozilla megjelenésével a helyzet megváltozott.

A szöveg-alapú böngészők között a lynx és a links a legnépszerűbbek. A megfelelő értékekkel definiálnunk kell a proxy szervereket a shellünkben. Ha nincs lehetőség grafikus felület használatára, a szöveg-alapú böngészőket használhatjuk, mert gyorsak és egyszerűek (a scriptekben például nagyon jól használhatóak).

A webes ajánlásokat készítő W3C Konzorcium saját weblapfejlesztő- és böngésző eszköze, az Amaya számos verziója között megtalálható a Linux-os is.

A kereskedelmi böngészők közül az Opera érdemel említést, mely kompakt és gyors, hatékonyságát más platformokon is bizonyította.

Több grafikus felület-kezelő fájlmenedzsere is kínál webböngésző funkciókat (pl. konqueror, nautilus).

10.3.4. FTP

10.3.4.1. FTP szerverek

Egy Linux rendszeren egy FTP szerver tipikusan az xinetd-ből fut, a WU-ftpd szervert használva. Ez alól kivételt képeznek azok az esetek, amikor egy nagyforgalmú FTP szervert önálló szerverként konfigurálnak. Természetesen más FTP szerverek (például Ncftpd vagy a Proftpd) használatára is lehetőség van.

A legtöbb Linux disztribúció tartalmazza az anonftp csomagot, ami beállít egy névtelen FTP szerver-fát és a megfelelő konfigurációs fájlokat.

10.3.4.2. FTP kliensek

A legtöbb Linux-on megtalálható az ncftp is, a UNIX-os ftp parancs egy fejlettebb változata (amit a Windows parancssorából is ismerhetünk). Az ncftp program további lehetőségeket kínál, a fájlnév-befejezéstől kezdve a könyvjelzőkön át a folyamatmenedzselésig. Felhasználói interfésze emellett kényelmes és érthető.

```
jani:~> ncftp tapolca
NcFTP 3.0.3 (April 15, 2001) by Mike Gleason (ncftp@ncftp.com).
Connecting to tapolca...
tapolca.hungaria.hu FTP server (Version wu-2.6.1-20) ready.
Logging in...
Guest login ok, access restrictions apply.
Logged in to tapolca.
ncftp / > help
Commands may be abbreviated. 'help showall' shows hidden and
unsupported commands.
'help <command>' gives a brief description of <command>.

ascii       cat     help    lpage   open    quote   site
bgget       cd      jobs    lpwd    page    rename  type
bgput       chmod   lcd     lrename pdir    rhelp   umask
bgstart     close   lchmod  lrm     pls     rm      version
binary      debug   lls     lrmdir  put     rmdir
bookmark    dir     lmkdir  ls      pwd     set
bookmarks   get     lookup  mkdir   quit    show
ncftp / >
```

Az FTP használatánál ne felejtsük, hogy rosszindulatú felhasználók hozzáférhetnek felhasználói nevünkhöz és jelszavunkhoz! Ajánlatos a biztonságos FTP (secure FTP) használata. Az sftp program a Secure Shell (SSH) csomag része.

10.3.5. Chat és konferencia

A kliensek és rendszerek disztribúciónként eltérnek. A legnépszerűbb programok a teljesség igénye nélkül:

- gaim: többprotokollos közvetlen üzenet-kliens, ami az MSN Messenger-rel, az ICQ-val, az IRC-vel kompatibilis (Windows és Macintosh verziója is létezik).
- gnomemeeting: videokonferencia-program UNIX-hoz
- jabber: nyílt forráskódú közvetlen üzenet-platform, ami az ICQ-val, az AIM-mel, a Yahoo-val, az MSN-nel, az IRC-vel, az SMTP-vel kompatibilis.

- JMSN: Java MSN Messenger klón, ami olyan képességekkel is fel van ruházva, melyekkel az eredeti program nem rendelkezik
- Konversation, KVIrc stb. a KDE csomaghoz
- xchat: IRC kliens X Window rendszerre

10.3.6. Levelező szolgáltatások

Egy Usenet szerver futtatása rengeteg szakértelmet és finomhangolást kíván.

Számos érdekes levelezési hírcsoport létezik a *.comp hierarchiában, ami rengeteg szöveges és grafikus klienssel elérhető. Sok levelező támogatja a hírcsoportok böngészését is.

Ha a régebbi hírcsoportok kereshető archívumára van szükségünk, érdemes ellátogatni a Google által üzemeltetett oldalakra. Nagyon hatékony segítség lehet egy ilyen oldal, hiszen könnyen előfordulhat, hogy egy felmerülő problémánk már másnak is okozott fejtörést, a megoldást pedig megosztotta a nagyközönséggel.

10.3.7. DNS

Az összes ilyen alkalmazás igényli a DNS szolgáltatást az IP-címek és a hosztnevek oda-vissza történő megfeleltetéséhez. Egy DNS szerver természetesen nem ismeri a világ összes IP-címét, de más DNS-szerverekkel kommunikálva az ismeretlen címek is lekérhetők. A legtöbb UNIX-rendszer képes futtatni a named-et, ami a bind (Berkeley Internet Name Domain) csomag része, melyet az Internet Software Consortium terjeszt. Ez egy önálló gyorsító névszerverként futhat, ami a Linux rendszerekben gyakran elő is fordul a hálózati sebesség gyorsítására.

Fő konfigurációs fájlunk az /etc/resolv.conf, ami meghatározza a Domain Name Server-ekkel való kapcsolatot:

```
search siki-soft.hu
nameserver 192.168.43.1
nameserver 193.74.209.138
```

10.3.8. DHCP

A DHCP (Dynamic Host Configuration Protocol, vagyis dinamikus hosztkonfigurációs protokoll) a nagyobb környezetekben fokozatosan kiszorítja a régi jó bootp-t. A fontos hálózati paraméterek (IP-címek, hosztok névszerverei) vezérléséhez használható. A DHCP visszafelé kompatibilis a bootp-vel.

A DHCP kliens gépek általában a DHCP kliens démont, a dhcpcd-t konfiguráló grafikus felhasználói interfész használatával kerülnek bekonfigurálásra.

10.3.9. Hitelesítő szolgáltatások

Tradicionálisan a felhasználók minden rendszeren helyileg kerülnek hitelesítésre, az /etc/passwd-ben és az /etc/shadow-ban tárolt információk alapján. Ha azonban egy hálózati szolgáltatást használunk hitelesítésre, a helyi fájlok mindig jelen vannak a rendszer adminisztratív (pl. rendszergazdai, démon és egyéb) azonosítóinál.

Ezek a fájlok gyakran a hackerek első számú célpontjai, így győződjünk meg róla, hogy a jogosultságok és tulajdonosi viszonyok a lehető legszigorúbbak:

```
bela:~> ls -l /etc/passwd /etc/shadow
-rw-r--r--      1 root   root     1803 Apr 16 16:05 /etc/passwd
-r--------      1 root   root     1116 Apr 16 16:05 /etc/shadow
```

10.3.9.1. PAM

A Linux képes a PAM (Pluggable Authentication Module) használatára, ami egy rugalmas UNIX-os hitelesítési módszer. A PAM előnyei:

- egy közös hitelesítési séma, melyet számos alkalmazással használhatunk
- anélkül implementálható különféle alkalmazásokba, hogy újra kellene fordítani azokat a PAM támogatásához
- nagyfokú rugalmasság és felügyelet a rendszergazda és az alkalmazás-fejlesztő hitelesítése felett
- az alkalmazás-fejlesztőknek nem kell programjukat egy meghatározott hitelesítési séma használatára programozniuk (így csak a program részleteivel kell foglalkoziuk)

Az /etc/pam.d könytár tartalmazza a PAM konfigurációs fájljait (/etc/pam.conf). Minden egyes alkalmazásnak vagy szolgáltatásnak saját fájlja van. A fájl minden sorának négy eleme van:

1. modul
   - auth: az aktuális hitelesítés, a tanúsítvány (csoporttagság, Kerberos jegyek) beállítása
   - account: ellenőrizzük, hogy a hozzáférés engedélyezve van-e a felhasználónak (az azonosító nem járt-e le, a felhasználó bejelentkezhet-e az adott napszakban stb.)
   - password: jelszavak beállítására szolgál
   - session: egy felhasználó hitelesítése után használjuk. Ez a modul további taszkokat hajt végre, melyek a hozzáféréshez szükségesek (pl. a felhasználó home könyvtárának becsatolása, a felhasználó postafiókjának elérhetővé tétele)

   A modulok sorrendje nagyon fontos.

2. vezérlő flag-ek: ezekből tudja a PAM, mi a teendő hiba, illetve siker esetén. Az érték lehet required, requisite, sufficient vagy optional.
3. modul útvonal: a használandó beépíthető modul elérési útvonala, általában a /lib/security
4. argumentumok: információk a modulokhoz

A jelszó-fájlokat a PAM automatikusan detektálja.

10.3.9.2. LDAP

Az LDAP (Lightweight Directory Access Protocol) egy kliens-szerver rendszer, mely globális vagy helyi könyvtár-szolgáltatások hálózaton keresztüli elérésére szolgál. Linux-on az OpenLDAP implementáció használatos. Ez tartalmazza az slapd-t, ami egy önálló szerver, illetve az slurpd-t, ami egy önálló LDAP replikáló szerver, az LDAP protokollba implementált könyvtárakat és számos hasznos alkalmazást, sőt még minta-klienseket is.

Az LDAP használatának legnagyobb előnye az adott típusú információk egyesítése. Ha például egy szervezeten belül a különféle felhasználókat egyetlen LDAP könyvtárba egyesítjük, azt bármely LDAP-ot engedélyező alkalmazás lekérheti, amelynek szüksége van erre az információra. Azok a felhasználók is elérhetik, akiknek könyvtárinformációkra van szükségük.

Az LDAP vagy X.500 Lite további előnye egyszerű implementálhatósága (az X.500-hoz képest), valamint jól definiált programozó interfésze (Application Programming Interface, API). Ezekből is következik, hogy az LDAP-ot engedélyező alkalmazások és az LDAP átjárók elterjedése várható a jövőben.

Az előnyök mellett persze a hátrányokról sem feledkezhetünk meg. Az LDAP használatához értelemszerűen az azt támogató alkalmazásokra vagy az LDAP átjárók támogatására van szükség. Bár az LDAP használata folyamatosan nő, jelenleg nincs túl sok LDAP-ot támogató alkalmazás Linux alá. További negatívum, hogy néhány hozzáférés-vezérlést nem támogat, kevesebb biztonsági tulajdonsággal rendelkezik, mint az X.500.

Mivel az LDAP egy nyílt és konfigurálható protokoll, szinte bármely, meghatározott szervezési struktúrába tartozó információtípus tárolásához használható. Erre jó példa a levélcím-keresés, a központi hitelesítés a PAM használatával kombinálva, a telefonkönyvtárak, a gép-konfigurációs adatbázisok.

10.4. Alkalmazások távoli futtatása

Számos különféle lehetőség van arra, hogy távoli gépen futtassunk alkalmazásokat, a kimenetet pedig (legyen az akár szöveg, akár grafika) saját munkaállomásunkon kapjuk meg. A kapcsolatok lehetnek biztonságosak és nem biztonságosak. Bár tanácsos a biztonságos kapcsolatok használata ahelyett, hogy kódolatlanul továbbítsuk jelszavunkat a hálózaton, érdemes megismer-

kedni néhány régebbi, nem biztonságos, de praktikus alkalmazással is. Ezek is hasznosak lehetnek ugyanis, ha például hibaelhárítást végzünk vagy különleges programokat futtatunk.

10.4.1. rsh, rlogin, telnet

A távoli bejelentkezéshez és a parancsok távoli futtatásához használatos rsh és rlogin parancsok a UNIX-ból származnak. Bár ritkán használjuk őket, mivel nem biztonságosak, szinte az összes Linux disztribúcióban fellelhetők a UNIX programokkal való visszafele kompatibilitás érdekében.

A telnet nem olyan ritkán használatos, még a rendszergazdák is alkalmazzák. A telnet az egyik leghatékonyabb eszköz fájlok távoli eléréséhez és a távoli adminisztráláshoz, mivel az internetről bárhonnan kapcsolódhatunk. Az X szerverrel kombinálva a távoli grafikus alkalmazásokat helyben is megjelenítheti jük. A telnet esetén nincs különbség a helyi gép és a távoli gép használata között.

