



KÖNNYEN TANULHATÓ

STEFAN STROBEL - THOMAS UHL

# LINUX



KOSSUTH KÖNYVKIADÓ

LAUREL

STEFAN STROBEL–THOMAS UHL

# LINUX

KOSSUTH KÖNYVKIADÓ 1996

STEFAN STROBEL ÉS THOMAS UHL KÖNYVE  
EREDETILEG NÉMETÜL JELENT MEG „LINUX – VON PC ZUR WORKSTATION” CÍMMEL.  
A MAGYAR KIADÁS ALAPJA AZ ANGOL NYELVŰ KIADÁS  
„LINUX – UNLEASHING THE WORKSTATION IN YOUR PC”.

FORDÍTotta  
K. PAPP LÁSZLÓNÉ ÉS TARIJÁN GYÖRGY

SZAKMAILAG ELLENŐRIZTE  
NEMKIN RÓBERT

ISBN 963 09 3896 0

© SPRINGER-VERLAG NEW YORK INC. 1994., 1996. ALL RIGHTS RESERVED  
© HUNGARIAN EDITION KOSSUTH KÖNYVKIADÓ 1996

A KIADÓ ENGEDÉLYE NÉLKÜL A KÖNYV EGYETLEN RÉSZÉT SEM LEHET LEMÁSOLNI,  
SEM MILYEN FORMÁBAN, SEMMIFÉLE ESZKÖZZEL  
(ELEKTRONIKUS, MECHANIKUS, FÉNYKÉP VAGY FÉNYMÁSOLAT),  
BÁRMILYEN INFORMÁCIÓSZOLGÁLTató RENDSZERBEN TÁROLNI VAGY TERJESZTENI.

FELELŐS KIADÓ KOCSIS ANDRÁS SÁNDOR,  
A KOSSUTH KÖNYVKIADÓ ÉS KERESKEDELMI KFT. VEZÉRIGAZGATÓJA  
A KÖTETET KORONCZAI MAGDOLNA Szerkesztette  
Műszaki vezető KUN GÁBOR  
Műszaki szerkesztő PÁNYI BÉLA  
TERJEDELME 32,2 (A/5) IV  
KÉSZÜLT A KOSSUTH KÖNYVKIADÓ NYOMDÁJÁBAN  
FELELŐS Vezető KÁMÁN ANDRÁS

# BEVEZETÉS

**M**a már a 32 bites PC-k számítanak a legkorszerűbbnek és egyre inkább úgy tűnik, hogy a DOS-t hamarosan felváltják a hatékonyabb operációs rendszerek. Ez idő szerint – legalábbis a szakirodalomban – élénk vita folyik arról, hogy mi lesz a jövőbeni szabvány. A szerver operációs rendszerek számára egyre inkább két alternatíva emelkedik ki: A Windows NT és a UNIX változatok, például a Solaris 2, UnixWare és a NestStep 486. Ebből a nézőpontból az OS/2 szerepe nem jelentős, mivel ez inkább a Windows versenytársának tekinthető mind a jelenlegi, minden a jövőbeni 32 bites verziója esetén is.

Még nem lehet megjósolni, hogy melyik rendszer lesz végül az uralkodó. Azonban az elmúlt években a hardver területén végbement jelentős fejlődés, igényt támasztott olyan korszerű operációs rendszerek iránt, amelyek e fejlesztések előnyeit is kihasználják. A modern szerver operációs rendszerek esetén a klasszikus UNIX munkaállomások és a „végeken” működő PC-k közötti határon belül már sokkal rugalmasabb.

## 1.1 A LINUX TÖRTÉNELMI ELŐZMÉNYEI

A jövő lehetséges operációs rendszerei közül az egyik nagyon ígéretes a Linux, amely az Intel processzorokra készített és ingyenesen hozzáférhető UNIX rendszer.

A Linux rendszert egy fiatal finn egyetemista, Linus Torvalds fejlesztette ki, távolról sem azzal a céllal, hogy egy minden igényt kielégítő operációs rendszert hozzon létre. Először csak a 80386-os processzor speciális feladat-kapsolásos parancsait kívánta megismerni. Ezen tesztprogram forgatása során a MINIX operációs rendszert használta, amely Andrew Tannenbaum által fejlesztett, és az operációs rendszerek oktatását szolgáló rendszer.

Linus Torvalds e tesztprogramból kiindulva lépésről lépésre fejlesztett ki egy olyan operációs rendszermagot – kernelt –, amely a 80386-os processzor védett üzemmódjában futott, így optimálisan kihasználta a processzor képességeit.

A feladatváltást megvalósító programrész után Torvalds egyszerű billentyűzet-kezelőt készített, amely a rendszerrel az interaktív munkát lehetővé tette. A fejlesztésben eddig eljutva a Linux még mindig a MINIX rendszer részeire támaszkodott, azonban ez is hamarosan megváltozott.

Annak érdekében, hogy ne kelljen új fájlrendszert is fejleszteni, Torvalds úgy döntött, hogy adaptálja a MINIX fájlrendszerét. Ezzel nagymennyiségű munkát takarított meg magának. Néhány hónap múlva a fejlesztő úgy ítélte meg, hogy rendszere már elég kidolgozott ahhoz, hogy azt szélesebb nyilvánosság is megismerje.

A Linux teljes forráskódja első alkalommal 1991. augusztusában jelent meg Finnország legnagyobb **FTP szerverén**, amelynek Internet címe `ftp.funet.fi`. A programot a MINIX szabadon terjesztett klónjaknál jelentették be, amely a hálózaton mindenkihez közel állt. A rendszer gyorsan elterjedt, hiszen a rendszert többek között a University of California, Berkeley is használta. A rendszer gyorsan fejlődött, és a rendszert következő verziójában (0.02), amely már néhány alapvető UNIX parancsot is tartalmazott. A hozzá tartozó GNU fordítóprogram (`gcc`) C programok fordítását is lehetővé tette, s ezzel a UNIX shell (`bash`) átvitelét linuxra.

Torvalds korábbi döntése, amelyben a **POSIX-nek** való megfelelést tűzte ki célul azz IEEE (Institute of Electrical and Electronics Engineers) szabványainak megfelelően, alapvető szerepet játszott a szabványos UNIX szoftverek, így a jelenleg létező Linux kialakításában is.

A Linux iránt csak az év vége felé mutatkozott nagyobb érdeklődés és az áttörés 1992. január 5-én következett be a program 0.12-es verziójának megjelenésekor. A Linux ezen verziójára már a programfejlesztők sokkal szélesebb tábora figyelt fel. A rendszer ugyanis egy olyan cseremechanizmussal (swap) rendelkezett, amely azt egyértelműen a MINIX félét emelte.

Az idők során egyre több érdeklődő programfejlesztő küldött javításokat és javaslatokat Finnországba, aktívan fejlesztve a Linux-rendszert. Ezen a módon olyan korai fejlesztések valósultak meg, mint a POSIX Job Control a 0.12-es verzióban és az átkapcsolható virtuális konzolok. (Azóta a Linux az összes több más platformra is, így fut Sun/Sparc, DEC/Alpha, SGI/Mips rendszereken is).

Az Internet, a hatalmas méretű vilaghálózat, amelyre már több mint 6 millió felhasználó között biztosít kapcsolatot, jelentős szerepet játszott a Linux gyors továbbfejlesztésében, mivel a Linux fejlesztői számára lehetővé tette a megjegyzések, észrevételek és programrészletek kölcsönös cseréjét.

Torvalds már kezdetben több mint 60 e-mail üzenetet kapott naponta, amelyek elolvasását és megrázolását alig tudta elvégezni. Néhány témacsoporthoz (discussion group) felhívása után az elektronikus posták valóságos adrádata indult be a témaiban. Manapság számos hírcsoport foglalkozik a Linux-szal, amelyek közül a legfontosabb a comp.os.linux.announce (c.o.l.a.), ahol az új fejlesztések és programverziókat bemutatják. Levezetési listákat is felállítottak a Linux fejlesztői számára ezzel is biztosítva a hasonló típusú információk cseréjét.

Az információ és lévén kívül az Interneten keresztül fájlok kölcsönös cseréje is lebonyolítható, ami lehetővé teszi nagyobb szoftverrendszer megosztott fejlesztését, ahogy ezt a Linux fejlesztése példázza.

Az információ ilyen gyors áramlása nemcsak a Linux fejlesztői, hanem felhasználói számára is kézzelfogható előnyt jelentett. Ha ugyanis a felhasználó már esetleg a telepítés során hibákat derít fel, akkor egy kis szertencsével és az Internet segítségével a kielégítő megoldást már néhány órán belül megkaphatja. Még kereskedelmi szervizszolgáltatások esetén is ritkán fordul elő ilyen, mindenki kiterjedő támogatás.

Térmezetesen nem minden Linux felhasználó rendelkezik hozzáféréshez az Internethez, azonban a felhasználó még ebben az esetben sincs elszigetelve, ugyanis sok postafiók hálózat állított fel Linux témacsoportokat, amelyek modern keresztül is könnyen és közvetlenül hozzáférhetők.

A Linux fejlesztésének történetében külön érdekkesség, hogy magát a fejlesztési folyamatot semmilyen szigorú szervezet nem irányította. (Ez nem jelent kaotikus állapotot, mert Linus Torvalds fenntartotta/tartja a jogot, hogy a hivatalosan terjesztett rendszermagba csak az ő engedélyével kerülhet be bármi, ezzel szavatolva a minőséget – a lektor.) Ehelyett lelkes és önkéntes Internet felhasználók – gyakran professzionális szoftverfejlesztő-szakemberek vagy nagy intézmények alkalmazottai – szabad idejüket felhasználva fejlesztésekkel és javaslatikkal „vitték a témet.”

Bár Linus Torvalds fogtatta a kernel fejlesztését, sok versenyképes változat jelent meg a rendszer további területén. Ilyen területek például a GNU C fordító illesztése és karbantartása, a Linux számára készített C könyvtárak, az X Window rendszer karbantartása és adaptálása, valamint a hálózat kezelése. Más Linux önkéntesek a felhasználói és rendszerdokumentáció dolgoztak vagy a telepíthető rendszert ültetnek át hajlékony- vagy lézerlemezre.

A forráskód nemcsak a kernel, hanem a legtöbb alkalmazói program esetén szabadon hozzáférhető az Internet UNIX-szal foglalkozó hatalmas archív tárában, továbbá időszakonként az Interneten keresztül egy szoftver-katalógust (Linux Software Map) is terjesztenek, amely jelenleg 1300 körül szoftvercsomagot tartalmaz, így nehéz olyan feladatot találni, amelyre valamelyik szoftverne lenne alkalmazható.

Mivel a UNIX-szal kapcsolatos legtöbb fejlesztést amerikai egyetemeken hajtották végre, és az ilyen fejlesztések nyilvánossá válnak, sok kutatási területen elérte eredményt át lehetett ültetni a Linuxba. Az egyik ilyen terület a jól, illetve a kevésbé ismert programozási nyelvek fordítóprogramjai, vagy az Ingres és a Postgres adatbázisrendszerök, amelyeket a Berkeley-ben levő University of California egyetemen fejlesztettek ki.

Bár a hangsúly továbbra is azon maradt, hogy a rendszerhez ingyenesen lehessen hozzáérni, a kereskedelemben is terjesztenek bizonyos alkalmazásokat, mint például a Modula-2 fordítóprogram, a Smalltalk fejlesztői rendszer, interfész építők, CAD szoftverek, számos adatbázisrendszer, és az OSF/Motif grafikus kezelőfelület, amely UNIX környezetben szabványossá vált.

## 1.2 VERZIÓK

A Linux további fejlesztése jelenleg olyan nagy változtatással kísérletezik, mint annak idején a kernel első megvalósítása. Az 1.0 verzió befejezését 1992. decemberére terveztek, azonban későt. Ennek oka nem a rendszer instabilitása volt, hanem az, hogy a tulajdonságai nem igazodtak még teljesen a UNIX-rendszeréhez.

Torvalds úgy döntött, hogy az első stabil és használható verziót 0.12-esnek nevezi el, a nullával jelezve, hogy az első verzió fejlesztése még folyamatban van.

A végső 1.0-ás verziót 1994. márciusában jelent meg, és a fejlesztéseket a továbbiakban az 1.1.x verziószámokkal jelezték. A számozásnak ez a rendszere a felhasználók és az érdeklődő felek között nagy keveredést okozott. A félreírások elkerülése végett ezért minden a kernel, a C könyvtár, a fordítóprogram vagy az X11 verziószámát kell megnevezni (megkérdezni), nem pedig a CD-n szereplő verziószámot. (Segít kiismerni a káoszt, ha tudjuk, hogy mit minél neveznek. A Linuxot helytelenül keverik össze a linux alapú rendszerekkel. Amit Linuxnak hívunk, az általában linux rendszermaggal, FSF/GNU, Xfree, TeX és más szoftverekkel felszerelt terjesztés. Ilyen pl. a Slackware 3.1, amely a Walnut Creek cég linux terjesztése és jelenleg 2.0.0-ás linux rendszermaggal van felszerelve. A terjesztésen belül a csomagoknak van saját verziószáma, pl. a gcc-2.7.2. – a lektor.).

Másik gyakori félreírás az a feltevés okozza, hogy a magasabb verziószám egyben jobb és megbízhatóbb szoftvert is jelent. Hosszú ideig ugyanis a Kernel 1.0 Patchlevel 9 (vagy röviden az 1.0.9.) volt az egyetlen megbízható kernel. Az 1.1.x verziószámmal rendelkező kernelek sok új, azonban nem tökéletesen tesztelt funkciót tartalmaztak. Ezeket a verziókat ugyanis a fejlesztőknek szánták, amelyek akár egy héttel alatt is többször változhattak. A linux dokumentációkban minden felhívják a figyelmet arra, hogy a páratlan második szám az úgynevezett fejlesztői példány, amelyet nem ajánlanak stabil helyre, míg a páros számok (a nulla után párosnak számít) a befejezett, stabil verziók.

Ez a fejlesztési folyamat az 1.2-es Linux verzió megjelenésével fejeződött be, amely már sok PCI alaplapon használt NCR SCSI chipset-hez is tartalmazott illesztőket. Az 1.2.13-as kernel új szolgáltatásai alapvető fontosságúak a DOS emulátorok utóbbi verzióhoz és az iBCS2 emulációhoz.

A verziószámok a GNU C fordítóprogram esetében is hasonló félreírásokhoz vezettek. A 2.5.8. verzió ugyanis sokkal megbízhatóbb, mint a 2.6.0. verzió. (Természetesen, hiszen a 2.5.0-ban bevezetett újdonságok hibáinak kijavítása a 2.5.8-as verzióra fejeződött be, mivel a 2.5-ös gcc befejező, hibáitlan verziója a pl8. Ezzel szemben a 2.6-os verzió első kibocsátása a 2.6.0. tartalmazhat hibákat – a lektor.)

A Linux jelenlegi verziója a kereskedelmi hálózatban terjesztett versenytársai összes fontos jellemzőjével rendelkezik, azonban hatékony tervezésének köszönhetően adott hardver-konfiguráció esetén sokkal nagyobb a teljesítőképessége. Ez a jellemző azonban nemcsak a kernelre, hanem a grafikus kezelőfelületre is vonatkozik.

## 1.3 LEHETŐSÉGEK

A felhasználási lehetőségek közt az egyik legérdekesebb területet a más PC alapú UNIX változatok COFF vagy ELF formátumban levő programjainak Linux alatti futtatása jelenti (más PC UNIX-rendszer programjai viszont nem ingyenesek! – a lektor). Linux esetén a felhasználó gyakorlatilag korlátlanul hozzáférhet olyan professzionális alkalmazásokhoz, amelyeket kifejezetten az erre

a céira kifejlesztett iBCS2 emulátor futtathat. A DOS programok DOS emulátorral futtathatók a Windows emulátor, röviden WINE, amellyel MS-Windows alkalmazások közvetlenül futtathatók X11 alatt, még nincs teljesen kész, de érdekes fejlesztési területnek igerkezik (a legújabb, 1996. augusztus 18-án kiadott változat már megdöbbentő haladásról tesz bizonyoságot, elődeihez képest igen jól futtata a Winword 6.0-át – A lektor).