Mivel a teljes kapcsolat kódolatlan, a telnet nagy biztonsági kockázatokat vet fel. A programok távoli futtatásához ezért ajánlatos az ssh (Secure SHell, vagyis biztonságos shell) használata (a későbbiekben még lesz róla szó).
Hátránya mellett a telnet bizonyos esetekben azonban használható. Erre lehet példa egy levelezőszerver működésének ellenőrzése:

```
[karesz@egyetem ~] telnet mailserver 25
Trying 192.168.42.1...
Connected to mailserver.
Escape character is '^]'.
220 m1.some.net ESMTP Sendmail 8.11.6/8.11.6; 200302281626
ehlo some.net
250-m1.some.net Hello blob.some.net [10.0.0.1], pleased to meet you
250-ENHANCEDSTATUSCODES
250-8BITMIME
250-SIZE
250-DSN
250-ONEX
250-ETRN
250-XUSR
250 HELP
mail from: karesz@veszprem.hu
250 2.1.0 karesz@veszprem.hu... Sender ok
rcpt to: jozsi@pest.hu
250 2.1.5 jozsi@pest.hu... Recipient ok
data
354 Enter mail, end with "." on a line by itself
test
.
250 2.0.0 g2MA1R619237 Message accepted for delivery
```

```
quit
221 2.0.0 m1.some.net closing connection
Connection closed by foreign host.
```

A telnet akkor is használható, ha ellenőrzünk egy webszervert, válaszol-e a kérésekre:

```
[karesz@egyetem ~] telnet www.siki-soft.hu 80
Trying 64.39.151.23...
Connected to www.siki-soft.hu.
Escape character is '^]'.
HEAD / ;HTTP/1.1

HTTP/1.1 200 OK
Date: Sun, 22 May 2005 19:35:24 GMT
Server: Apache/1.3.22 (UNIX) (Red-Hat/Linux)
mod_ssl/2.8.5 OpenSSL/0.9.6
DAV/1.0.2 PHP/4.0.6 mod_perl/1.24_01
Last-Modified: Sun, 06 Feb 2005 08:26:36 GMT
ETag: "70061-68-3c3565ec"
Accept-Ranges: bytes
Content-Length: 104
Connection: close
Content-Type: text/html

Connection closed by foreign host.

[karesz@egyetem ~]
```

Ez a megoldás teljesen biztonságos, hiszen sohasem kell megadnunk a felhasználói nevünket és/vagy jelszavunkat a számunkra érdekes információk lekéréséhez, így aztán senki sem férhet hozzá ezekhez az igen fontos információkhoz.

10.4.2. Az X Window rendszer

Az X Window rendszerrel a korábbiakban már megismerkedtünk. Tudjuk, hogy az X Window az X szerverrel kínál grafikus felületet.

Fontos, hogy megismerjük a különbséget az X szerver és az X kliens alkalmazás(ok) között. Az X szerver közvetlenül vezérli a kijelzőt és felelős az összes bemenetért és kimenetért (billentyűzet, egér, monitor). Az X kliens azonban nem közvetlenül éri el a bemeneti és kimeneti eszközöket. Az X szerverrel együtt kezeli a bemenetet és a kimenetet. Az X kliens végzi az igazi munkát (pl. értékek számítását, alkalmazások futtatását). Az X szerver csupán az adott kliens bementének és kimenetének kezeléséhez nyit ablakokat.

Normál működés alatt (5-ös futási szinten, grafikus módban) minden Linux munkaállomás egy X szerver is egyben, még akkor is, ha az csak kliens alkalmazásokat futtat. Az összes általunk futtatott alkalmazás (egy terminálablak,

egy böngésző, a CD-lejátszó programunk stb.) kliens az X szerverünk számára. Ebben az esetben a szerver és a kliens ugyanazon a gépen futnak.

Az X rendszer kliens/szerver természete ideális környezetet teremt az alkalmazások távoli futtatásához. Mivel a folyamat aktuálisan egy távoli gépen fut, a helyi gépen csak nagyon kicsi a processzor igénybevétele.

Az olyan gépeket, melyek pusztán X szerverként működnek, X terminálloknak nevezzük (ezek egykor nagyon népszerűek voltak).

10.4.2.1. A Telnet és az X

Ha a telnet-et szeretnénk használni távoli gépen futó grafikus alkalmazások megjelenítéséhez, először a távoli gép elérhetőségét kell megadnunk a kijelzőnknek (az X szerverünknek) az xhost parancs használatával. A helyi gépünk terminálablakában kiadandó parancs szintaktikája:

```
robi:~> xhost +tavoli-gep.hu
```

Ezután kapcsolódunk a távoli hoszthoz s a DISPLAY környezeti változó állításával tudatjuk vele, hogy a grafikus felületet a helyi gépen jelenítse meg:

```
[robi@tavol ~] export DISPLAY="helyi-gep.hu:0.0"
```

Miután végeztünk ezzel a lépéssel, minden, ebben a terminálablakban elindított alkalmazás a helyi gépünkön jelenik meg. Ekkor a számításokhoz a távoli erőforrások lesznek felhasználva, de az alkalmazást a helyi grafikus erőforrások (az X szerver) jeleníti meg.

Ez a folyamat feltételezi, hogy már van néhány X szerver (Xfree86, Exceed, Cygwin) azon a gépen, amit megjelenítésre szeretnénk használni. A kliens gép architektúrája és operációs rendszere lényegtelen. Az egyetlen fontos dolog, hogy tegyék lehetővé egy X szerver futtatását.

10.4.3. SSH

A telnet biztonsági problémáival felhagyva a legtöbb Linux rendszer manapság az SSH-t használja. A legtöbb Linux rendszer az OpenSSH egyik változatát használja, ami az SSH protokoll nyílt forráskódú implementációja. Segítségével megbízhatatlan hálózaton keresztül titkosítva kommunikálhatunk megbízhatatlan gépekkel.

A standard beállításnál az X kapcsolatok automatikusan továbbítódnak, de tetszőleges TCP/IP portok is elérhetők egy biztonságos csatorna használatával.

Az ssh kliens kapcsolódik és bejelentkezik a megadott hosztnévre. A felhasználónak azonosítania kell magát a távoli gépnek, amit az sshd_config fájl határoz meg (ami általában az /etc/ssh-ban van). A konfigurációs fájl általában magyarázatot nem igényel és alapból engedélyezi a legtöbb lehetőséget.

Ha a felhasználó azonosságát a szerver elfogadta, vagy futtatja a megadott parancsot vagy bejelentkezik a gépre és a felhasználónak egy normál shell-t kínál a távoli gépen. A távoli paranccsal vagy shell-lel végzett kommunikáció automatikusan kódolásra kerül.

A folyamat terminál, ha a parancs vagy shell a távoli gépen kilép és minden X11 és TCP/IP kapcsolat bezárásra kerül.

Ha először csatlakozunk egy hoszthoz az SSH gyűjtemény valamelyik programjával, meg kell állapítanunk a hosztnak a hitelességét:

```
adam ~> ssh disz
The authenticity of host 'disz (10.0.0.1)' can't be established.
RSA fingerprint is 18:30:50:46:ac:98:3c:93:1a:56:35:09:8d:97:e3:1d.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'disz,192.168.30.2' (RSA) to the list of
known hosts.
Last login: Thu Aug 25 11:49:39 2005 from octarine
This space for rent.

adam is in ~
```

Ilyenkor fontos, hogy az igenlő választ ne egy y-nal, hanem egy yes-szel adjuk meg. Ez a lépés a ~/.ssh/known_hosts fájlt szerkeszti.

Ha csak ellenőrizni szeretnénk valamit egy távoli gépen, majd a helyi gépen szeretnénk visszakapni a promptot, az ssh argumentumaként adhatjuk meg a távolban futtatandó parancsokat:

```
vera ~> ssh disz who
istvan@disz's password:
root     tty2         Jun 22 10:58
eva      tty3         Jun 21 21:22
eva      0:           Jun 23 21:03

vera ~> uname -n
siki-weblap.hu
```

10.4.3.1. X11 és TCP továbbítás

Ha az X11Forwarding értéke yes-re van állítva és a felhasználó X alkalmazásokat használ, a DISPLAY környezeti változó be van állítva, a kapcsolat az X11 kijelzéshez automatikusan továbbítódik a távolba. Az összes shell-ből indított X11 program a kódolt csatornán megy keresztül, a kapcsolat az igazi X szerverhez a helyi gépből lesz kiépítve. A felhasználó manuálisan nem állíthatja a DISPLAY változót. Az X11 kapcsolatok továbbítása a parancssorban vagy az sshd konfigurációs fájlban konfigurálhatók.

Az ssh által beállított DISPLAY értéke a szerver gépre mutat (0-nál nagyobb kijelző-számmal). Ennek oka, hogy az ssh létrehoz egy proxy X szervert a szerver gépen a kapcsolatok kódolt csatornán történő továbbításához.

Ez mind automatikusan megy végbe, így ha egy grafikus alkalmazás nevében gépelünk, az helyi gépünkön jelenik meg és nem a távoli hoszton.

Az SSH automatikusan beállítja az X hitelességi adatokat a szerver gépen. Ehhez generál egy véletlenszerű hitelesítő sütit, letárolja az Xauthority-ban a szerveren, majd ellenőrzi, hogy az összes továbbított kapcsolat viszi-e ezt a sütit. A kapcsolat megnyitásakor az igazi sütivel helyettesíti. A valódi hitelesítő süti soha nem kerül elküldésre a szerver gépnek.

Tetszőleges TCP/IP kapcsolatok biztonságos csatornán való továbbítása akár a parancssorban, akár egy konfigurációs fájlban meghatározható.

A folyamat feltételezi, hogy fut egy X szerver azon a kliens gépen, melyen a távoli hoszt alkalmazását megjeleníteni szeretnénk. Ez a kliens gép felépítésében és operációs rendszerében eltérhet a távoli géptől, ha az tud futtatni egy X szervert.

10.4.3.2. Szerver-hitelesítés

Az ssh kliens/szerver rendszer automatikusan karbantartja és ellenőrzi az összes hoszt azonosítását tartalmazó adatbázist. A hoszt kulcsok a felhasználó home könyvtárában, a $HOME/.ssh/known_hosts-ban vannak tárolva. Az /etc/ssh/ssh_known_hosts fájlt automatikusan ellenőrzi a rendszer az ismert hosztok miatt. Minden új hoszt automatikusan hozzáadódik a felhasználó fájljához. Ha egy hoszt azonosítása egyszer is megváltozik, az ssh figyelmeztet és letiltja a jelszó-hitelesítést (megelőzve egy esetleges trójai falovat, mely a felhasználó jelszavából származhat). Az olyan környezetekben, ahol magasszintű biztonságra van szükség, az sshd úgy is konfigurálható, hogy letiltsa a bejelentkezést az olyan gépek esetén, melyek hoszt kulcsa megváltozott vagy ismeretlen.

10.4.3.3. Biztonságos távoli másolás

Az SSH csomag kínálja az scp-t az rcp parancs (ami akkor volt használatos, mikor csak rsh létezett) biztonságos alternatívájaként. Az scp az ssh-t használja adatátvitelre, ugyanazt a hitelesítést alkalmazza, biztonsága tehát az ssh-val megegyezik. Az rcp-vel ellentétben az scp jelszavakat kér a hitelesítéshez:

```
geza /var/tmp> scp terv.gz disz:/var/tmp/
geza@disz's password:
terv.gz 100% |*****************************|  80 KB 00:00
geza /var/tmp>
```

Minden fájlnév tartalmazhat egy hosztot és felhasználói specifikációt annak jelzésére, hogy a fájlt erre a hosztra (illetve erről a hosztról) kell másolni. A két távoli hoszt közötti másolás nem engedélyezett.

Ha inkább egy FTP-szerű interfészt szeretnénk használni, alkalmazhatjuk az sftp-t is:

```
geza /var/tmp> sftp disz
Connecting to disz...
geza@disz's password:

sftp> cd /var/tmp

sftp> get ter*
Fetching /var/tmp/terv.gz to terv.gz

sftp> bye

geza /var/tmp>
```

10.4.3.4. Hitelesítési kulcsok

Az ssh-keygen parancs hitelesítési kulcsokat generál, menedzsel és konvertál az ssh számára. Létrehozhat RSA kulcsot az 1-es verziójú SSH protokollhoz, illetve RSA vagy DSA kulcsokat a 2-es verziójú SSH protokollal való munkához.

Normál esetben minden felhasználó, aki az SSH-t RSA vagy DSA hitelesítéssel szeretné használni, egyszer futtatja a parancsot a hitelesítési kulcs $HOME/.ssh/identity-ben, id_dsa-ban vagy id_rsa-ban történő létrehozásához. A rendszergazda emellett ennek használatával hoszt kulcsokat generálhat a rendszerhez.

A program generálja a kulcsot és kér egy fájlt a privát kulcs tárolásához. A publikus kulcs egy ugyanilyen nevű, de .pub végű fájlba kerül. A program egy jelszókifejezést is kér. Ez lehet üres (jelezve, hogy nincs jelszókifejezés) vagy egy tetszőleges hosszúságú string.

Egy esetlegesen elveszett jelszókifejezés visszanyerésére nincs lehetőség. Ha a jelszókifejezés elveszett vagy netán elfelejtettük, egy új kulcsot kell generálni és azt bemásolni a megfelelő publikus kulcsokba.

10.4.4. VNC

A VNC (Virtual Network Computing) egy távoli megjelenítő rendszer, ami lehetővé teszi, hogy a grafikus környezetet ne csak a helyi gépről nézhessük, amelyen az fut, hanem bárhonnan az internetről, valamint számos különféle felépítésű számítógépről (Windows és számos UNIX disztribúció alól). Lehetőség van például arra, hogy egy Microsoft Word szövegszerkesztő programot futtassunk egy Windows NT alapú rendszeren, a kimenetet pedig egy Linux asztalon nézhetjük meg.

A VNC szervereket és klienseket egyaránt kínál, így a Linux programokat akár Windows klienseken is megjeleníthetjük. A VNC az egyik legegyszerűbb megoldás X kapcsolatokhoz személyi számítógépeken. Az alábbi tulajdonságai teszik a VNC-t mássá, mint egy normál X szerver vagy egy kereskedelmi implementáció:

- a néző oldalán állapot nem kerül mentésre: ha egy PC X szervert futtatunk és a rendszer megakad vagy a gép újraindul, az összes távolban futtatott alkalmazásunk meghal. A VNC-vel más a helyzet, mert itt a távoli alkalmazások tovább futnak.
- ingyenes: GPL licensznek megfelelően használható
- kicsi és egyszerű: nem szükséges telepíteni, ha szükséges, még egy floppyról is futtatható
- megosztható: egyetlen felület több kijelzőn is megjeleníthető
- platform-független: a Java kliens miatt virtuálisan mindenhol fut, ami támogatja az X-et

10.4.5. Az rdesktop protokoll

A Microsoft Windows hosztok menedzselésének megkönnyítésére az újabb Linux disztribúciók támogatják a Remote Desktop Protocol-t (RDP), ami az rdesktop kliensbe van implementálva. A protokolt számos Microsoft termék használja (pl. Windows NT Terminal Server, Windows 2000 Server, Windows XP és Windows 2003 Server).

10.5. Biztonság

Amint egy számítógépet hálózatba kötünk, mindenféle visszaélésre kell számítani. Sokan megkérdezhetik maguktól, kell-e foglalkozniuk a biztonság kérdésével (mert pl. egy meglehetősen védettnek tűnő irodában dolgoznak). A válasz helyett valójában azzal a kérdéssorozattal kell foglalkoznunk, hogy szeretnénk-e

- saját rendszerünk önállósodásával bajlódni?
- (tudtunk nélkül) részt venni bűncselekményekben?
- eszközeinket megosztani másokkal?
- kockáztatni internet-kapcsolatunkat?
- minden alkalommal rendszerhelyreállítással foglalkozni, mert valaki betört a rendszerünkbe?
- személyes vagy fontos adatokat elveszíteni?

Valószínűleg mindannyian nemmel válaszolunk, ezért fontos kicsit boncolgatni a biztonsági kérdéseket.

10.5.1. Szolgáltatások

Alapvető biztonsági fogás, hogy a lehető legkevesebb szolgáltatást futtassuk. Minél kevesebb nyitott portunk van a nagyvilág felé, annál könnyebben áttekinthetők. Ha a szolgáltatások nem kapcsolhatók ki a helyi hálózat számára, próbáljuk meg legalább a kívülről jövő kapcsolatokat letiltani.

Akik már ismerik a rendszerüket, azoknak egyszerű szabály, hogy minden olyan szolgáltatás, melyet nem ismernek, nem szükséges. Ne felejtsük el azt sem, hogy nem minden szolgáltatás az interneten használatos. Ellenőrizzük,

hogy az egyes szolgáltatások mely TCP portot figyelik. Ehhez adjuk ki a netstat parancsot:

```
[nori@keszthely ~] netstat -l | grep tcp
tcp        0      0 *:32769                 *:*                     LISTEN
tcp        0      0 *:32771                 *:*                     LISTEN
tcp        0      0 *:printer               *:*                     LISTEN
tcp        0      0 *:kerberos_master       *:* LISTEN
tcp        0      0 *:sunrpc *:* LISTEN
tcp        0      0 *:6001 *:* LISTEN
tcp        0      0 *:785 *:* LISTEN
tcp        0      0 localhost.localdom:smtp *:* LISTEN
tcp        0      0 *:ftp *:* LISTEN
tcp        0      0 *:ssh *:* LISTEN
tcp        0      0 ::1:x11-ssh-offset *:* LISTEN
```

Néhány fontos dolog, amire érdemes vigyázni:

- az exec, rlogin és rsh, illetve a telnet csak a biztonságos oldalon lehet
- a szerver gépeken legyen az X11
- ha nincs nyomtató fizikailag csatlakoztatva, nem kell lp
- ha nincs Microsoft Windows hoszt a hálózaton, nem kell a Samba
- ne engedélyezzünk FTP-t, csak ha FTP szerverre van szükség
- ne engedélyezzünk NFS-t és NIS-t az interneten keresztül, tiltsunk le minden, nem szükséges szolgáltatást egy önálló telepítésnél
- ne futtassunk MTA-t, hacsak aktuálisan nem egy levelezőszerveren vagyunk

A szolgáltatásokat a chkconfig használatával, inicializáló scriptekkel vagy az (x)inetd konfigurációs fájljainak szerkesztésével állíthatjuk le.

10.5.2. A rendszeres frissítés jelentősége

Mivel a Linux rendszer és az alá írt alkalmazások rendkívül dinamikusan fejlődnek, az újabb szoftver-verziók szinte napról napra jelennek meg. Az újabb változatok a korábbiakban időközben felfedezett hibák javításai miatt nemcsak stabilabbak, de biztonságosabbak is lehetnek elődeiknél. Ezért fontos a rendszeres frissítés. Ez vonatkozik a legkülönfélébb csomagokra, de akár magára az operációs rendszerre is.

Egyes frissítések új konfigurációs fájlokat igényelnek, a régi fájlok pedig lecserélődnek. Az adott program dokumentációjában ellenőrizhetjük, hogy a frissítés során (és után) minden rendben zajlik-e.

A legtöbb Linux disztribúció levelezőlista-szolgáltatásokat is nyújt a biztonságos frissítések kihirdetéséhez. Emellett a rendszer frissítéséhez szükséges programokat is többnyire megtaláljuk disztribúciónkban.

A frissítés egy véget nem érő folyamat, melyet szinte bármely nap végezhetünk (folyamatosan működő rendszereken érdemes éjszakai órákra ütemezni, így a folyamat egy része automatizálható).

10.5.3. Tűzfalak

A tűzfal egy eléggé tág (és bizonytalan) fogalom, mely alapjában véve egy védőkorlátot jelent köztünk és a külvilág, általában az internet között. Lehet egy konkrét védelmi rendszer vagy egy ilyen funkcionalitással rendelkező különleges program. Lehet komponensek kombinációja is, mely számos hardver- és szoftverkombinációt tartalmaz. A tűzfalak úgynevezett szabályokból építkeznek, melyek azt írják le, mit lehet tenni egy adott rendszerbe vagy hálózatba történő belépéskor és/vagy kilépéskor.

Miután letiltottuk a szükségtelen szolgáltatásokat, a szükséges szolgáltatások számára a lehető legkevesebb kapcsolatot engedélyezzük. Jó példa erre egy otthonról dolgozó távmunkás, aki csak az irodai géppel kommunikál, az internet más gépeinek kapcsolatai pedig blokkolva vannak.

10.5.3.1. Csomagszűrők

A védelem első vonalában állnak a csomagszűrők, melyek belenézhetnek az IP csomagokba és a tartalomtól függően döntést hozhatnak. Az ipchains tűzfalat futtató rendszerek a 2.2-es kernelen alapulnak. Az újabb rendszerek (2.4-es kernel) már az iptables-t használják, ami a Linux-os csomagszűrők következő generációja, valamint a Gnome Lokkit programját. Ez az eszköz a normál felhasználók számára nyújt egyszerű interfészt. Beállít egy alapvető tűzfal-konfigurációt az asztalra, egy tárcsázós vagy kábelmodem-kapcsolatot. Nagyobb környezetekben nem használatos.

Az újabb kernelek lehetőségei között szerepel az is, hogy nemcsak azt nézhetjük meg, mi van a csomagokban, hanem még azt is megtudhatjuk, ha egy csomag egy új vagy egy létező kapcsolattal összefügg.

10.5.3.2. Proxy-k

A proxy-k különféle feladatok elvégzésére is alkalmasak. Ezek nem mindegyike biztonsági jellegű. Mivel azonban a proxy-k közvetítők, rendkívül alkalmasak a hozzáférés-vezérlés kikényszerítésére, a közvetlen kapcsolatok tűzfalon keresztüli korlátozására és annak vezérlésére, hogy a mögötte levő hálózat hogyan nézzen az internetre.

Gyakran csomagszűrővel is kombinálják, de sokszor csak az egyiket alkalmazzák. Így vagy úgy, de a proxy a vezérlés egy külön szintjét jelenti.

10.5.3.3. Egyedi alkalmazások elérése

Egyes szervereknek lehet saját hozzáférés-vezérlése is. Ilyen például a Samba, az X11, a Bind, az Apache vagy a CUPS. Minden felkínálandó szolgáltatáshoz ellenőrizzük az alkalmazott konfigurációs fájlokat.

10.5.3.4. Naplófájlok

A UNIX-alapú rendszerek a legkülönfélébb műveleteket különféle fájlokba naplózzák. A naplófájlokat természetesen rendszeresen ellenőrizni kell, akár kézzel, akár automatikusan. A tűzfalak és a különféle hozzáférés-vezérlések hatalmas mennyiségű naplófájlt készítenek, így érdemes csak az abnormális műveleteket naplózni.

10.5.4. Betörés-észlelés

A betörés-észlelő rendszerek arra lettek tervezve, hogy ami a tűzfalon véletlenül átjött, azt elkaphassuk. Kétféle típus létezik. Az egyik egy aktív betörési folyamatot észlel, a másik csak értesít minket a sikeres betörésről. Az utóbbi esetben túl késő bárminemű adatsérülést megelőzni, de legalább tudomást szerzünk a szomorú tényről. A betörés-észlelés két fő típusa a hálózatokat, illetve az egyéni hosztokat védő változat.

A hoszt-alapú rendszer olyan alkalmazásokkal dolgozik, melyek észlelik a fájlrendszer változásait. Bármilyen módon is változnak meg a rendszerfájlok, az jót nem jelent. Bárki is jut be a rendszerünkbe és fér hozzá rendszergazdai jogosultságokhoz, feltehetően változásokat idéz elő a rendszerben. Egyes rendszerek tartalmazzák a tripwire megfigyelő rendszert, mely ilyen esetekben hasznos lehet.