A legtöbb UNIX rendszerrrel ellentében a Linux már alkalmazza az **X Window System** legújabb verzióját az X11R6-ot (azóta a legújabb X verzió az X11R6.1, amely letölthető linuxra – a lektor). További Linux lehetőségek, amelyekkel a védjegyzett UNIX-rendszerek ritkán rendelkeznek, az INMOS transzputer-kártyák kezelésé és a TCP/IP futtatásának lehetősége a soros vagy a párhuzamos porton keresztül. A ISDN kártyák közvetlen kernelben keresztüli támogatása nagy távolságokon, gyors hálózati csatlakozások megvalósításához a Linuxot különösen vonzóvá teszi kommunikációs feladatok megvalósításához. És mivel a Linux további fejlesztésének nincsenek meghatározott és előre definiált irányvonala, a rendszer a továbbfejlesztése során még egyéb kellemes meglepetések is okozhat.

## 1.4 UNIX FEJLESZTÉSEK ÉS SZABVÁNYOK

A UNIX fejlesztése az 1960-as évek végén kezdődött. 1969-ben Dennis Ritchie és Ken Thompson az ATT Bell laboratóriumban készítette el az első verziót assembly nyelven. 1973-ban a **C nyelv** fordítóprogramja is elkészült, a UNIX rendszert újraírták, amely később nagy előnyt jelentett a rendszer hordozhatóságában.

A Egyesült Államok kormányával való megegyezésnek megfelelően az ATT nem árusíthatja ezt a sikeres rendszerét, így a UNIX-ot forráskódban átadta az egyetemeknek, ahol a rendszer népszerűsége tovább nőtt. 1979-ben a 7-es verzió megjelenésével az ATT megváltoztatta licencpolitikáját és a UNIX forráskódját csak a megfelelő díj megfizetése mellett biztosította. Ez viszont a Berkeley-ben működő University of California egyetemet saját változatának kifejlesztésére készítette, amely mint **BSD** (Berkeley Software Distribution) UNIX jelent meg. 1983-ban az ATT megjelentette a piacra a legújabb továbbfejlesztett specifikációját, a **System V-t**.

Az olyan cégek, mint a Sun Microsystems, a Microsoft és a DEC saját UNIX-verziókat készítettek (**SunOS**, **Xenix**, **ULTRIX**), amelyek nagyon sorosan kötődtek a BSD-hez. Annak érdekében, hogy a UNIX két legnagyobb ága (BSD és System V) egyesítve legyen, az ATT 1990-ben megjelentette a **System V 4-es változatát (Release 4)**, mint egy olyan új szabványt, amely a UNIX összes előző változatát magában foglalja.

Más intézmények is felismerték azonban a UNIX szabványosításának szükségeségét. Az IEEE (Institute of Electrical and Electronic Engineers) fejlesztette ki a **POSIX**-et, mint a UNIX-os operációs rendszerek szabványát. Ez a szabvány több részre oszlik. A **POSIX 1003.1** csak a legalsó szintű rendszerinterfész írja le. Az 1003.2 definíálja a shellre és a parancsokra vonatkozó szabványt, míg az 1003.7 fedeli le a rendszerfelügyelet lehetőségeit. Bár a **POSIX** valójában a UNIX-rendszer interfész szabványa, ezt a szabványt más operációs rendszerek – így a Windows NT – is támogatják.

A UNIX fejlesztőiből álló közösséggel azonban egy másik szabványt is megjelentetett. Bár az **X/Open Portability Guide** a **POSIX 1003.1**-en alapszik, bizonyos pontokban továbblépést jelent. A jelenlegi cél minden rendelkezésre álló UNIX változat esetén az egységes felhasználói kezelőfelület (Common Desktop Environment – CDE) és programozási interfész kialakítása. Ezzel ugyanis jelentősen csökkenhet a szoftverek hordozhatósága érdekében eddig szükséges munka menyiségét. A Linux megfelel a **POSIX** szabványoknak.

## 1.5 A FREE SOFTWARE FOUNDATION

A POSIX szabványon kívül a Linux a Free Software Foundation (FSF) General Public License (GPL) szerint másolható, terjeszthető. Az FSF-et egy évvel ezelőtt alapította Richard Stallman, aki a legendás GNU Emacs szövegszerkesztő-program fejlesztője volt. A szervezet célja, hogy a jó minőségű szoftverek bárki számára **szabadon (free) hozzáférhetők** legyenek (megjegyzés: az angol kifejezésben a free nem ingyeneset, hanem szabadságot jelent).

A szabadság azt jelenti, hogy a szoftver másolása és terjesztése, beleértve a forráskódot is, ne legyen semmilyen módon korlátozva. Az ilyen szoftver tehát alapvetően különözik a public domain szoftvertől és a shareware-től. A szabad szoftvereket ugyanis szerzői jog védi és a licencfeltételeket a GPL szabályozza.

Az a szoftver, amely a GPL szerinti licencfeltételekkel rendelkezik, a kereskedelemben **terjeszthető**, azonban bárki számára lehetséges annak másolása és továbbadása. A szoftverrel együtt a forráskódot is biztosítani kell. Ha a fejlesztők a saját szoftvereik kialakításánál szabadon hozzáférhető szoftver forráskódját használják, akkor az így készített szoftvernek ugyancsak GPL licencsel kell rendelkeznie. Ez a megkötés azonban nem vonatkozik olyan szoftverekre, amelyeket a GNU C fordítóprogramjával fordítottak vagy a GNU Emacs szerkesztőprogramjával szerkesztettek, csak azokra a programokra, amelyek GPL licenccel rendelkező forráskódot használnak.

E gyakorlat eredményeképpen jobb minőségű szoftvereket lehet készíteni, amelyeknek mindenki hasznát látja. Például a Next Company cég a GNU C fordítóprogramját használta kiindulási alapként a saját Objective-C nevű fordítóprogramjának kifejlesztéséhez. Tehát az a fordítóprogram, amelyből ők kiindultak, már viszonylag kidolgozott, ugyanakkor szabadon hozzáférhető anyag volt. A GPL kezelése mellett további kiegészítések váltak elérhetővé a felhasználók széles tábora számára, így most már a GNU C fordítóprogramja nem csak az ANSI C-vel és a C++-szal, hanem az Objective C-vel is együtt tud működni.

Az FSF azonban elindított egy olyan projektet is, amelynek célja egy olyan operációs rendszer kifejlesztése volt, amely szabadon hozzáférhető, ugyanakkor nagymértékben kompatibilis a UNIX-szal. Így tehát a GNU C fordítóprogramon az Emacs szerkesztőprogramon kívül számos UNIX kompatibilis parancsot és eszközt fejlesztettek ki, amelyet jelenleg majdnem minden Linux változat tartalmaz. Ami azonban minden az FSF mind a GNU munkatervből hiányzott, az az operációs rendszer magja. Annak ellenére, hogy a GNU kernel már a Linux megjelenése előtt fejlesztés alatt volt, a felhasználók nem alkalmazhatták (*mert túlságosan korai stádiumban van még ma is* – a lektor).

Mivel e projekt eredményeképpen számos UNIX parancsot és segédprogramot fejlesztettek ki, felhasználva a GPL licencével rendelkező forráskódokat is, a Linux fejlesztésére is jó hatással volt ez a munka. Ugyanakkor a Linux kielégítette a GNU által kitűzött célokat, mivel a Linux kernelje az FSF eszközeivel és egyéb szabadon hozzáférhető segédprogramjaival teljes, ugyanakkor ingyenesen hozzáférhető UNIX-rendszer alkotott.

Jelenleg az olyan alapvető fontosságú elemek, mint a C fordítóprogram és C könyvtár fejlesztése együttműködésben zajlik a GNU és a Linux fejlesztői között. A népszerű FTP szerverek most az érdeklődő programozók számára igen sokféle szoftvert ajánlanak, amelyek a GPL licencsel rendelkeznek. Az olyan programozási nyelveken kívül, mint a C, C++, Smalltalk, Lisp és Fortran, számos szövegszerkesztő, hibakereső program (gdb) és még PostScript interpreter is hozzáférhető.

## 1.6 A LINUX JELLEMZÓINEK ÁTTEKINTÉSE

A jobb áttekinthetőség érdekében, a következőkben röviden összefoglaljuk a Linux legfontosabb jellemzőit:

- 32 bites, többfelhasználós és több feladat párhuzamos végrehajtására képes UNIX-rendszer. A Linux lehetővé teszi, hogy több felhasználó különböző programokat egyidejűleg hajtsan végre, kihasználva az Intel 80386-os processzor illetve azt ezt követő processzorok képességeit. Az eredményként keletkező teljesítmény a klasszikus RISC munkaállomással már mindenekpenn összehethető.
- Igazodás a leggyakoribb UNIX szabványokhoz (POSIX). Mivel a Linux igazodik a meglevő UNIX szabványokhoz, a rendelkezésre álló szoftverek általában minden nehézség nélkül átvihetők Linux alá.
- Hálózati támogatás (TCP/IP). Az a számítógép, amelyen a Linux fut, a TCP/IP hálózatba könnyen integrálható. A Linux támogatást biztosít a TCP/IP használatához a legismertebb Ethernet kártyákon, modernen (SLIPP/PPP) és ISDN-en keresztül.
- Grafikus felhasználói kezelőfelület (XII). A Linux rendszer tartalmazza az X Window rendszer legújabb verzióját (Release 6), valamint a UNIX rendszerek szabványos felhasználói kezelőfelületét (OSF/Motif) megvásárolható.
- GNU segéd- és alkalmazói programok. A Linux parancsai és segédprogramjai közül sok olyan van, amely ugyan a GNU projektből származik, azonban sok funkcionális bővítést tartalmaz.
- Teljes UNIX fejlesztői környezet. A Linux alatt olyan programok fejleszthetők, amelyek problémamentesen futtathatók más UNIX rendszerek alatt is és a programozó munkáját a GNU C / C++ / Objective C fordítóprogramokon és számos szövegszerkesztő programon kívül egyéb szoftverfejlesztő eszközök is segítik.

# ALAPFOGALMAK

**A**hoz, hogy a könyvben tárgyalt témaikról jól érthetők legyenek, bizonyos általános számítógépes ismeretekkel rendelkeznie kell az olvasónak, különös tekintettel a UNIX-ra. Ez a fejezet – a legfontosabb alapelvek és szakkifejezések bemutatásával – azon olvasók munkáját segíti, akik még nem járatosak ebben a témaban.

## 2.1 TÖBB FELHASZNÁLÓ EGYIDEJŰ KISZOLGÁLÁSA

A klasszikus adatfeldolgozásban egy központi nagyszámítógép kezeli az összes adat-feldolgozási igényt, amelyhez soros csatlakozással egyszerű terminálok (csak billentyűzettel és monitorral rendelkező konzolok) tartoznak. Az egyetlen nagyszámítógépen ebben az esetben tehát szükségszerűen több felhasználó osztzik, maga után vonva a hozzáférés és a felhasználók kezelésének felügyeletét és irányítását a megosztott erőforrások megfelelő szétosztása érdekében.

Az ilyen rendszer alapját a felhasználókhoz tartozó egyedi azonosítók – *user ID* – képezik. A felhasználó csak úgy léphet be a rendszerbe, hogy megadja az azonosítóját és a hozzá tartozó jelszót (*password*). Ezt az eljárást a rendszerbe való bejelentkezésnek – *login* – nevezik. minden egyes felhasználói azonosítóhoz előre definiált hozzáférési jogok és engedélyek tartoznak, amelyeket a rendszer akkor ellenőrzi, amikor a felhasználó a fájlohoz és egyéb erőforrásokhoz hozzá szeretne férni.

A UNIX tipikus példája az ilyen rendszereknek, mivel minden egyes felhasználóhoz azonosítót, a csoport vagy csoportok számára pedig csoportazonosítót biztosít.

Egy adott felhasználó egyidejűleg több csoportnak is tagja lehet, amely különösen akkor hasznos, ha egyidejűleg több munkában vesz részt, vagy különböző körzetekben levő adatokhoz kell hozzáférnie. A könyvtárak és fájlok elérésére vonatkozó engedélyek a felhasználókhoz, illetve csoportokhoz egyedileg is hozzárendelhetők.

Minden többfelhasználós rendszerhez tartozik egy olyan különleges jogokkal rendelkező felhasználó, aki ellátja a rendszerrel kapcsolatos adminisztrációs teendőket. Ezt a személyt rendszer-adminisztrátornak nevezik, aki UNIX esetén a *root* névvel és a 0 numerikus azonosítóval rendelkezik. Az adminisztrátor hozzáférési joga nincs korlátozva. Új felhasználókat lephetet be, és hozzáférési jogokat is adhat. A 7. fejezet a rendszeradminisztrátor teendőit részletesen tárgyalja.

A utóbbi években a hardver árai csökkentek, a számítógépek teljesítménye viszont jelentősen nőtt. A grafikus felhasználói kezelőfelülettel rendelkező decentralizált rendszerek által nyújtott lehetőségek a hagyományos nagyszámítógépes rendszerek hanyatlásához vezettek. A központi nagyszámítógép és a hozzá csatlakozó terminálok rendszerét ma már olyan hálózatot alkotó számítógépek – munkaállomások – váltják fel, amelyben az egyes felhasználók egy vagy több számítógéphez illetve szerverhez rendelkeznek használati jogokkal.

A megváltozott igényeket és körülményeket a UNIX-rendszer is követte: a korábban tisztán karakter UNIX-rendszer ma már grafikus kezelőfelületet biztosít. Az az irányvonallal, amely lehetővé teszi, hogy egy adott felhasználó több számítógépet használjon egyidejűleg a különböző adatok és programok eléréséhez, az úgynevezett *virtuális terminál*ok rendszeréhez vezetett. Ez lehetővé teszi, hogy a felhasználó ugyanahoz a számítógéphez jelentkezzen be ismételten, még akkor is, ha csak egyetlen monitorral rendelkezik. A különböző virtuális terminálok közötti átkapcsolás speciális billentyűkombinációkkal végezhető el.

## 2.2 TÖBB FELADAT EGYIDEJŰ VÉGREHAJTÁSA

A többfelhasználós rendszerek több feladat közel egyidejű végrehajtását is lehetővé teszik. Az a legkisebb egység, amely ilyen rendszerben párhuzamos feldolgozásra kerülhet, az úgynevezett processz. Azok a programok, amelyek a UNIX rendszeren párhuzamosan futnak, különböző felhasználóktól származó programok, vagy a háttérben futó programok, ún. **démonprogramok**, illetve rezidens programok lehetnek.

A modern, több folyamat párhuzamos végrehajtására képes rendszer egyik legfontosabb jellemzője a processzek közötti kommunikáció biztosítása, amely a *processzek közötti szinkronizáció* és *adatcsere* lebonyolítását jelenti.

A hagyományos, egy processzorral rendelkező számítógépek esetén a CPU-t az egyes önálló processzek között felváltva kell lefoglalni, annak érdekében, hogy az a felhasználóban a párhuzamos feldolgozás benyomását keltse. Ez a feladatot az ún. *ütemező* (scheduler) látja el, amely listát készít normál feldolgozási folyamatokról, és gondoskodik arról, hogy a processzor a megadott időintervallum végen a processz futtatásátelfüggeszze és másik processzt indítson el.

A következőként kezelendő feldolgozási folyamat meghatározására az ütemező többféle stratégiát alkalmazhat. Az egyik legegyszerűbb az, amely rendszeres időközönként (például 50 ms) kiválasztja a listában szereplő következő megfelelő processzt, és ha azzal a számára lefoglalt időtartam vége előtt nem végez, akkor a lista végén helyezi el. Egy másik módszer az, amely minden egyes processzhez prioritást rendel, és minden a legmagasabb prioritással rendelkező futásképes processzt indítja el.

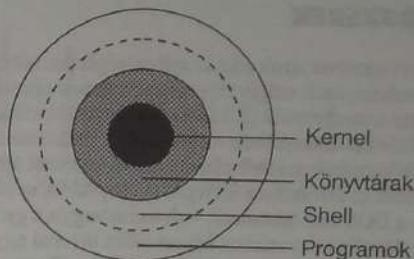
A UNIX úgynevezett prioritási szinteket (nice levels) alkalmaz, amelyek lehetővé teszik a felhasználó számára a processzek belső prioritásának befolyásolását. Így a felhasználó jelentősen csökkenheti a rendszer terheltségét, illetve a rendszer adminisztrátora a gyorsabb futás érdekében növelheti a fontos processzek prioritását.

## 2.3 MEMÓRIAKEZELÉS

A mai UNIX-rendszerek memóriakezelése alapvetően különbözik az egyszerűbb operációs rendszerekétől. Ugyanis a UNIX operációs rendszer képes arra, hogy látszólag több memóriát biztosítson a programoknak, mint amennyi valójában rendelkezésre áll.

Az a módszer, amely a virtuális memóriakezelést a Linuxban megvalósítja, az úgynevezett *lapozás* (paging). Az operációs rendszer táblázatok segítségével a nagy logikai címtérületet kisebb fizikai címtérületre képezi le. Amikor a feldolgozás során több memóriára van szükség, mint ami fizikailag rendelkezésre áll, a logikai memória azon lapjai, amelyekre újabban nem érkezett hívás, áthelyeződnek a merevlemezre.

Ha a programnak olyan logikai címre van szüksége, amely éppen a merevlemezen van, akkor a megfelelő memórialap (page) betöltődik a memóriába, míg egy másik memórialap a merevlemezre frödik, hogy a helyet biztosítja.



2.1 ábra. A UNIX-rendszer sematikus szerkezete

Annak következtében, hogy a merevlemez elérési ideje sokkal nagyobb, mint a memóriáé, az ilyen memóriakezelést lassúbb programfutással kell megfizetni.

Annak érdekében, hogy a merevlemez a virtuális és logikai memóriakezeléshez alkalmazni lehessen, úgynevezett *cserefájlokat* (*swap files*) vagy *cserepartíciókat* (*swap partitions*) kell a merevlemezen létrehozni. Ilyen fájlok vagy partíciók nélkül a főmemória a valójában fizikailag rendelkezésre álló memóriára korlátozódik.

## 2.4 A SHELL MODELL

A UNIX-rendszer szerkezetét gyakran egy olyan héjszerkezetként ábrázolják, mint ami a 2.1 ábrán is látható. A UNIX-rendszer *kernelje* olyan összetevőket tartalmaz, mint az *ütemező* vagy az *eszközkezelő*. Ezek a rutinok biztosítanak hozzáférést a hardver és a külső eszközök interfészéhez. A memóriakezelő rutinokat ugyancsak a kernel tartalmazza.

Mindegyik folyamathoz bizonyos memóriaterület tartozik. Ha a felhasználói processz a saját memóriaterületén kívüli terület elérésére tesz kísérletet, akkor a folyamat megszakad és a „segmentation fault” hibaüzenet jelenik meg. A processzhez tartozó aktuális memória tartalma ezután a *core* nevű fájba frödik ki, amely hibakereséskor hasznos.

A processzek futásának két különböző módjáról szokás beszélni: az egyik a felhasználói mód, a másik pedig a *kernel mód*. A UNIX-rendszer *shellje* (héja) olyan program, amely a felhasználóval teremt közvetlen kapcsolatot. Ez a shell tartalmazza a parancsfeldolgozót, amely a felhasználó parancsait hajtja végre, valamint az olyan alkalmazói programokat, mint a szövegszerkesztők és adatbáziskezelők.

A shell és a kernel között olyan különböző *könyvtárak* foglalnak helyet, amelyek a könyvtári függvényekhez (ezek általában C nyelven íródtak) és a kernel rutinokhoz biztosítanak hozzáférést. Ezek a könyvtárak a program lefordítása után szokásan a programhoz szerkesztődnek.

Mivel a statikus csatolással elláttott programok nagyméretű memóriát igényelnek, a modern programozási gyakorlat ma már megosztott könyvtárakat használ, amelyek két részből állnak. Egy rövid rész csak a programhoz szerkesztett könyvtárhoz a hivatkozásokat tartalmazza. Maga a könyvtár azonban csak akkor töltődik be, amikor a program fut. Ez a módszer lehetővé teszi, hogy több program a megosztott könyvtárakban levő rutinokat egyidejűleg használja, amely memóriatákarékosságot tesz lehetővé.

További előnyt jelent az a lehetőség, hogy a megosztott könyvtárat az újabb verziójával ki lehet cserélni anélkül, hogy ehhez a könyvtárakat a programokhoz újra kelljen szerkeszteni. Ehhez azonban az is szükséges, hogy az új könyvtárban a rutinokat a korábbi verzióval azonos módon kelljen meghívni.

## 2.5 FÁJLRENDSZEREK

A fájlrendszer a merevlemezen tárolt adat kezeltetőségét biztosítja. Bár ilyennel rendelkezik minden számítógépes rendszer, ezek mégis lényegesen eltérnek egymástól. A modern fájlkezelő rendszereknek hierarchikus a szerkezetük (2.2 ábra) A felhasználó fájlokat megfelelő alkönyvtárakban helyezheti el. Így a fájlokat könnyebben megtalálhatja, és a rendszer is áttekinthetőbb. A fájlokat a hozzájuk vezető alkönyvtárak nevéről álló útvonalon keresztül lehet elérni. A DOS-szal ellentétben azonban a UNIX esetén az útvonalban szereplő alkönyvtárak nevét a () jel választja el egymástól.

A UNIX esetén – a DOS-hoz hasonlóan – abszolút vagyis a gyökérkönyvtártól kezdődő útvonal és relatív, vagyis az aktuális könyvtárhoz képest előírt útvonal egyaránt megadható. A bejelentkezési könyvtár (angol home dir vagy login dir) különösen fontos. Az adatok tárolása – ha erre más rendelkezés nincs – ebben a könyvtárban megy végbe, és ez az a könyvtár, ahová a felhasználó a rendszerbe való bejelentkezéskor kerül.



2.2 ábra. Könyvtárszerkezet a UNIX-ban

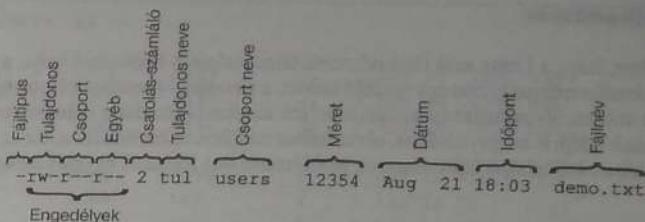
A DOS a lemezmeghajtókhoz és a merevlemezen létesített partiókhoz betűjeleket rendel. A UNIX azonban ezeket egyetlen fájlrendszerben egyesít, és a meghajtókhoz nem biztosít egyedi betűjelet. Ez azt jelenti, hogy a felhasználó nem tud különböztetni az önálló meghajtók és partiók között. Számára csak egy nagy meghajtó létezik egyetlen könyvtárendszerrrel. A lemezek és az eltávolítható adathordozók kezelése nehézségekbe ütközhet, ha ezek előzetesen nem lettek a rendszerbe iktatva. Ugyanis az eltávolítható adathordozók hozzáférése előtt azokat a módon kell parancssal a rendszerhez kell csatolni. Ezt a műveletet általában csak a rendszer adminisztrátora végezheti el.

A UNIX-ban a szabad blokkok és fájlok kezelése hasonlít a DOS-hoz. A DOS minden meghajtón létrehoz egy úgynevezett fájllallokációs táblázatot (File Allocation Table, azaz FAT), és ebben rögzít a szabad és a lefoglalt szektorokat. Egy másik szektor a gyökérkönyvtárat tartalmazza. A DOS könyvtár a tárolt fájlok nevén kívül azok attribútumát, méretét és dátumbelügét is tárolja.

A UNIX minden egyes fájl számára úgynevezett i-csomópontot (i-node) foglal le, amelyben fontos rendszerelmezőket tárol. A név, az Engedélyek és a kezdő i-node a könyvtárbejegyzésben található. Mivel ez a rendszer tárolási hely és hozzáférés szempontjából is gazdaságosabb mint a FAT, alkalmasabb a nagyobb fájlrendszerek kezeléséhez.

### Engedélyek

A UNIX alatt létrehozott fájtról az operációs rendszer nemesak a nevét és létrehozásának dátumát tárolja, hanem annak azonosítóját is, aki létrehozta, továbbá annak a csoportnak a nevét, amelyhez a fájl tartozik.



2.3 ábra. Az engedélyek megjelenítése az ls parancssal

Annak érdekében, hogy a rendszer fájllait a nemkívánatos hozzáférésektől védeni lehessen, a rendszer az engedélyeket minden egyes fájlhoz külön tárolja. Így a hozzáférés a saját tulajdonosára, vagy bizonyos felhasználók csoportjára korlátozható. Általános engedélyek azonban ugyancsak definiálhatók. További megkülönböztetés lehetséges az olvasási, írási és végrehajtási engedélyek között, amelyekre az ls (list) parancs adatkivitelében az r (read), w (write) és x (execute) karakterek utalnak. E karakterek pozíciója arra utal, hogy az adott engedély a fájl tulajdonosára, a fájl tulajdonosainak csoportjára vagy valaki másra vonatkozik-e.

Az alkönyvtárak azonban kivételek képeznek abban, hogy olvasási joggal csak a tartalom olvasható. Az alkönyvtár módosítása csak akkor lehetséges, ha a felhasználó olvasási és végrehajtási joggal is rendelkezik.

## Csatolások

A UNIX fájlrendszerének egy másik jellemzője az, hogy lehetővé teszi csatolások létesítését. Ha a fájlhoz a hozzáférés a fájlrendszer különböző pontjairól megy végbe, akkor ezt a fájlt másolni is lehetne. Ez a megközelítés azonban feleslegesen pazarolná a tárolási helyet. Ilyen esetekben a UNIX csatolást létesít, ami sokkal takarékosabb megoldás.

Azok a csatolások, amelyeket a UNIX a fájlhoz létesít, kötött (hard) vagy szimbolikus csatolások lehetnek. A kötött csatolás egy további hivatkozást jelent a könyvtárból annak i-csomópontjához. A csatolás-számláló tartja nyilván az ilyen hivatkozások számát. Ha olyan fájlt kell törölni, amelyhez több csatolás tartozik, akkor a rendszer a fájl törlését nem végzi el, csak a csatolás-számítálo értékét csökkeni 1-gyel. A fájl fizikai törlését csak akkor hajtja végre, ha a csatolás-számláló a 0 értéket éri el.

Mivel az i-csomópontok száma csak a fájlrendszeren belül egyértelmű, kötött csatolások nem hozhatók létre olyan fájlok esetén, amelyeket az adott fájlrendszeren kívüli helyekről is használni szeretnének. Ebben az esetben szimbolikus csatolások utalnak a könyvtári és alkönyvtári, illetve fájlbejegyzésekre. A szimbolikus csatolás létrehozásában nem játszik szerepet, hogy a hivatkozott fájl valójában létezik-e. A ls parancs azonban egyértelműen jelzi a különbséget a kétféle csatolás között:

```
nicetry:/etc$ ls -l hosts passwd
-rw-r--r-- 1 root      root          914 Aug 24 14:20 hosts
-rw-r--r-- 1 root      root        1454 Aug 25 17:52 passwd
nicetry:/etc$
```

A szimbolikus csatolást a fájltípus oszlopában az 1 karakter, és az eredeti fájlra egy látható hivatkozás jelzi. A kötött csatolást csak onnan lehet észrevenni, hogy a csatolás-számláló értéke megnő.

### Virtuális fájlrendszer

Annak érdekében, hogy a Linux más fájlrendszerek támogatását is biztosítani tudja, a kernel és a virtuális fájlrendszer aktuális rutinai között egy további szintet, a *virtuális fájlrendszer* biztosítja. A virtuális fájlrendszer számos olyan rutint tartalmaz, amelyek minden fájlrendszer számára rendelkezésre állnak a fájlok megnyitásához, olvasásához frásához és bezáráshoz. Ez az egyértelmű interfész biztosítja a különböző fájlrendserek problémamentes létezését egymás mellett.

## 2.6 ESZKÖZÖK

A UNIX a merevlemezeket, a terminálokat és egyéb eszközöket a fájlrendszer /dev könyvtárában speciális fájlokba képezi le. Így a programozó ezeket az eszközöket a fájlokhoz hasonló módon tudja elérni.

Ez a szemlélet, amely az eszközöket is fájlként kezeli, a felhasználó számára is előnyökkel jár. Például egy fájl tartalmát nem a konzolra, hanem a nyomtatóra szeretné küldeni, akkor csak annyit kell tennie, hogy a szabványos adatkivitel a megfelelő eszközre - /dev/lpt1 - irányítja át:

```
nicetry:/etc$ cat output.txt >/dev/lpt1
```

/dev/console	Rendszerkonzol
/dev/mouse	Soros egér
/dev/hda	Az első EIDE merevlemez
/dev/hda1	Az első EIDE merevlemez első partíciója
/dev/hda2	Az első EIDE merevlemez második partíciója
/dev/hdb	A második EIDE merevlemez
/dev/hdb1	A második EIDE merevlemez első partíciója
/dev/sda	Az első SCSI merevlemez
/dev/sdb	A második SCSI merevlemez
/dev/lp0	Az első nyomtatóport (LPT1)
/dev/null	A null eszköz (az összes adatkivitel elnyomva)
/dev/ttyN	Virtuális konzol
/dev/ptyN	Pszeudoterminál a hálózatba való bejelentkezéshez
/dev/ttySN	Soros port

2.4 ábra. A legfontosabb eszközkezelők

Ahogy a táblázatból látható, az egér (/dev/mouse), vagy a merevlemez meghajtók a /dev könyvtáron keresztül érhetők el. A /dev könyvtárban levő fájlok és az egyes eszközök két számmal állnak kapcsolatban, egy fő- és egy aleszközszámból. Ezek a kernelhez való interfészről alkotják. Az eszközöknek ráadásul két fajtájuk van: a karakter- és a blokkészszökök. Az első karakterorientált és elsőlegesen olyan eszközökhez használhatos, mint a terminál és a soros port. Az utóbbi esetében. A blokkészszököknél a hozzáérési pozíció megváltoztatható. A fő- és aleszközszám az ls -l parancs kimenetében rögtön egyértelművé válik:

```
nicetry:/dev$ ls -l
brw-rw---- 1 root      disk      3,   0 Apr 28 1995 hda
brw-rw---- 1 root      disk      3,   1 Apr 28 1995 hda1
brw-rw---- 1 root      disk      3,   2 Apr 28 1995 hda2
...
crw--w---- 1 root      root      4,   0 Jul 18 1994 tty0
crw--w---- 1 root      tty       4,   1 Aug 25 22:17 tty1
crw--w---- 1 demo     tty       4,   2 Aug 25 22:17 tty2
...
nicetry:/dev$
```

A főeszköszsám (2.4 ábra 5. oszlop) adja meg az eszköz típusát. Szokásan minden egyes főeszközözhöz a kernelben **eszközkezelő** tartozik. Ha ugyanazon típusból több eszköz is jelen van a rendszerben, akkor ezeket az aleszköszsám (2.4 ábra 6. oszlop) különbözteti meg egymástól, és ugyanaz a kezelő szolgálja ki.

## 2.7 SHELL

A shell interaktív interfészét biztosít az operációs rendszer és a felhasználó között.

A shell által elvégzett feladatokat a DOS esetén működő **command.com**-hoz lehet hasonlítni, bár a UNIX shell sokkal több lehetőséggel rendelkezik, mint a DOS parancsértelmezője.

### Szabványos shellek

A legtöbb UNIX rendszer az alábbi háromfélé shelllel rendelkezik: a Bourne shell (**sh**), Korn shell (**ksh**) és a C shell (**csh**). A Bourne shell volt az első UNIX shell, amelynek használata még nem volt túlságosan kényelmes. A Korn shell, amely a Bourne shell bővítése, a UNIX rendszereknél meglehetősen gyakran fordul elő.