10.5.5. Talán betörtek hozzánk

A leggyakoribb árulkodó jelek az alábbiak:

- a rendszereszközök (belső parancsok) működése szokatlan
- az interfészek szokatlan módban vannak
- azonosíthatatlan ideiglenes fájlok
- bejelentkezési problémáink vannak
- furcsa elérések a shell kiadott parancsait tartalmazó fájlokban
- furcsa nyitott portok, különös folyamatok
- indokolatlan sávszélesség-használat
- sérült vagy hiányzó naplófájlok
- váratlanul módosított konfigurációs fájlok

10.5.6. Tippek és trükkök

Az alábbi általános irányelveket betartva kevesebb meglepetés érhet minket:

- a privát szolgáltatásokat szokatlan portokon futtassuk
- a rendszergazda leveleit más ne olvashassa el
- az új szoftverek eredetét ellenőrizzük, a csomagokat ne telepítsük ellenőrzés nélkül
- ha nem állandó internet-kapcsolatunk van, állítsuk le, amíg nincs rá többé szükség
- komolyan bánjunk a jelszavakkal és rendszeresen változtassuk meg őket
- korlátozzuk az erőforrásokat a quota és/vagy az ulimit használatával
- minél jobban ismerjük meg saját rendszerünket, mert így egy idő után megérezzük, ha valami szokatlan történik
- ne engedélyezzünk rendszergazdai bejelentkezéseket
- próbáljunk meg mindig SSH-t használni és mellőzük a telnet-et, az FTP klienseket és más kliens programokat (pl. a POP3 levelezőklienst, ami kódolatlan jelszavakat küld a hálózaton keresztül)

10.5.7. Helyreállítási munkálatok

A legfontosabb, hogy ne veszítsük el a hidegvérünket. A helyreállítás fő lépései:

1. csatoljuk le a gépet a hálózatról
2. próbáljunk meg minél többet megtudni róla, hogyan törtek be hozzánk
3. készítsünk biztonsági mentéseket azokról a fontos adatokról, melyek nem rendszeradatok
4. telepítsük újra a rendszert
5. használjunk új jelszavakat
6. állítsuk vissza a rendszer eredeti beállításait és adatainkat
7. frissítsünk minden lehetséges szoftvert
8. vizsgáljuk meg újra a rendszert, a szükségtelen szolgáltatásokat állítsuk le, ellenőrizzük a tűzfal-szabályokat és a hozzáférési jogosultságokat
9. csatlakozzunk újra

FÜGGELÉK

A. Grafikus alkalmazások, alkalmazástípusok

A számos Linux disztribúció mindegyikéhez más és más programokat, programcsomagokat mellékelnek a rendszer készítői. Ezek a legkülönfélébb feladatok elvégzésére alkalmasak. Az egyik népszerű változat, a SuSE 8.0-s verziójának főbb alkalmazásai:

Alkalmazás, csomag	Kategória
OpenOffice.org	irodai programcsomag (szöveg-szerkesztő, táblázatkezelő, stb.)
Acrobat Reader	PDF-olvasó
Konqueror, Galeon	böngésző
KMail	levelezőprogram
Alsamix, alsamixergui, kscd, alsaplayer, xmms, aRts, kradio	audiokeverő, -szerkesztő,-lejátszó
Broadcast2000, motv, alevt, nxtvepg	videoszerkesztő, TV-s és teletex-tes alkalmazás, webkamera
KonCD	CD író
GIMP, KPaint	képszerkesztő
Kooka	lapolvasóprogram

B. Folyamatkezelő parancsok

Parancs	Jelentés
at	Feladatok sorba rendezése későbbi futtatáshoz.
atq	A felhasználó megakadt feladatainak kilistázása.
atrm	A feladatszámukkal megadott feladatok törlése.
batch	Akkor futtatja a parancsokat, ha a rendszer terheltsége ezt lehetővé teszi.
crontab	Egyéni felhasználók crontab fájljainak karbantartásához használható.
halt	A rendszer megállítása.
init futasi_szint	Folyamatvezérlés inicializálása.
jobs	Az aktuálisan futó feladatok kiírása.
kill	Egy folyamat befejezése.
mesg	Írási jogosultság vezérlése terminálunkhoz.
netstat	Hálózati kapcsolatok, interfész-statisztikák és egyéb hálózati információk megjelenítése.
nice	Egy program futtatása módosított prioritású ütemezéssel.
ps	A folyamat állapotáról szolgál információval.
pstree	Egy fastruktúrában jeleníti meg a folyamatokat.
reboot	Leállítja/újraindítja a rendszert.
renice	Futó folyamatok prioritásának megváltoztatása.
shutdown	A rendszer leállítása.
sleep	Megadott ideig történő várakozás.
time	Információk egy parancs futási idejéről, illetve az erőforrás-használatról.
top	A legmagasabb CPU-folyamatok megjelenítése.
uptime	Megtudhatjuk, mennyi ideig futott a rendszer.
vmstat	Statisztikák a virtuális memóriáról.
w	Megmutatja, kik vannak bejelentkezve és mit csinálnak.
wall	Üzenet küldése az összes felhasználó terminájára.
who	Megtudhatjuk, kik vannak bejelentkezve.
write	Üzenet küldése egy másik felhasználónak.

C. Nyomtatási parancsok

Parancs	Jelentés
lpr	fájl nyomtatása
lpq	nyomtatási sor lekérése
lprm	nyomtatási feladat elvetése
groff	formázó eszköz
gv	PostScript nézegető
printconf	nyomtató-konfiguráló
xdvi	DVI nézegető
xpdf	PDF nézegető
*2ps	fájlkonvertálás PostScript formátumba

D. Archíválási parancsok

Parancs	Jelentés
bzip2	egy blokkrendező fájltömörítő
cdrecord	audio CD vagy CD-ROM másolása írható lemezre
dd	egy fájl konvertálása és másolása
fdformat	egy floppy lemez alacsony szintű formázása
gzip	fájlok tömörítése és kicsomagolása
mcopy	MS-DOS fájlok másolása UNIX-alapú rendszerre vagy rendszerről
mdir	MS-DOS könyvtár megjelenítése
mformat	MS-DOS-os fájlrendszer létrehozása egy alacsony szinten formázott lemezen
mkbootdisk	egy önálló rendszerbetöltő lemez készítése a futó rendszerhez
mount	egy fájlrendszer becsatolása (beintegrálása az aktuális fájlrendszerbe egy csatlakozási ponthoz történő csatlakoztatásával)
tar	elsősorban szalagos archíválásra készült eszköz, mellyel más meghajtókra is készíthetünk biztonsági másolatot
umount	fájlrendszerek lecsatolása

E. Egyéb parancsok

Parancs	Jelentés
acroread	PDF-nézegető
adduser	Létrehoz egy új felhasználót vagy frissíti az alapértelmezett új felhasználó-információt
alias	Létrehoz egy shell aliaszt egy parancsnak
anacron	Periódikusan futtat parancsokat (nem tévesztendő össze a folyamatosan futó géppel!)
apropos	A whatis adatbázisban keres stringet
apt-get	APT csomagkezelő program
aspell	Betűzés-vizsgáló
aumix	Az audio-mixer beállítása
awk, gawk	Mintavizsgáló és -feldolgozó nyelv
bash	Bourne Again SHell
bg	Egy folyamat futtatása a háttérben
bitmap	Bitképes szerkesztő és konverter az X Window rendszerhez.
cat	Fájlok összefűzése és standard kimenetre történő kiírása
cd	Könyvtárváltás
cdp, cdplay	Egy interaktív szöveges módú program Audio CD-k vezérlésére, lejátszására
cdparanoia	Audio CD-lejátszó program, mely különleges adatvizsgálatokat is támogat
chattr	Fájlattribútumok megváltoztatása
chgrp	Csoport tulajdonjogának megváltoztatása
chkconfig	Futási szint-információk frissítése vagy lekérése rendszerszolgáltatásokhoz
chmod	Fájl hozzáférési jogosultságainak megváltoztatása
chown	Fájltulajdonos és -tulajdonoscsoport megváltoztatása
compress	Fájlok tömörítése
cp	Fájlok és könyvtárak másolása
csh	Egy C shell megnyitása
cut	Részek eltávolítása fájl(ok) minden sorából
date	Kiírja vagy beállítja a rendszerdátumot és -időt
df	Jelentés a fájlrendszer lemezhasználatáról
dhcpcd	DHCP kliens démon
diff	Különbségek keresése két fájl között
dig	Domain név kérés-csomagok küldése névszerverekhez

Parancs	Jelentés
dmesg	Kernel puffer kiírása vagy vezérlése
du	Becslés a fájl helyhasználatáról
echo	Egy sor szöveget jelenít meg
egrep	Kibővített grep
eject	Eltávolítható média (pl. CD) lecsatolása és kiadása
emacs	Az Emacs szerkesztő indítása
exec	Alfolyamat(ok) meghívása
exit	Kilépés az aktuális shellből
export	Funkció(k) hozzáadása a shell környezethez
fdisk	Partíciós tábla-manipulátor
fetchmail	Levél fogadása egy POP-os, IMAP-os, ETRN-es vagy ODMR-es szervertől
fg	Egy folyamat előhozása az előtérbe
file	Fájltípus megállapítása
find	Fájlok keresése
formail	Levél(újra)formázó
fortune	Egy véletlenszerű, remélhetőleg érdekes mondást jelenít meg
ftp	Fájlátvitel (névtelen azonosítónál nem biztonságos)
galeon	Grafikus webböngésző
gdm	Gnome Display Manager
getty, mingetty, agetty	Konzoleszközök vezérlése
gimp	Képmanipuláló program
grep	Egy mintával megegyező sorokat nyomtat
grub	A grub shell
head	Fájlok első részének kiírása
help	A shell egy beépített parancsának súgója
host	DNS-kereső
httpd	Apache hiperszöveges átviteli protokoll szerver
id	Valódi és effektív felhasználói és csoportazonosítókat jelenít meg
ifconfig	Hálózati interfész konfigurálása vagy konfiguráció megmutatása
info	Info dokumentumok olvasása
iostat	I/O statisztika megjelenítése
ip	Hálózati interfész-állapot megjelenítése vagy megváltoztatása
ipchains	IP tűzfal adminisztráció
iptables	IP csomagszűrő adminisztráció

Parancs	Jelentés
jar	Java archíváló program
kdm	KDE ablakozó
kill, killall	Folyamat(ok) terminálása
ksh	Egy Korn shell megnyitása
ldapmodify	Egy LDAP módosítása
ldapsearch	LDAP kereső
less	a more bővített változata
lilo	Linux boot loader (rendszerbetöltő)
links	Szöveges webböngésző
ln	Hivatkozások készítése fájlokhoz
loadkeys	Billentyűzet-kiosztás betöltése
locate	Fájlok keresése
logout	Az aktuális shell bezárása (kijelentkezés)
lp	Kérések küldése az lp nyomtatószolgáltatásnak
lpc	Sornyomtató vezérlőprogram
ls	Könyvtártartalom listázása
lynx	Szöveg-alapú webböngésző
mail	Levélküldés és -fogadás
man	Man oldal olvasása (a manual, azaz kézikönyv rövidítése)
memusage	Memóriahasználat megjelenítése
memusagestat	Memóriahasználati statisztika megjelenítése
mkisofs	Írási jogosultság vezérlése terminálunkhoz
mplayer	Filmlejátszó és -kódoló
more	A megjelenítendő szöveg oldalakra tördelésére szolgáló szűrő
mozilla	Webböngésző
mt	Mágnesszalagos meghajtó vezérlése
mtr	Hálózat-diagnosztikai eszköz
mv	Fájlok átnevezése
named	Internet domain névszerver
ncftp	Böngészőprogram FTP-szolgáltatásokhoz (nem biztonságos)
nfsstat	Hálózati kapcsolatok, útvonal-táblázatok, interfész-statisztikák stb. megjelenítése
nmap	Hálózat- és biztonság-vizsgáló
ntsyssv	Egyszerű interfész futási szintek konfigurálásához
ogle	DVD-lejátszó
passwd	Jelszó megváltoztatása
perl	Practical Extraction and Report Language

Parancs	Jelentés
pg	Oldalas szövegkimenet
ping	Tesztkérés küldése egy hosztnak
pr	Szöveges fájlok konvertálása nyomtatáshoz
printenv	A környezet egészének vagy egy részének nyomtatása
procmail	Önálló levélfeldolgozó
pwd	Az aktuális munkakönyvtár kiírása
quota	Lemezhasználat megjelenítése és korlátozások beállítása
rcp	Távoli másolás (nem biztonságos)
rdesktop	Remote Desktop Protocol kliens
reboot	A rendszer leállítása (újraindítás)
rlogin	Távoli bejelentkezés (telnet, nem biztonságos)
rm	Egy fájl áthelyezése
rmdir	Egy könyvtár áthelyezése
rpm	RPM csomagkezelő
rsh	Távoli shell (nem biztonságos)
scp	Biztonságos távoli másolás
screen	Képernyő-menedzser VT100 emulációval
set	Változó megjelenítése, beállítása vagy megváltoztatása
setterm	Terminál-attribútumok beállítása
sftp	Biztonságos (kódolt) FTP
sh	Egy standard shell megnyitása
slocate	A GNU-s locate biztonságos változata
slrnn	Szöveges Usenet kliens
snort	Hálózati betörés-észlelő
sort	Szövegfájlok sorainak rendezése
ssh	Biztonságos shell
ssh-keygen	Hitelesítési kulcs generálása, menedzselése és konverziója
stty	Terminál sor-beállítások megváltoztatása és megjelenítése
su	Felhasználó-váltás
tac	Fájlok fordított összefűzése és nyomtatása
tail	Fájlok utolsó részének megjelenítése
talk	Beszéd egy felhasználóhoz
tcsh	Egy Turbo C shell megnyitása
telnet	Felhasználói interfész a Telnet protokollhoz (nem biztonságos)
tex	Szövegformázás és gépelési beállítások
tin	Hírolvasó program
touch	Fájl időjellemzőinek megváltoztatása
traceroute	Az útcsomagok megjelenítése hálózati hoszthoz

Parancs	Jelentés
tripwire	Fájlintegritás-ellenőrző
troff	Dokumentumok formázása
tvime	Nagytudású televíziós alkalmazás
twm	Ablakkezelő az X Window rendszerhez
ulimit	Erőforrások vezérlése
umask	Felhasználó fájllétrehozási maszkjának beállítása
uncompress	Tömörített fájlok kicsomagolása
uniq	Egy rendszerezett fájlból eltávolítja a dupla sorokat
uptime	Átlagos rendszer-terheltség megjelenítése
userdel	Egy felhasználói azonosító és a kapcsolódó fájlok törlése
vi, vim	Nagytudású, de „fapados" szövegszerkesztő
vimtutor	A vim szövegszerkesztő oktatóprogramja
wc	Megadott számú bájt, szó vagy sor kiírása fájlokban
which	(Shell) parancsok teljes elérési útvonalának megjelenítése
whoami	Az effektív felhasználói azonosító megjelenítése
whois	Egy whois (jelentése különírva: ki ez) vagy nickname (becenév) adatbázis lekérdezése
xauth	X hitelességi program
xawtv	Egy X11 program TV-nézéshez
xcdroast	Grafikus felület CD-íráshoz
xclock	Analóg/digitális óra az X-hez
xconsole	Rendszerkonzol-megfigyelő
xdm	X Display Manager XDMCP-támogatással (hoszt-választó)
xfs	X fontszerver
xhost	Szerver hozzáférés-vezérlő X-hez
xine	Ingyenes videólejátszó
xinetd	Egy bővített internet-szolgáltatás démon
xload	Átlagos rendszerterheltség megjelenítése X-hez
xlsfonts	Szerver-fontlista megjelenítő X-hez
xmms	Zenelejátszó X-hez
xterm	X-es terminálemulátor
zapping	Egy TV-néző program Gnome-hoz
zcat	Fájlok tömörítése vagy kicsomagolása
zgrep	Reguláris kifejezés keresése tömörített fájlokban
zmore	Szűrő tömörített szöveg megtekintéséhez

F. A DOS-os (Windows-os) és a Linux-os parancsok összehasonlítása

Akik úgy szeretnék megismerni a Linux operációs rendszert, hogy a Windows parancssorát, illetve a DOS-os parancsokat ismerik, érdemes párhuzamot vonniuk a hasonló hatású parancsok között. Azt azonban ne felejtsük, hogy sok parancsnak számos opciója is van.