A BSD UNIX-hoz hasonlóan a C shellt is a Berkeley-ben levő University of California egyetemen fejlesztették ki. A Bourne shelllel szemben a Korn és a C shell tárolja a beírt parancsokat, lehetővé téve a felhasználó számára, hogy olyan parancsokhoz térjen vissza, amelyeket már korábban beírt a parancssorra. Az alias helyettesítés lehetővé teszi azon parancsok automatikus helyettesítését, amelyeket az inicializáló fájl tartalmaz, vagy interaktív módon a felhasználó előírt. A C-szerű **szintaxis** következetében a C shell különösen kedvelt volt a programozók körében.

Ha a C shell bejelentkezési (**login**) shellként indul el, akkor a **.login** fájlban levő parancsokat hajtja végre, amely fájlnak chhez a home könyvtárban kell lennie. Mivel a felhasználó szokásan csak egy aktivitás bejelentkezési (**login**) shellt birtokol, az itt tárolt parancsok csak egyszer, a bejelentkezés után kerülnek végrehajtásra. Másrészről azonban az a shell szkript, amelyet a rendszer minden egyes C shell elején végrehajt, a **.cshrc** fájl, amelynek ugyancsak a home könyvtárban kell lennie.

### Kibővített shellek

A Linux az előbbi shellek egyikét sem alkalmazza, csak ezek **kibővített változatait**, amelyek ingyenesen hozzáérhetők, ugyanakkor a használatuk kényelmesebb. A Bourne Again shell (**bash**) a Bourne shellt, a **tcsh** shell pedig a C shellt váltotta fel. A rendszeradminisztrátorok rendszerint a Bourne shellt használják, mivel a legtöbb adminisztrációs szkriptnek erre van szüksége. A következő rész elsősorban erre a shell változatra vonatkozik.

**Interaktív használat**

Az ls, cd és man UNIX parancsokat, amelyeket a parancssorba kell beírni, az UNIX operációs rendszer hajja végre. Ezek a programok általában a /bin vagy a /usr/bin könyvtárakban vannak. A UNIX-ba való bejelentkezés után a felhasználók rendszerint az interaktív shellben találják magukat. A shell, amely csak néhány parancs felismerésére képes, a legtöbb időt a szabványos adatbeviteli cszközörlő, azaz billentyűzetről származó adatbevitel olvasásával és a megfelelő program elindításával tölti. Például az ls -l parancs beírására a shell a /bin/ls programot indítja el a -l kapcsolóval, és eredményként az aktuális könyvtár tartalmát jeleníti meg a konzolon:

```
nicetry:~$ ls
Disktools/ Help/      News/      demo.tex
Documents/ Motif/     UsrAdmin/   demo.txt
nicetry:~$
```

A legtöbb parancs a paraméterátadást is lehetővé teszi:

```
nicetry:~$ ls -l
total 10
drwxr-xr-x  2 demo    other        1024 Aug 25 17:37 Disktools/
drwxr-xr-x  2 demo    other        1024 Aug 25 17:37 Documents/
drwxr-xr-x  2 demo    other        1024 Aug 25 17:37 Help/
drwxr-xr-x  2 demo    other        1024 Aug 25 17:37 Motif/
drwxr-xr-x  2 demo    other        1024 Aug 25 17:37 News/
drwxr-xr-x  2 demo    other        1024 Aug 25 17:57 UsrAdmin/
-rw-r--r--  1 demo    other       388 Aug 25 17:38 demo.tex
-rw-r--r--  1 demo    other      452 Aug 25 17:38 demo.txt
nicetry:~$
```

**Környezet**

A program számára a paraméterátadás a **környezeten** keresztül is lehetséges. Az alábbiakban egy olyan változólistát adunk meg a hozzá tartozó értékekkel együtt, amelyeket a program a futtatása előtt automatikusan megkap. A Bourne shell esetén a set parancsral lehet lekérdezni az éppen szintánsan definíált környezeti változók listáját, ahogyan ezt a következő képernyőkép is mutatja:

```
nicetry:~$ set
PS1=;
PS2=>
PS4=+
PWD=/home/demo
SHELL=/bin/bash
SHLVL=1
TERM=linux
UID=985
_=1
ignoreeof=0
```

Ezek a változók szükség esetén törölhetők és újradefiniálhatók. Az alábbi példa az AUTO változót definiálja át VW-re:

```
nicetry:~$ export AUTO=VW
nicetry:~$ set
PS1=:
PS2=>
PS4=+
PWD=/home/demo
SHELL=/bin/bash
SHLVL=1
TERM=linux
UID=985
AUTO=VW
nicetry:~$
```

A környezeti változókat a globális rendszerbeállítások elvégzéséhez általánosan használják, például a parancsok helyéhez vezető útvonalak definiálásához (PATH), vagy a megfelelő készenléti jel tulajdonságainak beállításához (PS1). A beállításokat a felhasználó home könyvtárában a .profile nevű speciális fájlba beírva elkerülhető a műveletek ismételt végrehajtása, mivel a rendszer ezt a fájlt minden bejelentkezéskor automatikusan végrehajtja.

```
#
# Demo user .profile
#
HOST='hostname'
PATH=$PATH:/usr/local/bin
AUTO=VW
```

A Bourne shell egy másik indítófájl a .bashrc felismerésére is képes, amelynek aktiválására nemesak a bejelentkezéskor, hanem minden egyes shell elindításakor sor kerül. Ha azt szeretné, hogy bizonyos környezeti változók vélegesen legyenek beállítva, akkor azok definiálását a .bashrc fájlból kell elvégeznie.

## Átfirányítás

A szabványos adatbevitel (stdin) és a szabványos adatkivitel (stdout) elvét a UNIX-ban találták ki, és általában a billentyűzetet és a konzolt jelenti. Az olyan egyszerű parancsok, mint az ls, a művelet végrehajtásának eredményét a szabványos adatkivitelre küldik. A shell azonban lehetővé teszi a szabványos adatbevitel illetve adatkivitel fájlból illetve fájlba irányítását a (kacsafelek) és az () operátorok felhasználásával:

```
nicetry:~$ ls >list.txt
nicetry:~$
```

Az előbbi példa mutatja, hogy ebben az esetben az `ls` parancs adatkivitele a konzolon nem jelenik meg. Az adatkivitel ugyanis az operátor a `list.txt` fájlba irányítja, amelynek tartalma viszont a `cat` parancssal tekinthető meg:

```
nicetry:~$ cat list.txt
Disktools/
Documents/
Help/
Motif/
News/
UsrAdmin/
demo.tex
demo.txt
list.txt
nicetry:~$
```

Azok a parancsok, amelyek a szabványos adatbeviteliől várnak adatokat (például a `cat`), az átirányítással adatbevitelt fájlból is megkaphatják:

```
nicetry:~$ cat <list.txt
Disktools/
Documents/
Help/
Motif/
News/
UsrAdmin/
demo.tex
demo.txt
list.txt
nicetry:~$
```

A szabványos adatbevitelen és adatkivitelen kívül van még egy harmadik csatorna is, amely a konzolhoz tartozik. Mivel ez a hibaüzenetek megjelenéséhez használatos, szabványos hibacsatornának nevezik (`stderr`), amely ugyancsak átirányítható. Az egyes önálló csatornák a fájl-leírón keresztül címzhetők meg.

Kifejezés	Rövidítés	Fájl-leíró	Szabványos eszköz
Szabványos adatbevitel	stdin	0	Billentyűzet
Szabványos adatkivitel	stdout	1	Konzol
Szabványos hibaüzenet	stderr	2	Konzol

A következő utasítás a szabványos hibaüzenet átirányítását végzi el:

```
nicetry:~$ cat list.txt >error.txt
```

Az `stdout` és `stderr` fájlba irányítása egyszerre is elvégezhető:

```
nicetry:~$ cat list.txt 2>&1 >output.txt
```

Ebben az esetben a rendszer először az `stderr`, majd pedig a `stdout` átírányítását végzi el. Sok esetben szükség lehet arra, hogy a fájl tartalma az `stdout` átírányítás után megmaradjon. Ilyen esetben a `>` operátor használatára van szükség, amely az átírányított adatkivitelt a fájl vége-hez csatolja, azaz nem írja felül a fájl érvényes tartalmát.

A **cső-karakter** (`()`) használata még érdekesebb, amellyel az egyik parancs szabványos adatkivitele a másik parancs adatbevitelleként használható:

```
nicetry:~$ ls | wc
      8      8     69
nicetry:~$
```

A `wc` parancs a számára megadott adatbevitelben a szavakat, sorokat és karaktereket számolja meg. Az előző parancsban az adatbevitel az aktuális könyvtár katalógusa volt, így az eredmény a fájlok darabszámát jelenti.

### Fájlnévek tartományának kiterjesztése

Ha ugyanazt a parancsot több fájlra kell végrehajtani, akkor hasznos a helyettesítő karakterek használata, amelyekkel a parancshoz egyszerre több fájl rendelhető, így nem kell a fájlokkal az adott műveletet egyesével végrehajtani. Ha a parancs neve után szereplő paraméter helyettesítő karaktert tartalmaz, akkor a shell ellenőrzi, hogy az így képzett mintának mely fájlok felelnek meg a megfelelő könyvtárban, és a parancsot úgy hajtja végre, hogy ezeket a fájlnéveket felsorolja a fájlnév-tartomány helyett. A következő példában szereplő `cat` parancs az aktuális könyvtár összes olyan fájljára végrehajtódik, amely `.txt` kiterjesztéssel rendelkezik:

```
nicetry:~$ cat *.txt
```

A legfontosabb helyettesítő karaktereket a következő táblázat tartalmazza:

Karakter	Funkció
*	Tetszőleges számú karakter helyettesítése
?	Egy karakter helyettesítése
[abc...]	A definiáltak közül egy karakter helyettesítése
[a-z]	Tartományok definiálása kötőjellel
[!abc...]	Minden olyan karakter helyettesítése, amely nem szerepel a definiáltak között

### Karaktersorozatok kezelése

Ahhoz, hogy az önálló paramétereket karaktersorozatként lehessen átadni a parancsnak, azokat idézőjelek vagy aposztrófok közé kell foglalni:

```
nicetry:~$ echo 'Hello World!'
Hello World!
nicetry:~$ echo "Hey \"you\" there!"
Hey "you" there!
nicetry:~$
```

Az előbbi példa azt is szemlélteti, hogy amennyiben a szövegen belül szükséges az idézőjel használata, akkor a karakterszorozatot aposztrófok között kell szerepeltetni. Az aposztrófok hatására ugyanis a rendszer nem veszi figyelembe azokat a hatásokat (műveleteket), amelyekkel az egyes karakterek (pl. \$) rendelkeznek.

Az idézőjelek harmadik fajtája a fordított-aposztróf, amellyel parancsokat tartalmazó karakterszorozatok kezelése válik lehetővé.

```
nicetry:~$ cp 'which ls' .
```

A fenti parancs az ls programot az aktuális könyvtárba másolja. A cp parancs első paramétere-ként használt which az ls parancshoz vezető útvonalat adja meg. A cp után álló kifejezés azonban változóhoz is hozzárendelhető:

```
nicetry:~$ path='which ls'
nicetry:~$ echo $path
/bin/ls
nicetry:~$ cp $path .
```

### Parancsnevek rövidítése

Az alias parancssal a gyakran használt parancsok neve lerövidíthető, vagyis alias név rendelhető hozzájuk:

A Bourne shell esetében az éppen aktuálisan érvényes alias neveket az alias parancs paraméterek nélküli végrehajtásával lehet megkapni:

```
nicetry:~$ alias l='ls -l'
nicetry:~$ l
total 8
drwxr-xr-x  2 demo    other            1024 Aug 25 17:37 Disktools/
drwxr-xr-x  2 demo    other            1024 Aug 25 17:37 Documents/
drwxr-xr-x  2 demo    other            1024 Aug 25 17:37 Help/
drwxr-xr-x  2 demo    other            1024 Aug 25 17:37 Motif/
drwxr-xr-x  2 demo    other            1024 Aug 25 17:37 News/
drwkr-xr-x  1 demo    other            1024 Aug 25 17:57 UsrAdmin/
-rw-r--r--  1 demo    other             388 Aug 25 17:38 demo.tex
-rw-r--r--  1 demo    other            452 Aug 25 17:38 demo.txt
```

```
nicetry:~$ alias
alias l='ls -l'
alias ll='ls -laF'
nicetry:~$
```

Shell szkriptek készítésével hosszú parancssorok ismételt beírása kerülhető el. A shell szkript az egyéb UNIX parancsokhoz hasonlóan hajtható végre. A shell szkript készítéséhez a szkript nevével azonos néven egy fájl kell létrehozni, amelyben a megfelelő utasításokat kell elhelyezni:

```
#!/bin/sh
#
# filecount: counts the number of files in the actual directory
#
ls | wc
```

Ahhoz azonban, hogy a shell szkriptet végre lehessen hajtani, a fájl-engedélyeken módosítani kell:

```
nicetry:~$ chmod +x filecount
nicetry:~$ filecount
      9      9     73
nicetry:~$
```

A shell szkriptek azonban sokkal többre képesek, mint néhány parancs végrehajtása. A Bourne shellben levő szkript nyelv segítségével ugyanis kifejezések alkothatók, ciklusok, feltételes vizsgálatok és ennek alapján elágazások is programozhatók. Így tehát meglehetősen bonyolult rutinok készítésére nylik lehetőség. A UNIX-ban ilyen módon jelentős részket fejlesztettek ki. A rendszer a behúzáskor számos különböző shell szkriptet hajt végre. A következő részekben ilyen szkriptek készítésének legfontosabb építőköveit mutatjuk be.

## Változók

Az előzőekben már ismertetett környezeti változókon kívül – amelyekkel paraméterek adhatók át a programnak akkor, amikor azt meghívják – lokális változók is használhatók a shellben. Az export kulcsszót ezen változók előtt elhelyezve a lokális változók globálisan felismert környezeti változókká alakíthatók. A változóhoz tetszőleges karakterSOROZAT rendelhető hozzá:

```
COMPUTER=IBM
```

Ebben a példában a COMPUTER változóhoz az IBM értéket rendeltük.

Ha a változatot tartalmára van szükség, akkor a neve előtt a \$ jelet kell szerepelteni:

```
echo $COMPUTER
```

Vannak olyan speciális változók, amelyek már előre definiáltak. Ezek közé tartoznak például a parancsban használható paraméterek:

\$#	Az áltadott paraméterek száma
\$0	A shell szkript neve
\$n	Az n-ik paraméter
\$*	Az összes paraméter
\$#	Az aktuális processz ID-je
\$?	Az utoljára végrehajtott parancs által visszaadott érték

Ha a változó értékére való hivatkozás (\*) idézőjelek között szerepel, akkor a rendszer azt az aktuális értékkel cseréli fel. Ha azonban nem a változó értékére, hanem magára a hivatkozásra van szükség, akkor a karaktereket aposztrófok közé kell foglalni. Az alábbi példa ezt a két esetet szemlélteti:

```
nicetry:-$ echo "Terminal: $TERM"
Terminal: linux
nicetry:-$ echo 'Terminal: $TERM'
Terminal: $TERM
nicetry:-$
```

### Adatbevitel a billentyűzetről

A Bourne shell a `read` parancsval teszi lehetővé az adatbevitelt a billentyűzetről. A `read` egy shell változó nevét várja paraméterként, amely a felhasználótól származó adatbevitelt tárolja.

```
echo -n "Input: "
read line
echo $line
```

A példában szereplő programrészlet megjeleníti az `Input promptot`, beolvashat egy sort a szabványos adatbevitelből amelyet a `line` változóban rögzít, és végül a változó tartalmát megjeleníti.

### Elágazások

A Bourne shellben az IF segítségével hozhatók létre egyszerű elágazások. Így tehát olyan rutinok készíthetők, amelyek végrehajtása bizonyos feltételektől függ. Az IF feltételes kifejezések alkalmazását az alábbi példa szemlélteti. Ha az első feltétel (`condition1`) teljesül, akkor csak az `command1` néven képviselt parancsok kerülnek végrehajtáshoz. A második feltétel (`condition2`) teljesüléskor a `command2` parancsokat végzi el a rendszer, és ha az egymás alatt megfogalmazott egyetlen feltétel sem teljesül, akkor a `command3` parancsot hajtja végre.

```

if condition1
then
    commands1
[ elif condition2
then
    commands2 ]
...
[ else
    commands3 ]
fi

```

A feltétel általában egy külső program meghívását jelenti, amely ha nullát ad vissza, akkor a feltétel teljesült. A **test** egy speciális parancs, amely a vizsgálat elvégzésére utasítja a rendszert. A **test** parancs szöveges zárójellel is helyettesíthető. Az alábbi példa két karaktersorozat összehasonlításának eredményétől függően jeleníti meg a megfelelő üzenetet:

```

if [ "$car" = "vw" [
then
    echo "You have bought the right car!"
else
    echo "Buy a different car!"
fi

```

A **test** parancs egyéb argumentumok felismerésére is képes. (Az argumentumok részletes felsorolása a könyv végén a referencia részben található.)

A **case** parancs az elágaztatás további lehetőségét biztosítja:

```

case Value in
    Expression1)
        commands1;;
    Expression2)
        commands2;;
    ...
*)
    commands3;;
esac

```

Több feltételek kifejezés egymáshoz kapcsolása pedig a **|** vagy az **OR** operátorral valósítható meg:

```

while true
do
    echo -n """
    read line
    case $line in
        monitor|screen)
            echo kepernyo;;
        car)
            echo auto;;
        house)
            echo hazi;;
        END)
            exit 0;;
    *)
        echo 'Nem ismerem a szot!';;
    esac
done

```

### Ciklusok

A FOR ciklus segítségével az előírt elemekre ugyanaz az utasítás vagy utasítások hajthatók végre. A FOR ciklus általános szintaxisa a következő:

```

for i [ in list ]
do
    Instructions
done

```

Ha a FOR után nem áll paraméter, akkor az előírt utasítások a parancssor minden egyes elemére egyszer hajtódnak végre:

```

for i
do
    echo $i
done

```

A következő programrészlet a FOR parancs után álló paramétereket jeleníti meg egymás után:

```

for i in audi bmw mercedes volvo vw
do
    echo $i
done

```

Az alábbi példának az előzőknél több gyakorlati haszna van. Ez a szkript ugyanis az összes olyan fájlt, amelynek kiterjesztése .doc, megváltoztatja .txt kiterjesztésüre;

```

for i in *.doc
do
    echo $i
    tmp='basename $i .doc'
    mv $i $tmp.txt
done

```

A WHILE ciklussal az előírt utasítások végrehajtása mindenkor ismételhető, amíg az átadott feltétel teljesül. Az IF-hez hasonlóan a feltételt itt is egy külső parancs végrehajtásának eredménye kezeli:

```

while condition
do
    commands
done

```

A következő programrészlet a WHILE ciklus használatára mutat példát:

```

while [ "$line" != "END" ]
do
    echo -n "* "
    read line
    echo $line
done

```

## 2.8 KERESÉSI MINTÁK

Keresési mintákat nemcsak a shellben, hanem az olyan szövegszerkesztőkben, mint az Emacs és a vi, valamint a grephez hasonló keresőprogramokban is alkalmazni lehet. A keresési minták kialakítása elsősorban a ? és a \* metakarakterekkel meg van végbe. E karakterek jelentése azonban különböző akkor, ha shellben illetve egyéb programokban szerepelnek.

Különbséget kell tenni az egyszerű helyettesítő karakterek és az egyéb keresési minták kialakításhoz használatos karakterek között, mivel az egyszerű karakterek fájlnév-minták és reguláris kifejezések kialakításához is használatosak. A metakarakterek behelyettesítésének eljárását shelllek esetén kiterjesztésnek, míg a hagyományos kifejezések esetén mintaegyzettétésnek nevezik.

A könyv előző alfejezetéinek egyike már tárgyalta a helyettesítő karakterek használatát, így erre rölkötünk le. A reguláris kifejezések képzéséhez használatos metakarakterek felsorolását és jelentését ismertetjük.

. Bármelyik adott karakter.

\* Tetszőleges darabszám, bármilyen karakter

+ A megelőző karakterek vagy kifejezésnek legalább egyszer elő kell fordulnia. Az a + egy vagy több a karaktert jelent.

? Az utolsó karakterek vagy kifejezésnek pontosan egyszer vagy egyszer sem szabad előfordulnia.

<sup>8</sup> Sor eleje. Olyan kifejezések létrhozásához használatos, amelyek a sor elején fordulnak elő. Például a `^ab*` a sor elején egy karaktert jelent, amelyet tetszőleges számú előfordulásban b követhet.

- § Sor vége.
- [ ] A zárójelben belül bármelyik karakter. A ^ és a - karaktereknek speciális szerepe van itt. Ha a zárójelbe foglalt karakterek a ^ karakterrel kezdődnek, akkor az utasítás negálódik, vagyis minden karakter engedélyezve lesz, kivéve azokat, amelyek fel vannak sorolva. A karakterek tartománya a - karakterrel adható meg. Az [A-Z] az összes nagybóljű jelentő, míg az [^a-c] minden karakter, kivéve az a-t a b-t és a c-t. Az [123] az 1, 2 vagy a 3 karakterek bármelyikét jelenti.
- └ Felfüggeszti az utána következő karakter speciális jelentését, így lehetővé teszi metakarakterek használatát a kifejezésekben.
- ( ) Bezárja a reguláris kifejezést.

E karakterek jelentése a programtól függően nemileg elérhető, és további metakarakterek is definiálhatók. A következő példák a reguláris kifejezések használatát mutatják a grep parancs felhasználásával. Ezt a parancsot a -E kapcsolóval meghívva kibővíttet kifejezések használata válik lehetővé.

```
nicetry:/root/konyv/scripts$ cat testfile
EZ egy olyan probaszoveg, amelyet a grep
parancs kipróbálásához készítettem. A parancs jellemzoje
a gyors mukodes, amely segitsegevel nagy állományokat
is ellenorizni lehet.
nicetry:/root/konyv/scripts$ grep amely testfile
EZ egy olyan probaszoveg, amelyet a grep
a gyors mukodes, amely segitsegevel nagy állományokat
nicetry:/root/konyv/scripts$ grep '^a testfile'
a gyors mukodes, amely segitsegevel nagy állományokat
nicetry:/root/konyv/scripts$ grep e$ testfile
parancs kipróbálásához készítettem. A parancs jellemzoje
nicetry:/root/konyv/scripts$ grep 'A.*ll' testfile
parancs kipróbálásához készítettem. A parancs jellemzoje
```

Az l+ ?m keresési minta az l karakter egy vagy több előfordulását jelenti, amelyet opcionálisan egy szóköz, és az m karakter követ. Az (l1)+[ ^\$] minta azt jelenti, hogy az l1 kifejezés legalább egyszer előfordul, amelyet azonban a sor vége karakternek nem szükségszerűen kell követnie. A kifejezést aposztrófok közé kell foglalni, hogy azt a shell ne próbálja meg végrehajtani, hanem közvetlenül adja át a grep parancsnak.

## 2.9 DÉMONOK

A démonok speciális processzek, amelyek a háttérben futnak és általában fontos feladatokat látnak el. A démonokkal az operációs rendszer nagy egységei önálló programokként futnak. Továbbá az önálló démonok aktiválhatók, a konfiguráció módosítása után, de még a működéstük során is újraindíthatók. Mivel a démonok önálló processzként futnak, egymáshoz képest párhuzamosan is futnak, így más programokat nem blokkolnak. A következőkben a démonokra mutatunk néhány példát.

### Nyomtató démon (lpd)

A sornyomtató démon (`lpd`) rendszeres időközökben ellenőrzi a `/usr/spool` könyvtárat, hogy abban nem szerepel-e új nyomtatási munka és ha ilyen előfordul, akkor azt a megfelelő nyomtatóra küldi. A Linux a BSD UNIX-ból ismeretes `lp` parancsral biztosítja a fájlok nyomtatára küldését. Az új nyomtatási munkák rendszerint a nyomtatási sor végéhez csatolódnak, mielőtt a nyomtató démon azokat a megfelelő nyomtatóra küldené.

### Cron démon

Ha a felhasználó bizonyos időpontokban vagy szabályos időközönként egy programot szeretne végrehajtani, akkor ezt a cron démon teszi lehetővé. Ez a démon minden egyes felhasználóhoz egy táblázatot rendel, amelyben nyilvántartja, hogy egy-egy processznek mikor kell elkezdenie. A végrehajtott parancs adatkivitelt vagy az erre vonatkozó hibaüzenetet a felhasználó e-mailen keresztül kapja meg. Ha a szkriptet egy bizonyos időpontban csak egyszer kell végrehajtani, akkor az a t parancs használatára van szükség. Ha valamelyik processz rendszeres időközönként ismételt végrehajtására van szükség, akkor ehhez a felhasználó cron démon táblázatában (`crontab`) egy bejegyzésre van szükség. Ez a feladat a `crontab` parancssal végezhető el.

### Syslog démon

Mivel a démon szokásosan nem küld adatokat a konzolra, önálló protokoll démon szolgálja a más démonoktól származó adatkivitel és hibaüzenetek kezelését. Az így feldolgozott adatkivitel már megjelenik a konzolon, fájlba írható, vagy e-mailként továbbítható a rendszer-adminisztrátornak.

## 2.10 A PARANCSOK ÁTTEKINTÉSE

Annak érdekében, hogy a Linux-szal azok a felhasználók is könnyen megismerkedjenek, akik számról új ez a rendszer, a következőkben felsoroljuk a legfontosabb UNIX parancsokat és rövid leírást is nyújtunk hozzájuk. A részletes információt a szabványos UNIX kézikönyvekben vagy a számítógépen keresztül közvetlenül hozzáférhető elektronikus kézikönyvekben találhatja meg.

- **ls** – megjeleníti a fájlok és könyvtárak nevét. További lehetőségekkel a fájlméret, a hozzá tartozó engedélyek, a fájl tulajdonosai is megjeleníthetők. Rekurzív módon a teljes könyvtárfa is lekérdezhető.
- **cd** – paraméter nélkül előirva a home könyvtárat aktuális könyvtárnak jelöli ki.
- **cp** – az előírt fájlokat az egyik könyvtárból a másikba vagy másik fájlba másolja. További opcióként rekurzív módon a teljes könyvtárfa másolását is lehetővé teszi.
- **mv** – a fájlrendszeren belül áthelyezi vagy átnevezi a fájlt, illetve a könyvtárat.
- **rm** – törli a fájlt. További opcióként a teljes könyvtárfa törlését is lehetővé teszi rekurzív módon.
- **mkdir** – új könyvtárat hoz létre.
- **rmdir** – törli az üres könyvtárat.
- **exit** – kilépett az aktuális shellből.

- more – a szövegfájl tartalmát oldalról oldalra haladva megjeleníti a konzolon. További lehető ségékként karakterszorozatok keresését is lehetővé teszi a fájlból.
- man – megjeleníti az on-line dokumentációt, azaz a parancshoz tartozó Manual oldalt.
- cat – lehetővé teszi a szövegfájlok egymás után másolását
- grep – keresi a megadott fájlokban az előírt mintát.
- passwd – a felhasználó jelszavának módosítását teszi lehetővé.
- ps – felsorolja az összes futó processzt az azonosítójukkal együtt.
- kill – az előírt ID-vel rendelkező processz futását leállítja (befejezi).
- su – lehetővé teszi a felhasználói azonosító (user ID) átmenneti módosítását anélkül, hogy ehhez a bejelentkezési eljárást meg kellene ismételni.

# A LINUX JELLEMZŐI

**E**z a fejezet a Linux legfontosabb jellemzői közül azokat tárgyalja, amelyek ezt a rendszert más UNIX változatuktól és egyéb PC operációs rendszerektől megkülönböztetik.

## 3.1 VIRTUÁLIS KONZOLOK

Sok PC UNIX implementáció támogatja a *virtuális konzolokat*, amelyek több független bejelentkezési eljárás kezelését teszik lehetővé egyetlen konzolon. Az egyes önálló munkafolyamatok közötti átkapcsolás általában speciális billentyűkombinációval végezhető el.

A Linux esetén ezt a műveletet valamelyik funkcióbillentyű **<Alt>** billentyűvel képzett kombinációja teszi lehetővé. A virtuális konzolok maximális száma a kernelben van meghatározva. Az `/etc/inittab` fájl tartalmától függ az, hogy ezen konzolok közül melyiken kell megjeleníteni a bejelentkezési promptot.

Az X11 esetén az **<Alt>** billentyű az alkalmazások számára van fenntartva. Az X Window System esetén a másik virtuális konzolra való átkapcsolás a megfelelő funkcióbillentyű és a **<Ctrl-Alt>** billentyűkkel alkottott kombináció lenyomásával érhető el. Így válik lehetővé a felhasználó számára az X Window System grafikus interfésze és a virtuális konzolok szövegorientált interfész közötti átkapcsolás.

Több X szerver elindítása is lehetséges. Ezt a műveletet azonban lehetőség szerint el kell kerülni, mivel ehhez ritkán elég a memória, és így a rendszer teljesítőképessége jelentősen csökkenhet. Helyette az X Window System virtuális ablakkezelője – például `olvwm` vagy `fwm` – javasolt, amely a többszörös virtuális konzol lehetőségét ugyanekké felajánlja.

## 3.2 A LINUX FÁJLRENDSZERE

A Linux alatt rendelkezésre álló fájlrendszerek sokasága első pillantásra lehet hogy zavaró, ezért a következőkben felsoroljuk ezeket, és megadjuk a legfontosabb jellemzőket is.

### MINIX fájlrendszer

Az első Linux verziók csak egyetlen fájlrendszerrel rendelkeztek, amely erősen támászkodott a MINIX fájlrendszerre. Ezzel a fejlesztők nemcsak a teljesen új rendszer kialakításának munkáját takaríthatták meg, hanem már kezdettől biztonságosan működő fájlrendszerrel dolgozhattak, bár a MINIX fájlrendszer néhány komoly hátrányaival is rendelkezik:

a fájlnévek nem lehetnek hosszabbak 14 karakternél, és a partició mérete 64 Mb-ra korlátozott. Annak ellenére, hogy a Linux/MINIX rendszer újabb verziói már lehetővé teszik a hosszabb fájlnévek használatát is (30 karakter), ezt a fájlrendszer szinte alig használják.

Az azonban érdemes megemlíteni, hogy sok védjegyzett System V implementációval szemben már ez az első rendszer is támogatta a szimbolikus csatolásokat.

**Extended fájlrendszer (ext)**

Az előbbi korlátok kiküszöbölésére egy francia programozó, Remy Card, alkotta meg azt az első fájlrendszer, amely maximálisan 2 Gb méretű particiók és fájlok létrehozását tette lehetővé és a fájlnévek méretének maximumát 255-ben szabta meg.

Azban ennek a rendszernek is voltak gyenge pontjai. A szabad blokkok és az i-esomópontok kezelése nem bitvettorban, hanem csatolt listában történt. Ez viszont hosszabb működtetés után nagyon felszabdalta a memóriát, ami viszont jelentős mértékben növelte a hozzáférési időt.

**Extended2 fájlrendszer (ext2)**

Az Extended fájlrendszerből kifejlesztett Extended2 a legszélesebb körben elterjedt fájlrendszer. Ebben a rendszerben már kiküszöbíték a fragmentációs problémákat és a fájlrendszerekre vonatkozó 2 Gb-os méretkorlátost. Az Extended2 továbbá egy olyan mechanizmust is támogat, amely az elveszett szektorokat speciális könyvtárban (*Lost+found*) tárolja és az esetleges rendszerrözzegsorral megsérült fájlrendszeret az indításkor felismeri, és a javítást egy speciális segédprogrammal (`e2fsck`) teszi lehetővé.

**Xia fájlrendszer**

Újabb és gyorsabb fájlrendszer megalkotásában nem az Extended2 fájlrendszer volt az egyedüli próbálkozás. Majdnem ezzel egyszerre jelent meg a Xia fájlrendszer, amely a nevét a fejlesztője, Frank Xia, után kapta. Ez a rendszer a partició maximális méretét ugyancsak 4 Gb-ra növelte, és fájlnévek maximális hosszúságát pedig 248 karakterben szabta meg. A fájlméret maximuma viszont továbbra is 64 Mb maradt.