DOS/Windows parancs	**Linux parancs**
parancs /?	man parancs
cd	cd
chdir	pwd
cls	clear
copy	cp
date	date
del	rm
dir	ls
echo	echo
edit	vim (vagy más szerkesztőprogram)
exit	exit
fc	diff
find	grep
format	mke2fs vagy mformat
mem	free
mkdir	mkdir
more	more vagy less
move	mv
ren	mv
time	date

G. Alapvető shell-parancsok

Érdemes külön is tudatosítani, hogy mely parancsok közösek minden Linux-shellben:

Parancs	**Jelentés**
>	kimenet átirányítása
>>	hozzáfűzés fájlhoz
<	bemenet átirányítása

Parancs	Jelentés
<<	bemenet átirányítása
&	folyamat futtatása a háttérben
;	parancsok tagolása egy soron belül
*	bármely karakter egyezősége fájlnévben
?	adott karakter egyezősége fájlnévben
[ ]	a megadott karakterek egyezősége
( )	futtatás alshellben
` `	a megadott kimenet helyettesítése
" "	részleges idézet (változó- és parancsbővítés engedélyezése)
' '	teljes idézet (nincs bővítés)
$valtozo	érték használata változóként
$$	folyamatazonosító
$0	parancsnév
$n	n-edik argumentum (az n 0 és 9 között lehet)
$*	minden argumentum egy egyszerű szó
#	magyarázat
bg	futtatás háttérben
break	ciklikus utasítások megszakítása
cd	könyvtárváltás
continue	egy programciklus folytatása
echo	kimenet megjelenítése
eval	argumentumok kiértékelése
exec	egy új shell futtatása
fg	előtérbeli futtatás
jobs	aktív folyamatok megtekintése
kill	futó folyamatok terminálása
newgrp	váltás egy új csoportra
shift	pozicionális paraméterek léptetése
stop	egy háttérbeli folyamat felfüggesztése
suspend	egy előtérbeli folyamat felfüggesztése
time	parancs mérése
umask	fájl-jogosultságok beállítása vagy megjelenítése
unset	változó- vagy függvénydefiníciók törlése
wait	várakozás egy háttérbeli folyamat befejezésére

H. Shell-függő parancsok

Az alábbiakban összefoglaljuk a standard shell (sh), a Bourne Again SHell (bash), a Korn shell (ksh) és a C shell (csh) fő különbségeit:

sh	bash	ksh	csh	Jelentés, művelet
$	$	$	%	alapértelmezett felhasználói prompt
	>|	>|	>!	erőltetett átirányítás
>fajl 2>&1	&>fajl vagy fajl2>&1	>fajl 2>&1	>&fajl	alapértelmezett kimenet átirányítása a fajl-ba
	{ }		{ }	elemek kibővítése a listában
`parancs`	`parancs` vagy $(parancs)	$(parancs)	`parancs`	kimenet helyettesítése a megadott paranccsal
$HOME	$HOME	$HOME	$home	a home könyvtár
	~	~	~	a home könyvtár szimbóluma
	~+, ~-, dirs	~+, ~-	=-, =N	könyvtárhalmaz elérése
val=ertek	VAL=ertek	val=ertek	set val=ertek	változó megadása
export val	export VAL=ertek	export val=ertek	setenv val ertek	környezeti változó beállítása
	${nnnn}	${nn}		9-nél több argumentumra hivatkozhatunk
"$@"	"$@"	"$@"		minden argumentum külön szó
$#	$#	$#	$#arg	argumentumok száma
$?	$?	$?	$status	kilépés a legutóbb futtatott parancs állapotából
$!	$!	$!		a legújabb háttérben futó folyamatok azonosítója
$-	$-	$-		aktuális opciók

sh	bash	ksh	csh	Jelentés, művelet
.fajl	source fajl vagy .fajl	.fajl	source fajl	parancsok olvasása a fájlban
	alias x='y'	alias x=y	alias x y	az x név lép az y parancs helyére
case	case	case	swith vagy case	választási alternatívák
done	done	done	end	ciklikus utasítás befejezése
esac	esac	esac	endsw	case vagy swith befejezése
exit n	exit n	exit n	exit (kif)	kilépés egy állapottal
for/do	for/do	for/do	foreach	ciklus változókkal
	set –f, set –o nullglob | dotglob | nocaseglob | noglob		noglob	karakterhelyettesítés elvetése fájlnév generálásánál
hash	hash	alias –t	hashstat	követett aliaszok
hash parancsok	hash parancsok	alias –t parancsok	rehash	parancsok helyének „megjegyzése"
hash –r	hash –r		unhash	parancsok helyének „elfelejtése"
	history	history	history	előző parancsok kiírása
	felfelé nyíl + Enter vagy !!	r	!!	előző parancs megismétlése
	!string	r string	! string	a string-gel kezdődő utolsó parancs megismétlése
	!par:s/x/y/	r x=y par	!par:s/x/y/	helyettesíti x-et y-nal a legutóbbi par-ral kezdődő parancsban, majd futtatja
if[$i -eq 5]	if[$i -eq 5]	if((i==5))	if((i==5))	egyszerű feltételvizsgálat

sh	bash	ksh	csh	Jelentés, művelet
fi	fi	fi	endif	if utasítás vége
ulimit	ulimit	ulimit	limit	erőforrás-korlátozás beállítása
pwd	pwd	pwd	dirs	munkakönyvtár kiírása
read	read	read	$<	olvasás terminálról
trap 2	trap 2	trap 2	onintr	megszakítások elvetése
	unalias	unalias	unalias	aliaszok eltávolítása
until	until	until		until ciklus kezdete
while/do	while/do	while/do	while	while ciklus kezdete

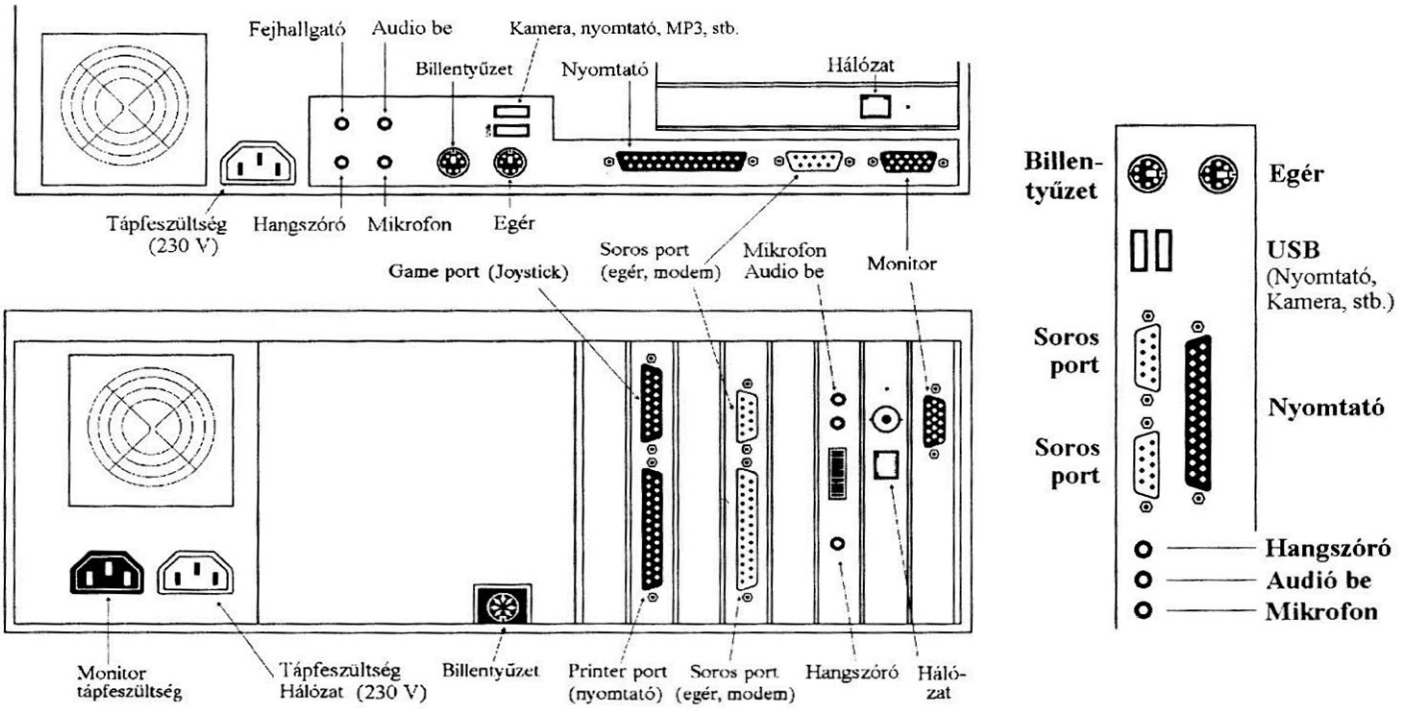
Mivel az sh a bash részhalmaza, az összes sh parancs a bash alatt is működik (fordítva értelemszerűen nem).
A csh részhalmaza a Turbo C shell-nek, ezért a csh parancsok a tcsh alatt is működnek (fordítva természetesen nem).
A táblázat nem tartalmazza a bash összes lehetőségét, pusztán arra világít rá, hogyan lehet a shell-eket felváltva felhasználni a hatékonyabb munka érdekében.

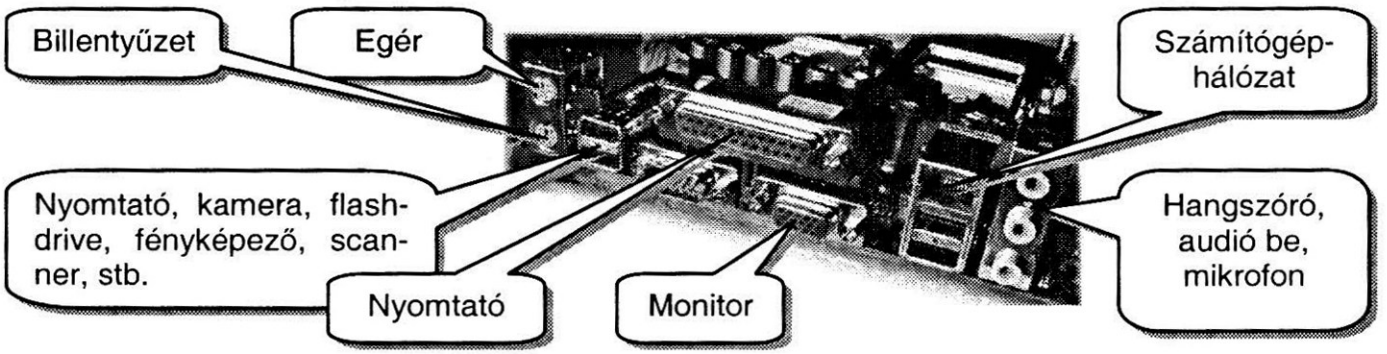
I. Nem linuxos állományok tipikus kitejesztései*

ARC	Tömörített állomány	LZH	Tömörített állomány
ARJ	Tömörített állomány	MID	Midi sorrendvezérlő zenefájl
ASC	Szöveges állomány	MP3	Zene
BAK	Módosítás előtti verzió	OLD	Módosítás előtti verzió
BMP	Képfájl	PCX	Képfájl
CDR	CorelDraw képfájl	PIC	Képfájl
DBF	Adatbázis állomány	SYS	Rendszerprogram
DOC	Szöveges állomány (Word)	TXT	Szöveges állomány
GIF	Képfájl	WAV	Hangfájl
HTM	Weboldal	WRI	Szöveges állomány (Write)
IMG	Képfájl	XLS	Excel táblázat
JPG	Képfájl	ZIP	Tömörített állomány

* Forrás: Bártfai Barnabás: A számítógép használata és a fájlkezelés c. könyv – BBS-INFO Kiadó.
A közlés a szerző hozzájárulásával történt.