**ISO 9660/HighSierra fájlrendszer**

A CD-ROM-ök elérésének biztosítására a Linux az ISO 9660 és a High Sierra fájlrendszereket egyaránt biztosítja és – a Rockridge Extensions rendszer alkalmazásával – hosszabb fájlnévek használatát is lehetővé teszi.

**Proc fájlrendszer**

Ez a fájlrendszer nem a fizikai fájlokat kezeli, hanem hozzáférést biztosít a kernelben levő adatokhoz és az éppen futó processzekhez. A Proc fájlrendszer a gyökérkönyvtár /proc alkönyvtárában kell elhelyezni. Ez a könyvtár minden egyes futó processzhez további alkönyvtárrakkal rendelkezik, amelyek neve a processz ID-jének felel meg. Ezek az alkönyvtárak olyan fájlokat tartalmaznak, amelyek interfész biztosítanak az adott processzre jellemző információkhoz és a kerneltáblákhoz. Általánosságban ezek virtuális ASCII fájlok, amelyek tartalma a `cat` parancssal jeleníthető meg. Igy a felhasználó lekérdezheti a processzhez tartozó parancssor tartalmát, illetve a környezeti változók értékét. Ezen a fájlrendszeren keresztül azonban a memóriaigényekről, a szüüf- vagy az éppen aktuális processz állapotáról is információ nyerhető.

### 3.3 ADATCSERE

Ritkán fordul elő, hogy a PC-n a Linux az egyedüli operációs rendszer. Rendszerint a merevlemez egy másik partíciója vagy egy másik merevlemez a DOS-t tartalmazza az MS-Windows-zal, az OS/2-öt vagy egyéb PC-re készített UNIX változatot. Ha a felhasználó a DOS-ról a Linux-ra tért át, nyilvánvaló és jogos igénye, hogy megszokott programjait használni tudja.

A következőkben azt mutatjuk be, hogy a Linux miként működtethető együtt más operációs rendszerekkel és hogyan lehet az adatok és programok kölcsönös cseréjét lebonyolítani.

A rendszer indításakor az ún. *boot manager* teszi lehetővé a betöltendő operációs rendszer kiválasztását. Ezt a funkciót – egyéb feladatok mellett – a Linux Loader (*LILO*) látja el a Linuxban. Az 5. fejezet, amely a rendszer telepítését ismerteti, a LILO telepítését és működését részletesen leírja.

#### MTools

A DOS-ban létrehozott fájlok olvasását és másolását szinte minden operációsrendszer lehetővé teszi ma már. A UNIX esetében is léteznek olyan szabadon hozzáférhető programok, amelyek ilyen a célt szolgálnak. A Linuxban ezt a feladatot az MTools segédprogram látja el. A következő képernyőképen egy DOS-os lemez könyvtárlistája látható, amelyet MTools *mdir* és *mcopy* parancsa készített:

```
nicetry:/home/demo# mdir a:
Volume in drive A has no label
Directory for A:/_
_MTEST EX_      55370    8-05-92   12:00a
SMSETUP MS_     2736     8-27-92   9:44a
PLUGTW EX_     15047    8-26-92  11:23a
SMSETUP EXE_    24624    8-05-92   6:38p
MSCOMSTF DL_   41320    8-05-92  12:00a
SMSETUP LST_    675     8-26-92   5:41p
NOTES          16346    7-14-93   9:14a
SMSETUP IN_     1424     7-21-93  12:32p
3RD PARTY      <DIR>    7-13-93   5:21p
TWAIN          <DIR>    7-13-93   5:21p
18 File(s)     18944 bytes free
nicetry:/home/demo# mcopy -t autoexec.bat .
Copying AUTOEXEC.BAT
nicetry:/home/demo# mdel a:autoexec.bat
nicetry:/home/demo#
```

A lemezmeghajtó eszközök szabadnak kell lennie, hogy hozzáférhessünk a hajlékonylemezen lévő MTools segédprogramhoz. Ez azt jelenti, hogy a hajlékonylemeznek nem szabad a rendszerbe logikailag illeszthe lennie.

#### A DOS fájlrendszer

A Linux az MTools segédprogramon kívül egy további módszert is biztosít a DOS-os tároló eszközök eléréséhez, a DOS fájlrendszer, amely lehetővé teszi a DOS-os lemezek és a merevlemez DOS partícióinak logikai beillesztését a Linux könyvtárfájába. Ez az elérési mód gyorsabb annál, mint amit az MTools lehetővé tesz, mivel az adatkiviteli és adatbeviteli műveletek kihasználhatják

az operációs rendszer cache memóriája által nyújtott előnyökét. A következő példa egy DOS parti-ci elérését szemlélteti a DOS fájlrendszerrel:

```
nicetry:/home/demo$ cd /msdos
nicetry:/msdos# ls -l
total 0
nicetry:/msdos# cd ..
nicetry:/# mount -t msdos /dev/hd1 /msdos
nicetry:/# cd msdos
nicetry:/msdos# ls -a
./           command.com    format.com    tools/
../          config.sys     io.sys
autoexec.bat dos/         msdos.sys
nicetry:/msdos# cd dos
nicetry:/msdos/dos#
```

A lemezek logikai beillesztésének azonban van egy hátránya is. A lemezeket nem lehet szabadon be helyezni és kivenni a meghajtóból, ugyanis minden egyes lemezcserénél a mount és umount parancsokat is végre kell hajtani. Ha ugyanis a felhasználó olyan lemezet vesz ki a meghajtóból, amely a rendszerbe logikailag illesztve van, akkor a lemez tartalma megsérülhet (a lemez-cache-ből nem frídnak vissza az adatok fizikailag a lemezre).

Mivel a DOS sem a felhasználói azonosítókat (user ID), sem a csoport azonosítókat (group ID) nem ismeri, a logikailag rendszerbe illesztett DOS fájlrendszer önálló fájljainak eléréséről nem lehet egyedileg rendelkezni. Csak arra van lehetőség, hogy a kötet logikai beillesztéskor az általános fájlrendszerhez hozzáférési jogokat illetve felhasználói és csoport ID-ket rendeljünk. Ezzel tehát a fájlrendszer, mint teljes egység hozzáférése felügyelhető.

Az ftp szervereken megtalálható on-line Manualban a README.dosfs fájlban a DOS fájlrendszer és a rendelkezésre álló opcionális teljes leírása megtalálható a fájlrendszer forráskódjával együtt. A mount-tal két oldal foglalkozik. Az egyik magával a parancssal, a másik a C könyvtárban levő rutinnal. A megfelelő on-line Manual oldal megjelenítéséhez a man parancs hívásakor a szekciót is elő kell írni, ahogy ezt a következő példa is szemlélteti:

```
man 8 mount
```

Az alábbiakban azokat a legfontosabb kapcsolókat soroljuk fel, amelyek mount parancs hívásakor – DOS fájlrendszer esetén – alkalmazhatók:

- uid=<azonosítószám>
- gid=<azonosítószám>
- umask=<azonosítószám>
- conv=binary vagy text vagy auto (lásd a következő bekezdést)

A következő példa a mount parancs használatát szemlélteti kapcsolókkal együtt. (A kapcsolók leírása a könyv referencia részében található meg.)

```
nicetry:-# mount -t msdos -o umask=000 /dev/hd1 /dosc
```

A kapcsolók az /etc/fstab fájlban is elöírhatók:

/dev/hda3	swap	swap	defaults	1	1
/dev/hda2	/	ext2	defaults	1	1
/dev/hda1	/dos	msdos	defaults	1	1
none	/proc	proc	defaults	1	1

### Szövegkonverzió

A UNIX és a DOS közötti kölcsönös adatcseré alapvető problémáját az okozza, hogy a két rendszerben egymástól eltérő a kocsi vissza (CR) és a sorinelés (LF) karakterek használata. Az mcrypt esetében ezt a problémát a -t paraméterrel oldották meg, amellyel megadható, hogy a fájl bináris vagy – konvertálással – szöveges módban legyen másolva. Ez az eljárás azonban nem minden működik megbízhatóan. Azt ugyanis nem minden lehet bizonyossággal megállapítani, hogy a fájl szöveges vagy bináris.

## 3.4 BETÖLTHETŐ MODULOK

A klasszikus UNIX-rendszer általában **monolitikus kernelekkel** rendelkeznek így a teljes kernel újra kell szerkeszteni (link) akkor, amikor a rendszert új meghajtóval egészítik ki. Bár a Linux kerneljének felépítése alapjaiban véve ehhez hasonló, úgynevezett **betölthető modulokat** is biztosít, vagyis olyan tágymodulokat, amelyek a rendszer futása alatt betölthetők vagy eltávolíthatók. Számos olyan Linux kezelőprogram létezik, amely betölthető modulként használható. A 4.3 részben bemutatott iBCS2 emulátor is ilyen.

A következő táblázat a modulok adminisztrációjához szükséges parancsokat tartalmazza:

insmod	a paraméterként megjelölt modult beilleszti a rendszerbe
depmod	modulok egymástól való függőségét társa fel annak érdekében, hogy lehetőség legyen minden olyan modul automatikus betöltésére, amelyektől az általunk betölteni kívánt modul függ
modprobe	tesztelési céllal az összes elérhető modult megpróbálja betölteni és egy konfigurációs fájlt készít, amelyet az automatikus töltő használ fel (kerneld)
rmmod	eltávolítja a paraméterként megadott modult
lsmod	felsorolja az éppen betöltött modulokat

Azok a modulok, amelyekre majd szükség lesz, rendszerint az egyik rc fájlon keresztül töltődnék be (lásd a 7.2 szakasz) vagy automatikus modultöltő telepítésével tesszük elérhetővé azokat. A modprobe parancsot általában a modul betöltésére szokás használni, mivel ez a további szükséges modulokat is automatikusan betölti.

## 3.5 HANG

A Linux az ismertebb PC hangkártyák használatát is támogatja. A szükséges kezelőprogramok kernel kódban állnak rendelkezésre, azonban ezeket a befordítás előtt a make config parancssal konfigurálni kell. Amint a megfelelő kezelőprogram a kernelben jelen van, további kezelőprogramok is hozzáférhetővé válnak:

/dev/mixer	keverő a különböző hangcsatornákhoz
/dev/audio	sun audio eszköz
/dev/sequenzer	sorrendi vezérlő eszköz
/dev/midi	eszköz MIDI adatok lejátszáshoz
/dev/sndstat	státusz eszköz az audio rendszerhez
/dev/dsp	audio eszköz (raw-formátum)

Az audio adatok rögzítése és lejátszása számos programmal lehetséges (pl. vrec és vplay). Az egyszerűbb esetekben azonban a hangfájlok rögzítése illetve lejátszása a cat parancssal is elvégzhető.

Rögzítés:

```
nicetry:~$ cat < /dev/audio > sound.au
```

Lejátszás:

```
nicetry:~$ cat sound.au > /dev/audio
```

### 3.6 ALTERNATÍV SHELEK

Mivel a UNIX eredeti shelljeinek (sh, ksh és csh) forráskódja szabadon nem férhető hozzá, és mivel sokkal kényelmesebb alternatív shellek is léteznek, a Linux verziók inkább ezeket használják.

#### A bash shell

A bash shellt a többi egyéb programokhoz hasonlóan a Free Software Foundation GNU projektjén belül fejlesztették ki. Ez a shell a Korn shell bővítményének is tekinthető, amely sok jellemzőben hasonlít a tcsh shellhez (lásd a következő részt). Továbbá a vezérlő billentyűk lehetővé teszik olyan hasznos funkciók aktiválását, mint az útvonalnevek automatikus kiegészítése vagy a korábban beírt parancsok nevének ismételt felhasználása. A következő példa ezt a jellemzőt mutatja be részleteiben.

A /home/stefan könyvtáróból a /usr/src/linux könyvtára váltáshoz hagyományos esetben a cd /usr/src/linux parancsot kell betűről betűre beírni. A bash-ba beépített automatikus útvonal felismerő rendszer azonban lehetővé teszi, hogy az útvonal nevéből a felhasználónak csak annyit kelljen beírnia, amiből az már egyértelműen megállapítható. A <Tab> billentyű lenyomására a rendszer az útvonalnevet automatikusan kiegészít.

```
nicetry:~$ cd /usr/s
```

A <Tab> billentyű lenyomására az alábbi sor jelenik meg:

```
nicetry:~$ cd /usr/src/
```

Ezután következik az l (kis L) karakter beírása:

```
nicetry:~$ cd /usr/src/l
```

amely a <Tab> billentyű lenyomására a következőre egészül ki:

```
nicetry:~$ cd /usr/src/linux
```

Ahhoz, hogy az itt tárolt `Makefile` szerkesztése lehetővé váljon, csak az alábbi néhány karakter beírása szükséges:

```
nicetry:~$ emacs M
```

A <Tab> billentyű lenyomására a sor a következőre egészül ki:

```
nicetry:~$ emacs Makefile
```

Ebben a könyvtárban a `Makefile` volt az egyedüli fájl, amely az `M` karakterrel kezdődött.

Ha a kiegészítési kísérlet több változathoz vezet, akkor figyelmeztető hangjelzés hallatszik és a <Tab> billentyű ismételt lenyomására a `bash` az összes lehetséges változatot megadja.

A fel (up) és le (down) billentyűkkel a felhasználó a korábban beírt parancsok nevének listájában lépkedhet. Az előbbi példához visszatérve a fel (up) billentyű lenyomására a következő lesz látható a parancssorban:

```
nicetry:~$ emacs Makefile
```

Ugyanezt a billentyűt ismételten lenyomva pedig a

```
nicetry:~$ cd /usr/src/linux
```

parancs jelenik meg.

Ha a promptban a host nevét és az aktuális útvonalat is látni szeretnénk, akkor ehhez a megfelelő home könyvtárban levő `.bashrc` fájlt a következő, környezeti változót definiáló sorral kell kiegészíteni:

```
PS1=' : '
```

A Bourne Again shell számos egyéb jellemzővel is rendelkezik, amely itt részletesen nem tárgyalható. Ha továbbiakat szeretné erről a téma ról megtudni, keresse meg az on-line Manualban a `bash`-sal foglalkozó részt.

## A tcsh shell

A Bourne Again shellhez hasonló másik változat a `tcsh`, amely a C shell kiterjesztése. A `bash`-hoz hasonlóan ez is automatikusan biztosítja az útvonal- és parancsnevek automatikus kiegészítését,

és a <Tab> billentyű lenyomására, a kurzormozgató billentyűkkel a beírt parancsok névlistájának görögítését, valamint a parancsor szerkesztését. A rendelkezésre álló lehetőségek a tcsh program esetén a <Ctrl-D> kombináció lenyomására jeleníthetők meg. Az alábbi példa e billentyűkombináció használatát szemlélteti.

```
nirctry:~$ cd /usr/src/ <CTRL D>
```

Érdekes további jellemző a hibásan beírt parancs- vagy fájlnevek automatikus javítása. A jellemzőt a <Meta-> kombináció lenyomásával lehet aktiválni.

### 3.7 KIBÓVÍTETT PARANCSOK

Sok szabványos UNIX parancs, amelyet a Linux használ, és amelyeket a GNU projectben fejlesztek ki, a szokásos parancsok bővített változata. A Linux alatt például az ls parancs 20-nál is több paraméterrel futtatható, amelyekkel a könyvtárlista megjelenítésének módja, rendezésének sorrendje, illetve a szimbolikus megggyezések kezelése egyaránt előírható. Az ls parancsnak vanak olyan változatai is, amelyek a könyvtárlista sorait a típusról (csatolás, futtatható fájl, tar fájl, alkönyvtár stb.) függően különböző színnel jelzi.

Az alábbi részlet a GNU ls parancsra vonatkozó on-line Manual oldalból származik:

LS(1L)	LS(1L)
<b>NAME</b>	
ls, dir, vdir - list contents of directories	
<b>SYNOPSIS</b>	
<pre>ls [-abcdefghijklmnopqrstuvwxyz] [-w cols] [-T cols] [-I pattern] [--all] [--escape] [--directory] [--inode] [--kilobytes] [--numeric-uid-gid] [--no-group] [--hide-control-chars] [--reverse] [--size] [--width=cols] [--tabsize=cols] [--almost-all] [--ignore-backups] [--classify] [--file-type] [--full-time] [--ignore=pattern] [--dereference] [--literal] [--quote-name] [--recursive] [--sort=none,time,size,extension] [--format=long, verbose,commas,across,vertical,single-column] [--time=atime,access,use,ctime,status] [--color[=yes,no,tty]] [--colour[=yes,no,tty]] [--7bit] [--8bit] [--help] [--version] [name...]</pre>	

#### A GNU tar

A tar parancs GNU változata a tar archívok automatikus tömörítéséhez és a tömörítés megszüntetéséhez a z paraméterrel rendelkezik, az M paraméter pedig többköteles archívok létrehozását teszi lehetővé, vagyis olyan archívokat, amelyeket több adathordozó egységen tárolnak.

### A gzip parancs

A Linuxban a compress parancsot a gzip váltotta fel. Ez a program, amely sok más tömörítési technikával kompatibilis, sokkal hatékonyabb, mint a compress. A tar parancssal önálló fájlok és alkönyvtárak egyetlen archívba kombinálhatók. Például az a tömörített tar archív, amelyet a UNIX compress parancsával készítettek, 1.5 Mb méretű, míg a gzip parancssal készített tömörítés csak 900 Kb helyet igényel.

A kibővített programok mindegyike forráskódban is elérhető és más UNIX gépeken is lefordítható. A Linux előnye az, hogy ezeket a parancsokat már kezdettől fogva alkalmazza, így e segédprogramok konfigurálásához, lefordításához és telepítéséhez külön műveleteket nem kell a felhasználónak végrehajtania.

A kibővített parancsok és a hozzájuk tartozó paraméterek felsorolása önmagában egy könny terjedelmű lenne, ezért ettől eltekintünk. A parancsok közül a legfontosabbakat azonban a könyv végén a referencia részben megadjuk. Adott parancssal kapcsolatban továbbiak az on-line Manual megfelelő oldalán olvashatók.

# EMULÁTOROK

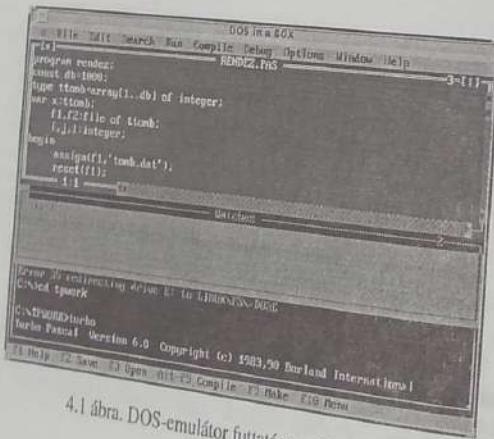
**A** Linuxhoz egyre több emulátor áll rendelkezésre. E könyv első kiadásakor csak a DOS-emulátor létezett. Időközben azonban számos egyéb emulátor is hozzáférhetővé vált. Ezek közül több olyan van, amely nemcsak a Linux, hanem egyéb, szabadon és a kereskedelemben terjesztett UNIX változat (pl. Free BSD) esetén is rendelkezésre áll.

## 4.1 DOS EMULÁTOR

A OS/2 és az egyéb újabb operációs rendszerekhez hasonlóan a Linux is rendelkezik DOS-emulátorral, amely lehetővé teszi DOS programok szimultán futtatását más Linux alkalmazásokkal. Nem emulálja a DOS operációs rendszert, csak az alapvető adatkiviteli/adatbeviteli rutinokat (BIOS), így az összes fontos eszközökhoz hozzáférést biztosít.

### Áttekintés

Bár az emulátor nem teszi lehetővé az összes DOS-os program problémamentes futtatását, a lehetőségek azonban meglehetősen nagyok. A programok futhatnak grafikus módban a felső memória (HMA), a felső memoriablokkok (UMB) és a kibővített memória (EMS) igénybevételével. A DOS védett üzemmódú interfészé (DPMI), és az emulátoron belül a Novell szerverek elérése ugyancsak lehetséges, azonban ez a rész még fejlesztés alatt van. Az olyan DOS-os programok, mint a Norton Commander vagy a QEdit szövegszerkesztő problémamentesen használható, de még az olyan nagyobb, és a kereskedelemben terjesztett termékek is, mint a Turbo Pascal vagy a WordPerfect (már rendelkezésre áll olyan DOS-emulátor, amely a Borland C++ 3.1-et DPMI-vel képes futtatni – a lektor).



4.1 ábra. DOS-emulátor futtatása az X11 alatt.

Az **XII** alatt az emulátor önálló ablakban vagy szöveges üzemmódban virtuális konzolban futtatható, de az emulátoron belül a konzolok közötti átváltás is lehetséges. Az XII-hez hasonlóan kell itt is eljárni, vagyis egyidejűleg kell lenyomni a <Ctrl> és <Alt> billentyűket, valamint a megfelelő funkcióbillentyűt.

A normál lemezen és DOS partíciókon kívül a DOS-emulátor a Linux fájlrendszerét is el tudja érni, valamint a lemez **image** fájlokat is. A DOS programok ezeket a fájlokat igazi adathordozó eszközöként kezelik, és elsősorban a DOS-emulátor indításához használják.

A DOS-emulátor konfigurálásához az */etc/dosemu.config* fájlt kell megfelelően módosítani. Egy további fájtra, az */etc/dosemu.users* nevű fájtra is szükség van, amely az összes felhasználó nevét és az emulátor használatára vonatkozó engedélyeket tartalmazza.

A lemez **image** fájlok rendszerint a */usr/lib/dosemu* vagy a */var/lib/dosemu* könyvtárakban vannak.

A DOS-emulátor, ahogy ezt már korábban is említettük, nem emulálja a teljes DOS operációs rendszert, csak a PC **hardverét** és **BIOS-át**. Így az emulátor elindításához DOS betöltőlemezre, a merevlemezen egy olyan partióra, amelyre a DOS telepítve van, vagy a megfelelő **image** fájtra van szükség.

A DOS betöltésének egyik lehetősége a DOS partió elérése. Ebben az esetben azonban a partió logikailag nem lehet beilleszteni a Linux fájlrendszerébe, mivel ez összeütközéshez, adatvesztéshez vezethet.

További lehetőség a **rendszerlemezről** való betöltés vagy speciális **image** fájl használata, amelyet a DOS-emulátor valódi lemezént használ. Betöltőlemez helyett az **image** fájl használata eleágás és gyors, azonban ehhez először a fájlt létre kell hozni.

### **Image fájl létrehozása a rendszer behúzásához**

A rendszer betöltésére alkalmas **image** fájl készítéséhez egy normál DOS boot lemezre van szükség. Azonban erre a lemezre számos olyan speciális DOS programot is rá kell másolni, amely az emulátorhoz hozzáartozik. Ezek a programok teszik lehetővé a Linux fájlrendszeréhez való hozzáférést és biztosítják az emulátorra vonatkozó bizonyos beállítások olvasását és módosítását. A lemezen egy egyszerű szerkesztőprogramot is érdemes elhelyezni.

A következő képernyőképen a DOS-emulátorhoz tartozó alkonyvtárak kezelő- és egyéb programjainak neve látható:

```
nicetry:~/work/dosemu/0.63.1.66/commands# ls
autoexec.bat compile.bat emufs.sys getcwd.exe system.com
bootoff.com config.sys emumouse.com isemu.com system.exe
booton.com dosdbg.com emumouse.exe lancheck.exe unix.com
cdrom.sys dumpconf.exe exitemu.com lredir.com vgaoff.com
chdir.com eject.com fossil.com mgarrot.com vgaon.com
chdir.exe ems.sys getcwd.com speed.com
nicetry:~/work/dosemu/0.63.1.66/commands#
```

Ha ezek a fájlok nem állnak rendelkezésre, akkor az emulátor teljes disztribúciója lemásolható egy **FTP** szerverről. Például a [sunsite.unc.edu](http://sunsite.unc.edu) szerveren a DOS-emulátor disztribúció a [/pub/Linux/system/emulators/dosemu](http://pub/Linux/system/emulators/dosemu) könyvtárból van. Az itt található fájlokat az **MTools** parancssal (lásd a 3.3 szakasz) lehet a lemezre másolni.

Az összes program és fájl behúzólemezre másolása után a lemez tartalma a **dd** parancssal egy fájlba másolható, ahogyan ezt a következő képernyőkép is mutatja. Ez a fájl bármelyik könyvtárban elhelyezhető. Az alábbi példában ez a könyvtár a **bdisk**. Ha a DOS-emulátort csak egy személy

szeretné használni, akkor a fájl a megfelelő home-könyvtárban is elhelyezhető, azonban ha a hálózatban több felhasználónak is szüksége lesz rá, akkor erre a céllra egy rendszer-könyvtárat kell létrehozni, például /var/lib/dosemu.

```
nicetry:-5 dd if=/dev/fd0 of=bdisk bs=64k
```

Ezután a DOS-emulátorra vonatkozó konfigurációs fájlt úgy kell módosítani, hogy a rendszer betölteséhez ezt a fájlt lehessen használni. A konfigurációs fájlban az erre vonatkozó bejegyzések a következők:

```
bootC # Startup drive valid values: bootA bootC  
bootdisk heads 2 sectors 18 tracks 80 threeinch file  
/var/lib/dosemu/bdisk
```

Ha az image fájlt nem az aktuális könyvtár tartalmazza, a konfigurációs fájlból a hozzá vezető teljes útvonalat elő kell írni.

Ezek után az DOS emulátor a dos parancs végrehajtásával már elmeletileg elindítható. A praktikus használathoz azonban még végre kell hajtani néhány beállítást.

### A DOS particiók elérése

Ha az előzőekben leírtaknak megfelelően a betöltő image fájl létrejött, így a DOS-emulátor már elindítható, azonban ez az állapot nem biztosít hozzáférést a merevlemez Linux fájlrendszeréhez vagy DOS particiójához.

Az emulátorból a Linux fájlrendszer eléréséhez az emuufs.sys kezelőprogramra illetve az lredir programra van szükség. Az emuufs.sys kezelőprogram a Linux fájlrendszer bármelyik könyvtárát a közvetkező szabad meghajtó betűjeléhez tudja hozzárendelni. Az lredir program például a következőképpen futtatható:

```
c:\linux\fs\lredir
```

A példában az első paraméter a c:, amely a kicsérélendő betűjelét jelenti. A második paraméter azonosítja a forrást, amely a jelen esetben a Linux fájlrendszer gyökérkönyvtára. A \linux\fs képviseli tehát a Linux fájlrendszert és a \ adja meg a fájlrendszeren belül az útvonalat. Az lredir.readme fájl, amely a DOS-emulátor distribúció, ugyanabban a könyvtárban található meg, mint maga az lredir program, s a programhoz használható összes paraméter leírását részletesen tartalmazza.

A rendszer betöltése a lemez image-ról más módon is megvalósítható, azonban ezek általában negatív következményekkel járnak a hajlékonysomelez- vagy egyéb meghajtók következő hozzáférésre vonatkozólag. A továbbiakban ezért ezeket a lehetőségeket is összefoglaljuk.

Tegyük fel, hogy a merevlemez tartalmaz egy DOS particiót, amely a /dos c logikai néven illeszkedik a rendszerbe. A DOS emulátorból ezt a particiót az alábbiak szerint lehet elérni:

- Betöltéskor lemezről, majd a config.sys állományból az emuufs.sys kezelőprogram, illetve az autoexec.bat fájlból az lredir program elindítása. Ez a művelet a /dos c:. Mivel az emulátor lemezről való betöltése elég lassú, ezt a módszert ritkán, általában csak a telepítéshez használják.

- Betöltés a merevlemezen létesített image fájlból az `emuefs.sys` illetve az `lreditr` felhasználásával a C: meghajtó eléréséhez. Ebben az esetben a merevlemezen az image fájl a C: meghajtót használja.
- Betöltés a merevlemezen létesített image fájlból és az `autoexec.bat` fájlból futtattott `lreditr` használatával a C: betűvel kicsérélés. Ez lehetővé teszi a DOS partició C: meghajtót kérőtől való elérését és a hajlékonylemez-meghajtók érintetlenül maradhatnak. Ez az elrendezés ugyanazt a környezetet biztosítja, mintha a rendszert közvetlenül a DOS-ból húznánk be.
- Ezzel a módszerrel az a probléma, hogy a merevlemez, amelyről az `autoexec.bat` fájl beolvasásra kerül, egy másikkal cserélődik ki a fájl beolvasása és végrehajtása alatt. Ebből következően ennek a fájlnak a merevlemezen létesített image fájlból és a valós merevlemez-partícióban azonosnak kell lennie.
- A lemez image fájl aktiválása virtuális A: meghajtótól és a `config.sys` vagy az `autoexec.bat` fájlok előzőekben leírtak szerinti használata. Ennek az elrendezésnek az az előnye, hogy a DOS-emulátor elindításához nincs szükség lemezre. A hátránya viszont az, hogy a lemezmeghajtót nem lehet a továbbiakban A: meghajtótól használni, mivel ezt a betűt már a lemez image fájl lefoglalja.
- A DOS-emulátor konfigurációs fájlijában a `bootdisk` speciális opció felhasználásával betöltés a lemez image fájlból, majd ezen image fájl kikapcsolása az `autoexec.bat` fájl végén. Ez a legelegánsabb módszer, mivel a betöltés normál környezetet teremt image fájlok nélkül, ugyanakkor a betöltés mégis image fájlból megvégzésre.

**Fontos megjegyzés!** Elméletileg lehetséges, hogy a merevlemezen levő DOS partició a DOS-emulátoron keresztül közvetlenül, illetve a Linux fájlrendszeren keresztül közvetetten is elérhető legyen. Azonban ezt a kettős hozzáférést az adatvesztés elkerülése érdekében nem szabad alkalmazni. A DOS partició biztonságos elérését a Linux fájlrendszeren keresztül az `emuefs.sys` vagy a `lreditr` programokkal kell megvalósítani, vagyis az emulátornak a DOS particiót közvetett módon kell elérnie.