J. A számítógép hátoldali csatlakozói*





K. A billentyűk általános funkciói*

Enter

Az ENTER a számítógép egyik legfontosabb billentyűje, sorváltásra, illetve a begépelt sor érvényesítésére szolgál. Akkor kell megnyomni, ha egy általunk begépelt parancsot érvényesíteni szeretnénk. Amíg csak gépelünk semmi érdemleges sem történik, hiszen a gép nem tudja, hogy mikor szándékozzuk befejezni a sor begépelését. Némely gépen az ENTER helyett RETURN vagy CR gombokat találunk, ez azonban ne zavarjon meg senkit, hiszen szerepük teljesen megegyezik az ENTER-rel. (Amennyiben a gép billentyűzetén több ENTER-t is találunk, bármelyiket használhatjuk.)

Shift

A SHIFT billentyű folyamatos nyomvatartása mellett a többi billentyű funkcióját tudjuk megváltoztatni. (Jelölése üres felfelé mutató nyíl is lehet.) Használatával a megfelelő billentyű nagybetűvel történő megjelenítése válik lehetővé. Alkalmazása szükséges akkor is, ha olyan írásjelet szeretnénk használni, amely a billentyűn másodlagosan (felül) helyezkedik el. Mivel a SHIFT gomb egyszeri lenyomása teljesen értelmetlen, ezért az ajánlott működtetési sorrend az, hogy először nyomjuk le a SHIFT billentyűt, tartsuk folyamatosan lenyomva, majd üssük le a kívánt másik billentyűt. A SHIFT billentyűt csak ennek befejeztével engedjük el.

A Caps Lock billentyű segítségével a kis- és nagybetűk használatát fordíthatjuk meg olymódon, hogy ha a Caps Lock billentyűt bekapcsoljuk, akkor a billentyűzeten leütött abc betűi nagybetűvel jelennek meg. Az ilyenkor használt SHIFT billentyű most a kisbetűk megjelenését eredményezi. (Capital letters lock = nagybetű zár.) Lenyomása nem teljesen egyenértékű a SHIFT billentyű állandó nyomvatartásával, a Caps Lock billentyű nem hatásos a számokra és az írásjelekre, ezeknél továbbra is az eredeti működés szerint kell eljárnunk. A Caps Lock billentyű bekapcsolt állapotát egy, a billentyűzet jobb felső sarkában elhelyezkedő világító dióda (LED) jelzi. (A LED mellett található felirattal azonosíthatjuk a hozzá tartozó billentyűket.) A funkció kikapcsolását szintén ugyanezzel a gombbal érhetjük el.

A tabulátor billentyű szövegszerkesztéskor, a pontosan egymás alatt elhelyezkedő oszlopok létrehozásában lehet segítségünkre. Ha megnyomjuk ezt a billentyűt, akkor a kurzort egy előre definiált oszlopba tudjuk mozgatni. Használható a mezők közti lépkedésre is. (jelölése oda-vissza mutató nyíl is lehet.)

A Ctrl (ejtsd: kontrol) megváltoztatja a párhuzamosan lenyomott gomb jelentését. A billentyűt vezérlési célokra használhatjuk. Működtetése a SHIFT billentyűhöz hasonlóan folyamatos nyomvatartással érhető el. Bizonyos programokban a Ctrl és egy megadott gomb egyidejű lenyomásával érhetünk el kívánt funkciókat.

Az Alt gomb működtetése a SHIFT és Ctrl billentyűkhöz hasonlóan szintén folyamatos nyomvatartással érhető el. Megváltoztatja a párhuzamosan lenyomott gomb jelentését. Szerepe speciális, programok által definiált, vagy alternatív funkciók elérése. Használatával oldható meg a tetszőleges kódú karakter bevitele is.

Az escape (ejtsd: iszkép) billentyű szerepe igen sokrétű, általában valamilyen visszalépést, kilépést vagy törlést vált ki. Szó szerint fordítva jelentése: menekülés. Elsősorban menüből, funkcióból való kilépésre szolgál, de a begépelt parancssor törlésére is használható.

A funkcióbillentyűknek (F1 ... F12) nincs előre meghatározott szerepe. Mindig az éppen futó program dönti el, hogy ezen billentyűket milyen célra lehet használni, de gyakran alkalmazzák az F1-et segítségkérésre. Leírásukat célszerű a programokhoz mellékelt használati útmutatóban keresni. Általában különböző menüpontok lehívására, speciális funkciók kiváltására szolgálnak.

Az ENTER billentyű felett találjuk a BACKSPACE (ejtsd: bekszpész) gombot, amely az utoljára beütött betű javítására szolgál. Megnyomásának hatására a kurzortól balra eső karakter törlődik, az esetlegesen jobbra lévők pedig balra gördülnek.

A kurzor elmozdítása egy karakterpozícióval balra. Ha a kurzor közvetlenül a prompt után áll, akkor a billentyű hatástalan.

A kurzor elmozdítása egy karakterpozícióval jobbra. Ha a kurzor a sor végén áll, akkor a billentyű hatástalan.

A kurzor elmozdítása egy sorral felfelé.

A kurzor elmozdítása egy sorral lefelé.

A kurzor sor elejére állítása. Amennyiben ezt, vagy a többi kurzormozgató billentyűt a jobboldali numerikus billentyűzetmezőn kívánjuk használni, a Num Lock gombot kikapcsolt állapotba kell hozni.

A kurzor a sor utolsó értelmezhető karakterére állítása.

A kurzor lap tetejére állítása, illetve lapozás egy oldallal visszafelé (alapállapotban nem értelmezhető).

A kurzor lap aljára állítása, illetve lapozás egy oldallal lefelé (alapállapotban nem értelmezhető).

A kurzor által jelölt (kurzor mögötti) karakter törlése. Amennyiben a kurzorral a sor közepén állunk, hatására törli a kurzor által jelölt karaktert, s helyére jobbról egy másik karakter gördül a sor hátralévő részével együtt. Ezzel a törléssel a kurzor egy helyben marad, s többszöri megnyomására folyamatosan tudjuk törölni a sor kurzortól jobbra eső részét. (A másik törlő billentyűvel, a backspace-szel a kurzortól balra lévő karaktereket törölhetjük a kurzor folyamatos mozgatása mellett.) A Del billentyű azonos a Delete feliratúval.

Váltás beszúrás és felülírás üzemmód között. Ha a kurzorral a sor közepén egy tetszőleges karakteren állunk és felülírás üzemmódban nyomunk meg egy betűt, akkor az törli a helyén lévőt, míg beszúrás üzemmódban az általunk leütött betű helyet szorít magának a sor jobbra tolásával, s nem törli ki a kurzor által jelölt karaktert.

A jobboldali numerikus billentyűzetmezőn elhelyezkedő gombok elsősorban a számok beírására szolgálnak. Ám, ha jobban megnézzük, a számok mellett kis nyilakat is találunk, amelyeket szintén használhatunk. E két üzemmód (számok és nyilak) váltására szolgál a Num Lock billentyű. Ehhez a gombhoz is tartozik egy világító dióda, melyen ellenőrizhetjük állapotát. A Num Lock gomb megnyomásával tudjuk a megfelelő üzemmódot (és a lámpát) kapcsolni (működése a Caps Lock-hoz hasonló). Ha a led világít, akkor a numerikus billentyűzetmezőn elhelyezkedő gombok számok bevitelére szolgálnak, ellenkező esetben a kurzor mozgatására van módunk, a nyilaknak megfelelően.

A Scroll Lock billentyű be- illetve kikapcsolt állapotát szintén egy dióda jelzi. Bekapcsolt állapotában a nyilak nem kurzormozgatásra, hanem objektummozgatásra szolgálnak. (Némely program bizonyos képernyőterületek mozgatását is lehetővé teszi, ilyenkor lehet alkalmazni a Scroll Lock billentyűt. Egyes szövegszerkesztő programokban a Scroll Lock kikapcsolt állapotában a kurzort mozgathatjuk a nyilakkal, bekapcsolt állapotában a szöveget, a kurzor egy helyben tartása mellett.)

A PAUSE billentyű a magnetofonokhoz hasonlóan itt is a ,,pillanat állj`` funkciót látja el. (pl. olyankor célszerű használni, amikor a képernyőn szaladó hosszú lista megállítására van szükségünk.)

A képernyő tartalmának vágólapra vagy nyomtatóra másolása.

A Start menü előhívása.

A helyi menü előhívása.

L. Az egér kezelése*

Az egeret úgy kell kézbe venni, hogy annak kábele a tőlünk távolabbi oldalán, a mutatóujjunk a baloldali gombon legyen. Még mielőtt bármihez is kezdenénk, barátkozzunk meg vele. Figyeljük meg, hogyan mozdul el a képernyőn lévő egérkurzor az egér mozgatásával. Lényegében nem történik más, mint amerre elmozdítjuk az egeret az asztalon, úgy mozdul el az egérkurzorunk is. Amíg ez magabiztosan nem megy, ne lépjünk tovább.

Kattintás

A kattintás nem más, mint az egér baloldali gombjának egyszeri rövid megnyomása az egér elmozdítása nélkül. Ezt a funkciót a kiválasztásoknál gyakran alkalmazzuk. A kattintás mindig ott történik, ahová a nyíl hegye mutat.

Dupla kattintás

A dupla kattintás az előzőhöz hasonlóan az egér baloldali gombjának kétszeri rövid gyors egymásutánban történő megnyomása az egér elmozdítása nélkül. Ezt a funkciót nem árt gyakorolni, ugyanis másnak érzékelheti a gép, ha a kattintások között hosszabb idő telik el, vagy elmozdítanánk egerünket.

* Forrás: Bártfai Barnabás: A számítógép használata és a fájlkezelés c. könyv – BBS-INFO Kiadó. A közlés a szerző hozzájárulásával történt.

Húzás (vonszolás), meszelés

A húzás funkciót az egérrel úgy tudjuk megvalósítani, hogy az egeret a baloldali gombjának nyomvatartása mellett mozgatjuk. Gyakorlatban ez azt jelenti, hogy a mozgatni kívánt objektumon az egér baloldali gombját megnyomjuk, de nem engedjük fel. A húzáshoz az egeret a gomb nyomvatartása mellett mozdítjuk el, majd a kívánt célhelyen elengedjük annak gombját. (A szöveg feletti húzás kijelölést eredményez – meszelés funkció.)

M. Ablakműveletek*

Ablak bezárása

Az ablak bezárása, azaz a programból való kilépés legegyszerűbben az alkalmazás ablakának jobb felső sarkában lévő ✖ bezárógombon való kattintással valósítható meg. A programok többségénél lehetőség van a File menüben található Kilépés menüponttal is bezárni az alkalmazást.

Méretváltoztatás

Az ablak méretének megváltoztatására nem minden ablak esetén van lehetőségünk. Méretváltoztatáshoz az ablakszélen állva kell a húzás műveletet alkalmaznunk. Amennyiben a nyilat az ablakszél részre mozdítjuk, változtatható méretű ablakok esetén annak formája megváltozik. Ha ekkor folyamatosan nyomva tartjuk az egér bal oldali gombját, úgy az egér mozgatásával egyszerre az ablak szélének helyzetét tudjuk beállítani. Az egér gombját a végleges pozícióban kell elengedni.
További lehetőség, amikor az ablakot teljes képernyő méretűre változtatjuk a ◘ vagy ▣ teljesméret gomb segítségével, illetve minimalizáljuk a ▪ vagy ▂ gomb segítségével. Az ablak korábbi méretének visszaállításához az ◘ vagy ❐ előzőméret gombon kell kattintanunk, illetve ha az ablakot minimalizáltuk, úgy a panelen lévő gombjával visszahozhatjuk.
Segítségkérésre az i gomb használható

Gördítősáv

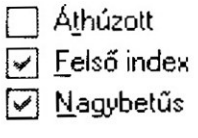
Ez a jel akkor válik láthatóvá, ha az ablak tartalma nagyobb, mint amennyi abban láthatóan elférne. A gördítőgombot húzva tudjuk kijelölni azt, hogy az ablak tartalmának mely része legyen látható.

Nyomógombok

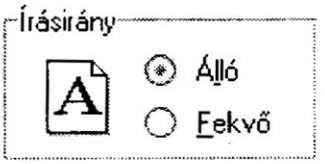
E nyomógombhoz hasonló területeken történő kattintással az adott gomb funkcióját aktivizálhatjuk. A nyomógombok között találjuk például az **OK** gombot, mellyel az adott párbeszédablak érvényesítését, illetve az ott bevitt adatok elfogadását valósíthatjuk meg. A Mégsem gomb az adott párbeszédablak tevékenység nélküli visszavonását váltja ki.

Kiválasztónégyzetek

A kiválasztónégyzetek egymástól független különböző szolgáltatások be-, illetve kikapcsolására szolgálnak. Amennyiben egy négyzet üres, úgy a négyzetben történő kattintás a négyzet bejelölését (beikszelését), bejelölt négyzet esetén pedig annak törlését jelenti. (A ☑ helyett bizonyos programok ☒-t helyeznek, melynek jelentése ugyanaz.)

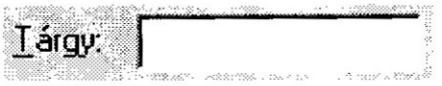


Választókapcsolók



A választókapcsolók, vagy más néven rádiógombok olyan kis kör alakú gombocskák, melyek közül, mindig csak egyet aktivizálhatunk, azok egymástól függetlenül történő kapcsolása értelmetlen lenne. A kiválasztás egyszerűen az adott választókapcsolón történő kattintással valósítható meg.

Kitöltőmezők

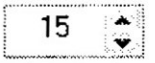


A kitöltőmezők olyan adatbeviteli területek, melyeket a párbeszédablakokban alkalmaznak szöveges vagy numerikus adatok megadására (pl. állománynév bevitelnél). Ha a kitöltendő mező már tartalmaz adatot, úgy a mezőre való rálépéskor annak tartalma gyakran inverzre vált. Ez esetben karakter leütése után a korábbi tartalom eltűnik s a bevitt karakterrel kezdve új adatot vihetünk be. Amennyiben nem karaktert gépelünk, hanem kurzort mozgatunk, úgy lehetőség van a korábbi tartalom módosítására.

Listakeretek, legördülő lista

A listakeret olyan terület, amelybe tetszőleges adatot nem írhatunk, csupán a gép által felajánlott listából választhatunk. Ha a listára nem férne rá az összes kiválasztható adat, úgy jobboldalt egy gördítősáv is megjelenik. A listáról elemet választani a megfelelő elemen történő kattintással tudunk. A legördülő lista nagymértékben hasonlít a listakerethez, de itt a gép alapállapotban csak egy elemet jelenít meg, amelynek esetleges megváltoztatásához a jobb oldalán elhelyezkedő legördítő nyíllal a listát külön kell legördíteni.

Numerikus adatok bevitele



Sok esetben numerikus adatot nemcsak a kitöltőmező használatával tudunk begépelni. Ha a beviteli mező mellett kis le- és felfelé mutató nyilat találunk, használhatjuk ezeket is olymódon, hogy a lefelé mutató nyíllal csökkenthetjük, míg a felfelé mutató nyíllal növelhetjük a beviteli mezőben található értéket.

* Forrás: Bártfai Barnabás: A számítógép használata és a fájlkezelés c. könyv – BBS-INFO Kiadó. A közlés a szerző hozzájárulásával történt.

TÁRGYMUTATÓ

KISSZÓTÁR

ablakozó: a Linux grafikus felületét vezérlő program. Más néven ablakmenedzser. Pl. KDE, Gnome, WindowMaker, fvwm

alap-Linux: ha a Linux-ot első alkalommal telepítjük, az ~ kerül fel először a gépre, ami a merevlemez nélkül is tud működni, hiszen ekkor az még nem elérhető. Kernele egy floppy-n vagy egy CD-ROM-on van

aliasz: a →shell-lel kapcsolatban gyakran használt kifejezés. Az ~ egy hivatkozás a hosszú nevű vagy gyakran használt parancsokra (kevesebbet kell gépelnünk). Jelentése: álnév.

állomány: Összetartozó adatok egy nagyobb egysége.

azonosító: felhasználói név-jelszó páros. A felhasználó ~-ját a rendszergazda hozza létre, aki egy vagy több csoportot, valamint jogosultságokat társít a felhasználóhoz. Egy felhasználói ~ létrehozása normál esetben egy home könyvtár meghatározását és e-mail-kézbesítést foglal magába.

becsatolási pont: a fájlrendszer azon része, ahova egy partíciót vagy valamely eszköz fájlrendszerét becsatoljuk

BIOS: alapvető bemeneti-kimeneti rendszer. Minden PC szerves része ez a kis különleges memóriaterület (külön chipben), ami az alapvető műveleteket (memóriaellenőrzés, merevlemez-felismerés stb.) végző programokat tartalmazza. A Linux kernele a ~-nál jóval hatékonyabb lehetőségeket kínál.

biztonsági mentés: ~-eket gyakran kell készíteni, különösen a fontos fájlokról, illetve a hosszú órák alatt beállított konfigurációs fájlokról. A legnépszerűbb archíválást a tar kínálja, melyet gyakran használunk a gzip-pel kombinálva.

démon: a kifejezés angol eredetije, a daemon a dist and execution monitor rövidítése. Ez egy olyan program, ami a háttérben figyel és ha szükséges, működésbe lép. Ilyen ~-ok válaszolnak az FTP- vagy HTTP-kérésekre, felügyelik a PCMCIA aktivitását stb.

ELF: executable and linking format. A Linux szabványos bináris fájlformátuma.

eszköz: a Linux eszközök a fájlrendszer különleges részei, melyek a /dev könyvtárban szoktak lenni. Az ~-ök száma alapján tudja a kernel elérni őket. Teljesen más értelemben jelent programot, alkalmazást is.

Ethernet: széles körben használt hálózati hardver LAN-okhoz busz topológiával. Eredetileg 10 Mbit/s-ot tudott koaxális kábelen, napjaink csavart érpárja már 100 Mbit/s fölötti sebességet is elér csillag topológiában.

EXT2: a Linux második generációs bővített fájlrendszere. Támogatja a hosszú fájlneveket, a jogosultságokat, a hibatűrést.

fájl: ld. állomány

folyamat: a Linux alatt nem egyszerűen programokat futtatunk, hanem ~-okat. A ~ egy program vagy egy futtatható fájl aktív változata. Egyes programok nem

indítanak el folyamatot, mások többet is elindítanak. Munkavégzésünket igen finoman szabályozhatjuk, hiszen a ~-ok nemcsak megállíthatók, de megszakíthatók, felfüggeszthetők, újraindíthatók, probléma esetén kilőhetők. Egyidejűleg értelemszerűen több folyamat is futhat. Azonosítóikkal egyértelműen beazonosíthatók. A ~ alternatív kifejezése a taszk, a munka és a feladat.

FTP: file transfer protocol, azaz fájlátviteli protokoll. Adatok átvitele egyik gépről a másikra hálózaton keresztül. Az egyik oldalon az ~-szerver áll (ez küldi a fájlokat), a másikon az ~-kliens (ami fogadja a fájlokat).

GNU: a GNU is Not Linux (a GNU nem Linux) rövidítése, ami a Free Software Foundation (FSF, Ingyenes Szoftver Alapítvány) terméke. Célja egy teljesen ingyenes UNIX-kompatibilis operációs rendszer közzététele.

Graphical User Interface: grafikus felhasználói interfész, rövidítve GUI. Egy valódi asztal reprezentációja. Míg azonban az asztalon papírokat tologatunk, addig a grafikus felületen ablakokat. Minden önálló folyamat külön ablakban fut. A ~ gyakori vezérlőeszköze az egér. Linux környezetben a ~ az X Window rendszer felülete.

háttérbeli folyamat: ha →shell-ünk csak egyetlen →folyamattal foglalkozik, azt előtérbeli folyamatnak nevezzük. A legtöbb →shell lehetővé teszi, hogy a folyamatokat a háttérben is futtathassuk. Ehhez egy &-et kell tennünk a →folyamat végére.

home könyvtár: a legtöbb felhasználó kiindulópontja egy gépen. Általában a privát fájljainkat is itt tároljuk. A rendszergazda mellett mi vagyunk az egyetlenek, akik hozzáférhetünk ehhez a könyvtárhoz. A ~ helyét a HOME környezeti változó tárolja.

HTTP: HyperText Transfer Protocol. Az internet egyik leggyakoribb protokollja. A webes dokumentumok nagy része is ezt a protokollt használja.

inode: ~-ok segítségével szervezi az →ext2 fájlrendszer a fájlinformációkat. Az ~ tartalmazza a fájl tulajdonosát, jogosultságokat, módosítás idejét stb.

interfész: általánosságban egy eszköz a különféle rendszerek információcseréjéhez. Három típusa van: hardver~ (párhuzamos, soros vagy SCSI), szoftver~ (a program kommunikációját határozza meg), felhasználói ~ (ember és gép közötti kommunikáció: billentyűzet, egér, monitor).

internet: az egész világra kiterjedő számítógép-hálózat. Korábban nagy kezdőbetűvel írták, így elkülönítve az útválasztókkal összekötött fizikai hálózatoktól.

internet-cím: az internetes erőforrás címe. Első része a protokoll neve, ezt követi a szervezet/kiszolgáló neve, majd az ország-azonosító (Észak-Amerikában a szervezet-típusazonosító).

IP: Internet Protocol. A →TCP/IP üzenetkezelőnek az IP-csomagok címzéséért, valamint a hálózaton át történő továbbításáért felelős protokollja. Nem garantálja a csomagok célba érkezését, valamint a sorrend-helyességet sem.

IP-cím: egy 32 bites cím, mely egy hálózatazonosító és egy állomásazonosító részből áll. Gyakori a pontokkal határolt, tízes számrendszerbeli megadása, például 192.168.8.25

kiterjesztés: Az állományazonosító azon utolsó (általában három betűs) része, amely a fájl típusára, szerkezetére utal.

konzol: egykor a terminál szinonímája volt. A Linux-ban virtuális ~-ok vannak. Ennek köszönhető, hogy a különféle futó folyamatoknak saját képernyője lehet. Az alapértelmezett 2-es futási szinten 6 virtuális ~-unk van, melyeket az Alt+F1 – Alt+F6 billentyűkombinációval érhetünk el. Ha X Window rendszert futtatunk, a Ctrl+Alt+F1 – Ctrl+Alt+F6 billentyűkombinációt használjuk.

környezet: a →shell többféle ~-et is kínál, ahol ideiglenesen különféle opciókat állíthatunk be, például programok

elérési útvonalát, a felhasználói nevet, az aktuális útvonalat, a prompt megjelenését stb. Ez az adat a →környezeti változóban kerül tárolásra.

környezeti változó: egy adatmentési hely a shell környezetében. Minden változó egy névből (általában nagybetűkkel írva) és egy értékből (pl. elérési útvonal) áll.

MBR: master boot record. Az első fizikai szektor (0. cilinder, 0.fej, 1. szektor) az első merevlemezen. Minden merevlemez tartalmaz ~-t. Tartalma az operációs rendszer, illetve egy rendszerindító elindulásához elengedhetetlen.

operációs rendszer: Olyan program, amely felügyeli és vezérli a számítógép működését.

parancssor: UNIX-alapú rendszereken a shell-ben való munka parancssor-orientált. Minden folyamat, amit a shell-ben indítunk, saját sorral rendelkezhet.

PID: process ID, azaz folyamatazonosító

puffer: egyfajta közbülső tároló, ami a különböző sebességgel működő médiák és eszközök közötti adatokhoz való hozzáférést gyorsítja

rendszerbetöltés: a számítógép indítása és az operációs rendszer rendelkezésre állása közötti folyamat. Linux-ban ez magában foglalja a kernel betöltését, ekkor a képernyőn az uncompressing linux... üzenet jelenik meg, végül megkapjuk a bejelentkezési login: promptot.

shell: alapvető kapcsolat a programok és az operációs rendszer között. Segítségével parancsokat adhatunk a rendszernek. A feladatokat nemcsak interaktív módon hajtathatjuk végre, hanem úgynevezett ~ scriptekkel automatizálva is.

süti: egy olyan mechanizmus, mellyel a kliens kisebb információkat tárolhat a kliens gépen. Az angol cookie terminológiával is gyakran találkozunk.

szerver: olyan számítógép, amely biztosítja a hálózathoz és annak erőforrásaihoz való hozzáférést, esetleg szolgáltatásokat nyújt más számítógépeknek (kiszolgáló)

TCP/IP: Transmission Control Protocol/Internet Protocol. Az internet hálózati protokollja. Két része van: a TCP a szállítási protokoll, az →IP a nagyszámú hálózat egymásba integrálását oldja meg.

URL: Uniform Resource Locator. Az internetes erőforrások egységes leírásának jelölésrendszere. Szintaktikája:

protokoll://internet_cim:port/konyvtar/fajl?parameter

melyből egyszerre nem szoktunk mindent használni, a port és a paraméter gyakran elmarad, és sokszor még a könyvtárat vagy a fájl sem kell megadni.

World Wide Web Consortium: a WWW ajánlásait készítő, publikáló és azt folyamatosan frissítő, fejlesztő konzorcium. Honlapjuk a www.w3.org címen érhető el.

WWW: World Wide Web, azaz világméretű pókháló. Az internet egyik nagyon gyakran használt része, elsősorban HTML dokumentumokat és grafikus internetes, tehát webes felületeket foglal magába.

X szerver: csak olyan gépek használhatnak grafikus felhasználói felületet, melyeken fut az ~. Egyik fontos feladata a képernyők kezelése. Egy terminál legtöbbször egy képernyővel rendelkezik.

X Window: X ablakozórendszer. Egy grafikus felhasználói felületet kezelő programok, rutinok, protokollok gyűjteménye. Rövidítve: X

A BBS-INFO KIADÓ KÖNYVAJÁNLATA

Bártfai Barnabás: ## Hogyan használjam?

560 oldal, B/5, 2190 Ft.

E közérthető nyelvezetű, kezdők számára is készült könyv minden fontosabb témára kiterjed, így ajánlható önálló tanuláshoz ECDL és OKJ vizsgákhoz, tanfolyamokhoz, de mivel a középiskolai kerettanterv anyagát is magába foglalja, iskolák részére is.

A könyv főbb témakörei: Alapismeretek • Gépkezelés • Szoftverek csoportosítása • Szoftverüzemeltetői feladatok • Programozás-elmélet • A DOS kezelése és parancsainak ismertetése • Windows (95, 98, NT, 2000, XP, Windows alkalmazások, beállítások) • Állománytömörítés • Vírusok • Setup • Számítógép-hálózatok • Internet • Weblapok készítése • Elektronikus levelezés • Outlook • Adatvédelem, törvény • Norton Commander • Szövegszerkesztés (EDIT, Wordpad, Word 6-7-97-2000-XP) • Táblázatkezelés (Excel 5-7-97-2000-XP) • Adatbázis-kezelés (Access) • Grafika (Paint, MS Photo Editor, CorelDraw) • Prezentáció készítés (PowerPoint) • Kifejezésgyűjtemény • Kódtáblázatok • Ellenőrző kérdések • Stb.

Bártfai Barnabás: ## Hogyan kezdjem?

240 oldal, B/5, 1290 Ft.

A könyv a Hogyan használjam? c. kiadvány szűkített változata.

Témakörei: Alapismeretek • Gépkezelés • Szoftverek • DOS • Windows (95, 98, NT) • Tömörítés • Vírusok • Setup • Hálózatok • Internet • E-mail • NC • Word • Excel • Paint • PowerPoint • Kifejezésgyűjtemény • Kódtáblázatok • Ellenőrző kérdések • Stb.

Informatikai teszt- és feladatgyűjtemény

352 oldal B/5, 1970 Ft

ECDL és OKJ vizsgákhoz, önálló tanuláshoz, gyakorláshoz. A könyvben szereplő több mint ezer feladat kellő gyakorlási lehetőséget biztosít arra, hogy elsajátíthassuk a számítógép használatának tudnivalóit, a majd kétezer teszt pedig segít az elméleti tudásunk ellenőrzésében. (Alapismeretek, DOS, Windows, Elektronikus kommunikáció, Tömörítés, Vírusok, Word, Excel, Access, Grafika, Prezentációkészítés, Programozás, stb.)

Hogyan használjam? CD

Interaktív számítástechnikai oktatóprogram - 4930 Ft.

Igen egyszerűen kezelhető, díjnyertes oktatóprogram az alapoktól a különböző programokig. Számos videó és oktatóprogram, ezer oldal szöveges anyag, ábrák, kérdések és válaszok, több ezer teszt, feladatlap készítés, stb. ECDL vizsgákhoz és önállóan tanuláshoz is ajánlott.

Bártfai Barnabás: ## Weblapkészítés házilag

200 oldal, B/5, 1760 Ft.

A könyv segítségével megismerhetjük a weblapkészítés alapjait, az elterjedt weblapkészítő programok kezelését, illetve a hozzá kapcsolódó egyéb feladatokat. (Internet, regisztrálás, feltöltés, képek, HTML, Netscape composer, Word, MS Frontpage, MS Image Composer, MS GIF Animator, Macromedia Dreamweaver, ötletek, mintaweblapok, stb.)

Sikos László: ## JavaScript kliens oldalon

188 oldal, B/5, 2900 Ft.

A példákat és részletes magyarázatokat is tartalmazó könyv segítségével a Webes programozáshoz nélkülözhetetlen JavaScript nyelvet ismerhetjük meg.

Sikos László: ## Szerver oldali webprogramozás

192 oldal, B/5, 2900 Ft.

A példákat és részletes magyarázatokat is tartalmazó könyv segítségével a Webes programozás szerver oldali lehetőségeit ismerhetjük meg.

Sikos László: ## XHTM

206 oldal, B/5, 2900 Ft.

A HTML megújulása XML alapokon.

Bártfai B.: ## Hálózatépítés otthonra és kisirodába

176 oldal, B/5, 1870 Ft.

A könyv megismerteti olvasóját a kisebb hálózatok építéséhez szükséges tudnivalókkal, segítséget nyújt a tervezéshez, kivitelezéshez, az eszközök kiválasztásához és konfigurálásához, de kitér az új technikákra is.

Tóth Tamás: ## Szövegszerkesztési feladatgyűjtemény

144 oldal, A/4, 1970 Ft. CD melléklettel

Ennek a százhúsz gyakorlati feladatot tartalmazó könyvnek a segítségével lehetőség van a szövegszerkesztő programok gyakorlására. Minden példához ötletek és elkészítési tanácsok tartoznak. A könyv egyaránt ajánlható önállóan tanulóknak, ECDL és OKJ vizsgára készülőknek, valamint tanároknak.

Bártfai B.: ## Táblázatkezelési gyakorlatok

176 oldal, B/5, 1760 Ft. CD melléklettel

A könyvben Excel példákat és részletes megoldási leírásokat találunk teljesen alapszinttől egészen a vizsgafeladatokig. A könyv egyaránt ajánlható önállóan tanulóknak, ECDL és OKJ vizsgára készülőknek, valamint tanároknak.

Informatikai füzetek

A könyvek, az ECDL vizsgákhoz, a Számítógép-kezelő és a Szoftverüzemeltető OKJ-s képzésekhez, illetve középiskolai és Gimnáziumi Informatika órákhoz ajánlható. Tartalma a Hogyan használjam c. könyvre épül.

- **Az Információ technológia fogalmai** *144 oldal, 970 Ft.*
- **Számítógép használat és fájlkezelés** *216 oldal, 1190 Ft.*
- **Szövegszerkesztés** *124 oldal, 970 Ft.*
- **Táblázatkezelés** *88 oldal, 870 Ft.*
- **Adatbázis-kezelés** *136 oldal, 970 Ft.*
- **Prezentáció és grafika** *100 oldal, 870 Ft.*
- **Információ és kommunikáció** *88 oldal, 870 Ft.*
- **Programozás** *296 oldal, 1870 Ft.*

Sikos László:

Digitális fényképezés a mindennapokban

192 oldal B/5, színes melléklettel, 1970 Ft.

A könyv közérthető módon mutatja be a digitális képfeldolgozás rendszerét, lehetőségeit, módszereit és eszközeit. Hasznos tanácsokat kapunk, mind a gép megvásárlásához, mind annak használatához.

Bártfai Barnabás: ## Office XP *352 oldal B/5, 1870 Ft.*

A könyv az Office XP programcsomag használatát mutatja. Főbb témakörök: Általános tudnivalók (Alapfunkciók, telepítés, segéd, irányítópult) • Word • Excel • PowerPoint • Outlook • Access • További szolgáltatások • Stb.

Molnár Csaba - Sági Gábor:

Programozás Turbo Pascal nyelven

232 oldal B/5, 1190 Ft.

A könyv a programozás elméletének ismertetését követően a Turbo Pascal programozási nyelv elsajátításához nyújt hasznos segítséget rengeteg példával, feladattal és mintaprogrammal illusztrálva.

Bártfai B.: ## Works 2000 *240 old. 1430 Ft.*

Ez a kezdő felhasználók számára is könnyedén érthető könyv a Works programcsomag használat, kezelésének fortélyait ismerteti.

Zsebkönyvek:

E kisméretű könyvecskék kiválóan alkalmasak arra, hogy zsebben tartva állandóan kéznél legyen. A könyvek egy rövid alapismereti rész után a különböző programok kezeléséhez szükséges ismereteket tárgyalják.

Windows XP • Word 2002 • Excel 2002 • Access 97-2000 • CorelDraw • Outlook 97-2000 • Powerpoint 2002

Dr. Magyar Elemér: ## Kezdő bűnözők kézikönyve

216 oldal, B/6, 970 Ft.

Bár első látásra e könyv a bűnözővé válás rögös útján előforduló buktatókkal foglalkozik, mégis a sorok közt lapuló irónikus mondanivaló sokszor többet árul el...

Rozványi Dávid:

Titkos hadműveletek pénztárcánk ellen

240 oldal, 540 Ft.

A mai világban rendszeresen megpróbálnak megtéveszteni és átverni minket. A kiadvány, akár a kereskedők ötletkönyve is lehetne, mégis a vásárlókhoz szól, hiszen a különböző vevőátverési módszereket és az azok elleni védekezési lehetőségeket ismerteti, humoros stílusban.

Lipták Béla: ## 35 NAP

224 oldal, B/5, keménytáblás, 1970 Ft.

A közismert környezetvédő professzor 1956-os naplója a forradalomról, számos új dokumentummal, adattal és információval.

Szeder Z: ## Elektronikai készülékek hulladékainak kezelése

80 oldal, B/5, 790 Ft.

Avagy mi történik például a számítógépekkel, miután kidobjuk őket? Ez az új gondolkodásmódú, átfogó könyv a környezetvédelem és - hulladékkezelés általános problémáin túl az elektronikus eszközök hulladékainak kezelésével kapcsolatos lehetőségeket ismerteti.

Hétköznapi marketing sorozat

80-150 oldal, 870 Ft.

A sorozat a marketing egyes területeit a hétköznapi ember számára fordítja le és teszi hasznosíthatóvá, kiszűrve a lényeget. A sorozat kötetei:

- **Reklám és társai**
- **Személyes eladás**
- **Üzletek kialakítása**

Abaházi Viktor: ## Számítástechnika másként

64 oldal, színes, B/6, keménytáblás, 1190 Ft.

Ez az irónikus hangvételű könyv humoros írások, képek, Murphy törvények és hasonló nyalánkságok formájában veszi el a kedvünket attól, hogy számítógép közelébe kerüljünk.

Abaházi Viktor: ## Puskázási kézikönyv

128 oldal B/6, keménytáblás, 970 Ft.

E rendhagyó könyv elsősorban azokhoz szól, akik nem szeretnek tanulni. A könyv különböző puskázási módszereket, elkészítési útmutatókat, tanácsokat és lehetőségeket is ismertet.

Nagy Z: ## Amit tudni kell a raktárkezelő és számlázó programokról

128 oldal CD melléklettel, 1870 Ft.

Raktrkezelő és számlázó programot vásárlóknak nélkülözhetetlen. A programok lehetőségeink ismertetése, programok összehasonlítása, stb.

Batos Gyula: ## Intrastat kézikönyv

A könyv az Európai unió részére nyújtandó kötelező adatszolgáltatás megtételéhez nyújt hasznos segítséget.

Kivaghy Krisztián: ## A manipuláció tudománya

160 oldal B/6, 760 Ft

A könyvből megtanulhatjuk, miként lehet emberek millióit befolyásolni, igényeit ízlésünk szerint formálni, velük a legnagyobb hazugságokat elhitetni.

RÉSZLETES INFORMÁCIÓK, ÚJDONSÁGOK ÉS TARTALOMJEGYZÉKEK AZ INTERNETETEN: WWW.BBS.HU

A fenti könyvek megrendelhetők a kiadó címén: **BBS-INFO Kft.** 1630 Bp., Pf. 21. *Nagyobb mennyiségű rendelés esetén kedvezményeket tudunk biztosítani.*