## Konfigurálás

A következő példa a DOS-emulátorhoz tartozó konfigurációs fájlt mutatja be, a `bootdisk` opciót az előzőekben leírtak szerint használva. A példa önálló logikai egységekből épül fel. A # karakterrel kezdődő sorok megjegyzések.

```

debug config off disk off warning off hardware off
port off read off general off IPC off
video off write off xms off ems off
serial off keyb off dpmi off
printer off mouse off

dosbanner on
timint on
keyboard layout us keybint on rawkeyboard on
HogThreshold 10
ipxsupport off
terminal charset latin updatefreq 4 color on
X updatefreq 8 title "DOS in a BOX" icon_name "xdos"
video vga console graphics chipset s3 memsize 2048
mathco on # Math coprocessor valid values: on off

```

```

cpu 80486 # CPU emulation valid values: 80286 80386 80486
bootC # Startup drive valid values: bootA bootC
dpmi off # DPMI size in K, or "off"
xms 1024 # XMS size in K, or "off"
ems 1024 # EMS size in K, or "off"
# maps 0xc8000..0xcffff and 0xcc000..0xcffff
sillyint off # this disables IRQ monitoring
speaker native # or "off" or "emulated"
disk partition "/dev/hdal" # 1st partition on 1st IDE.
EmuSys ENU
EmuBat ENU
floppy device /dev/fd0 threeinch

```

A következő hibakereső beállítások csak a fejlesztők számára érdekesek. A szokásos használat során az összes hibakereső beállítást off-ra kell állítani.

```

debug config off disk off warning off hardware off
port off read off general off IPC off
video off write off xms off ems off
serial off keyb off dpmi off
printer off mouse off

```

A dosemu indítóüzenet és az időzítő megszakítás (timer interrupt) be és kikapcsolt állapotban lehet:

```

***** MISCELLANEOUS
*****
#
# Want startup DOSEMU banner messages? Of course :-
dosbanner on
#
# timint is necessary for many programs to work.
timint on

```

A billentyűzetet ugyancsak definiálni kell:

```

***** KEYBOARD
*****
#
# QuickStart:
# With the "layout" keyword, you can specify your
# country's keyboard layout. The following layouts
# are implemented:
# finnish us dvorak sf
# finnish-latin1 uk sg sf-latin1
# de dk sg-latin1 es

```

```

#      de-latin1      dk-latin1  fr      es-latin1
#      be            no       fr-latin1 portuguese
#      it            sw
# The us-layout is selected by default if the "layout"
# keyword is omitted.

#
# The keyword "keybint" allows more accurate of keyboard
# interrupts, It is a bit unstable, but makes keyboard
# work better when set to "on".

#
# The keyword "rawkeyboard" allows for accurate keyboard
# emulation for DOS programs, and is only activated when
# DOSEMU starts up at the console. It only becomes a
# problem when DOSEMU prematurely exits with a
# "Segmentation Fault" fatal error, because the keyboard
# would have not been reset properly. In that case, you
# would have to run kbd_mode -a remotely, or use the
# RESET button. In reality, this should never happen. But
# if it does, please do report to the dosemu development
# team, of the problem and detailed circumstances, we're
# trying our best! If you don't need near complete
# keyboard emulation (needed by major software package),
# set it to "off"

#
# keyboard layout us keybint on rawkeyboard on
# keyboard layout de-latin1 keybint on rawkeyboard on
#
# The HogThreshold value determines how nice Dosemu will
# be about giving other Linux processes a chance to run.
# Setting the HogThreshold value to approximately half of
# you BogoMips value will slightly degrade Dosemu
# performance, but significantly increase overall system
# idle time. A zero value runs Dosemu at full tilt.
#
HogThreshold 10

```

A soros interfészre, a soros egérré és a modemre vonatkozó beállítások nem kötelezőek. Szokásos konfigurálás esetén – amikor az emulátor X11 alatt fut és a modemre vonatkozó programok pedig közvetlenül a Linux alatt – ezek a beállítások figyelmen kívül hagyhatók. Az egér X11 alatti használatához a DOS emulátor konfigurációs fájljában az egeret nem szabad előírni.

```

***** SERIAL *****
#
# QuickStart:
# You can specify up to 4 simultaneous serial ports here.
# If more than one ports have the same IRQ, only one of those ports
# can be used at the same time. Also, you can specify the com port,

```

```

# base address, irq, and device path! The defaults are:
# COM1 default is base 0x03F8, irq 4, and device /dev/cua0
# COM2 default is base 0x02F8, irq 3, and device /dev/cua1
# COM3 default is base 0x03E8, irq 4, and device /dev/cua2
# COM4 default is base 0x02E8, irq 3, and device /dev/cua3
# If the "com" keyword is omitted, the next unused COM port is assigned.
# Also, remember, these are only how you want the ports to be emulated
# in DOSEMU. That means what is COM3 on IRQ 5 in real DOS, can become
# COM1 on IRQ 4 in DOSEMU!
#
# Also, as an example of defaults, these two lines are functionally equal:
# serial com 1 mouse
# serial com 1 mouse base 0x03F8 irq 4 device /dev/cua0
#
# If you want to use a serial mouse with DOSEMU, the "mouse" keyword
# should be specified in only one of the serial lines. (For PS/2
# mice, it is not necessary, and device path is in mouse line instead)
#
# Uncomment/modify any of the following if you want to support a modem:
# (or any other serial device.)
#serial com 1 device /dev/modem
#serial com 2 device /dev/modem
#serial com 3 device /dev/modem
#serial com 4 device /dev/modem
#serial com 3 base 0x03E8 irq 5 device /dev/cua2
#
# If you are going to load a msdos mouse driver for mouse support
# uncomment/modify one of the following.
#serial mouse com 1 device /dev/mouse
#serial mouse com 2 device /dev/mouse
#
# What type is your mouse? Uncomment one of the following.
# Use the 'internaldriver' option to try Dosemu internaldriver.
# Use the 'emulate3buttons' for 3button mice.
#mouse microsoft
#mouse logitech
#mouse mmseries
#mouse mouseman
#mouse hitachi
#mouse mousesystems
#mouse busmouse
#mouse ps2 device /dev/mouse internaldriver emulate3buttons
#mouse mousesystems device /dev/mouse internaldriver cleardtr
#
# For tty locking capabilities:
# ttylocks directory /var/locks namestub LCK.. [binary]

```

A DOS-emulátor új verziói távoli módon is működtethetők. Az adatkivetel a normál xterm-en vagy a color-xterm-en keresztül valósítható meg. A terminál típusát és a karakterkészletet azonban definiálni kell.

```
***** TERMINALS *****
*****
# This section applies whenever you run DOSEMU remotely or in an xterm.
# Color terminal support is now built into DOSEMU. Skip this section for
# now to use terminal defaults, until you get DOSEMU to work.
#
# QuickStart:
#     There is a number of keywords for the terminal configuration line.
#     "charset" latin, ibm                               (default latin)
#         Select the character set to use with DOSEMU.
#     "color" off, on                                     (default on)
#         Enable or disable color terminal support.
#     "updatefreq" value                                (default 4)
#         A number indicating the frequency of terminal updates of the screen.
#         The smaller the number, the more frequent. A value of 20 gives a
#         frequency of about one per second, which is very slow. However, more
#         CPU time is given to DOS applications when updates are less
#         frequent.
#         A value of 4 is recommended in most cases, but if you have a fast
#         system or link, you can decrease this to 0.
#     "escchar" value                                 (default 30)
#         A number that specifies the control character used as a prefix
#         character for sending alt, shift, ctrl, and function keycodes. The
#         default value is 30 which is Ctrl-^. So, for example, F1 is
#         'Ctrl-^ 1', Alt-F7 is 'Ctrl-^ a Ctrl-^ 7'. For online help, press
#         'Ctrl-^ h'.
#
#     Use the following to enable the IBM character set.
#terminal charset ibm color on
#
#     Use this for color xterms or rxvt's with no IBM font, with only 8 colors.
#terminal charset latin color on
#
#     Use this for color xterms or rxvt's with IBM font, with only 8 colors.
#terminal charset ibm color on
#
#     More detailed line for user configuration:
```

Ha a DOS-emulátor futtatására X11 alatt kerül sor, akkor a képernyő frissítésének frekvenciáját be kell állítani, továbbá az olyan X11 opciókat, mint az ablak vagy az ikon neve is definiálni szükséges.

```

***** X SUPPORT *****

# valid keywords for the X config line:
# "updatefreq" value (default 8)
#   A number indicating the frequency of X updates of the screen.
#   The smaller the number, the more frequent. A value of 20 gives a
#   frequency of about one per second, which is very slow. However, more
#   CPU time is given to DOS applications when updates are less
#   frequent.

# "display" string (default ":0")
#   The X server to use. If this is not specified, dosemu will use
#   the DISPLAY environment variable. (This is the normal case)

# "title" string (default "dosemu")
#   What you want dosemu to display in the title bar of its window.

# "icon_name" string (default "dosemu")
#   Used when the dosemu window is iconified.

# "keycode" (default 0)
#   Used to give Xdos access to keycode part of XFree86.

# "blinkrate" value (default 8)
#   Used to add blinking to cursor.

# "font" value (default vga)
#   Used to pick a font other than vga. Must be monospaced.

```

A DOS-emulátor konzolon való futtatásához a videóadapter típusát be kell írni. Ez a beállítás biztosítja, hogy a videóadAPTER előző grafikus programok megfelelően működjeneK. A helytelen beállítások fekete/fehér képernyőt eredményeznek és a képernyő akkor is ilyen marad, ha az emulátor működése befejeződik, vagy virtuális konzolra történik átkapcsolás. Ebben az esetben csak egyet lehet tenni: újraindítani a számítógépet. A művelet végrehajtásához egyszerűen át kell kapcsolni egy másik virtuális konzolra, és a rendszert a shutdown -r now parancssal (lásd 7.3. szakasz) lezární.

```

***** VIDEO *****

\# !WARNING!!: A LOT OF THIS VIDEO CODE IS ALPHA! IF YOU ENABLE GRAPHICS
\# ON AN INCOMPATIBLE ADAPTOR, YOU COULD GET A BLANK SCREEN OR MESSY SCREEN
\# EVEN AFTER EXITING DOSEMU. JUST REBOOT (BLINDLY) AND THEN MODIFY CONFIG.
\# QuickStart:

```

```

\# Start with only text video using the following line, to get started.
\# then when DOSEMU is running, you can set up a better video
\# configuration.
\#
\# video { vga }           # Use this line, if you are using VGA
\#video { cga console }      # Use this line, if you are using CGA
\#video { ega console }      # Use this line, if you are using EGA
\#video { mda console }      # Use this line, if you are using MDA
\#
\# QuickStart Notes for Graphics:
\# - If your VGA-Bios resides at E000-EFFF, turn off video BIOS shadow
\#   for this address range and add the statement vbios_seg 0xe000
\#   to the correct vios-statement, see the example below
\# - If your VBios size is only 32K you set it with vbios_size 0x8000,
\#   you then gain some space for UMB or hardware ram locations.
\# - Set "allowvideoportaccess on" earlier in this configuration file
\#   if DOSEMU won't boot properly, such as hanging with a blank screen,
\#   beeping, leaving Linux video in a bad state, or the video card
\#   bootup message seems to stick.
\# - Video BIOS shadowing (in your CMOS setup) at C000-CFFF must be disabled.
\#
\#      *> CAUTION <*: TURN OFF VIDEO BIOS SHADOWING BEFORE ENABLING
\#      GRAPHICS!
\#
\#                           This is not always necessary, but a word to the wise
\#                           shall be sufficient.
\#
\# - If you have a dual-monitor configuration (e.g. MDA as second
\#   display),
\#   you then may run CAD programs on 2 displays or let play your debugger
\#   on the MDA while debugging a graphics program on the VGA (e.g TD -do).
\#   You also may switch to the MDA display by using the DOS command
\#   mode mono (mode co80 returns to your normal display).
\#   This feature can be enabled by the switch "dualmon" like this:
\#       video { vga console graphics dualmon }
\# and can be used on a xterm and the console, but of course not, if you
\# have the MDA as your primary display.
\# You also must set USE_DUALMON 1 in include/video.h.
\# NOTE: Make sure no more than one process is using this feature !
\#        (you will get funny garbage on your MDA display.)
\# Also, you must NOT have the dualmon-patches for kernel applied
\#        (having the MDA as Linux console)
\#
\# It may be necessary to set this to "on" if DOSEMU can't boot up properly
\# on your system when it's set "off" and when graphics are enabled.
\# Note: May interfere with serial ports when using certain video boards.
\#allowvideoportaccess on
\#
\# Any 100% compatible standard VGA card _MAY_ work with this:
\#video { vga yconsole graphics }

```

```
\#video { vga console graphics vbios_seg 0xe000 }
\#
\# Trident SVGA with 1 megabyte on board
\#video { vga console graphics chipset trident memsize 1024 }
\#
\# Diamond SVGA
\#video { vga console graphics chipset diamond }
\#
\# ET4000 SVGA card with 1 megabyte on board:
\#video { vga console graphics chipset et4000 memsize 1024 }
\# or
\#video { vga console graphics chipset et4000 memsize 1024 vbios_size
0x8000
\#
\# S3-based SVGA video card with 1 megabyte on board:
\#video { vga console graphics chipset s3 memsize 2048 }
\# For ATI graphic mode
\#ports { 0x1ce 0x1cf 0x238 0x23b 0x23c 0x23f 0x9ae8 0x9ae9 0x9aee 0x9aef }
```

A tövábbi beállítások arról tájékoztatják a DOS-t, hogy melyik processzor és társprocesszor telepítésére kerül sor, továbbá a betöltőlemez beviteljét állítja be A-ra vagy C-re. A jelenlegi példa a BootA-t alkalmazza.

```
***** MISCELLANEOUS
*****
#
# QuickStart:
# For "mathco", set this to "on" to enable the coprocessor during DOSEMU.
# This really only has an effect on kernels prior to 1.0.3.
# For "cpu", set this to the CPU you want recognized during DOSEMU.
# For "bootA"/"bootC", set this to the bootup drive you want to use.
# It's strongly recommended you start with "bootA" to get DOSEMU
# going, and during configuration of DOSEMU to recognize hard disks.
#
mathco on          # Math coprocessor valid values: on off
cpu 80486         # CPU emulation valid values: 80286 80386 80486
bootC             # Startup drive valid values: bootA bootc *
```

Az emulátoron belüli memóriakezelés részletesen konfigurálható. Az XMS, EMS és a DPMI, valamint az adapterekre vonatkozó RAM a méretével definíálható, illetve kikapcsolható.

```
***** MEMORY *****

#
# QuickStart:
#   These are memory parameters, stated in number of kilobytes.
#   If you get lots of disk swapping while DOSEMU runs, you should
#   reduce these values. Also, DPMI is still somewhat unstable,
#   (as of early April 1994) so be careful with DPMI parameters.
#
#   For ems, you now can set the .frame to any 16K between 0xc800..0xe000
#
#   If you have adapters, which have memory mapped IO, you may now
#   map those regions with hardware_ram ... You can only map in
#   entities of 4k, you give the address, not the segment.
#
#   umb_max is a new parameter which tells DOSEMU to be more aggressive
#   about finding upper memory blocks. The default is 'off'.
#
#umb_max on          # be more aggressive about finding XMS UMB blocks

dpmi off            # DPMI size in K, or "off"
xms 1024            # XMS size in K, or "off"
ems 1024            # EMS size in K, or "off"
#ems ems_size 1024 ems_frame 0xe000
#ems ems_size 2048 ems_frame 0xd000

hardware_ram 0xc8000 range 0xcc000 0xffff
                  # maps 0xc8000..0xc8fff and 0xcc000..0xffff

dosmem 640         # Maximum conventional RAM to show apps
```

A megszakítások és az I/O portok kezelése kritikus pont. A beállítási lehetőségek elsősorban azon programfejlesztők számára szólnak, akik ismerik a DOS belső működését. A rendszert csak felhasználói szinten ismerők lehetőleg ne változtassanak ezeken a beállításokon.

```
***** IRQ *****

#
# QuickStart:
#   These are parameters needed for SIG, the Silly Interrupt Generator.
#   To use this feature, you also must have the emumodule.o driver
#   loaded. For more details see emumod/README.emumod.
#
#   The sillyint statement accepts IRQ values between 3..15,
#   if using the ... syntax each value or range can be prefixed
#   by the keyword use_sigio to monitor the IRQ via SIGIO.
#   If this is missing the IRQ is monitored by SIGALRM.
```

```
sillyint off      # this disables IRQ monitoring
#sillyint 15
#sillyint 15
#sillyint use_sigio 15
#sillyint 10 use_sigio range 3-5
```

A PC hangszóróinak elérése vagy közvetlenül, vagy hangjelzésekkel konvertálva lehetséges.

```
***** SPEAKER *****

#
# These keywords are allowable on the "speaker" line:
#
# native      Enable DOSEMU direct access to the speaker ports.
# emulated    Enable simple beeps at the terminal.
# off         Disable speaker emulation.
#
# speaker native      # or "off" or "emulated"
```

A merevlemezek és azok particióinak közvetlen elérése nem tanácsos. Azonban ha erre valamilyen oknál fogva mégis feltétlenül szükség van, például egy olyan particióhoz kell hozzáférni, amelyet a stacker programmal tömörítettek, akkor ezt a hozzáférést itt lehet engedélyezni.

```
***** HARD DISKS *****

#
# !WARNING!!: DAMAGE MIGHT RESULT TO YOUR HARD DISK (LINUX AND/OR DOS)
# IF YOU FIDDLE WITH THIS SECTION WITHOUT KNOWING WHAT YOU'RE DOING!
#
# QuickStart:
#
# The best way to get started is to start with a boot floppy, and set
# "bootA" above in the configuration. Keep using the boot floppy
# while you are setting this hard disk configuration up for DOSEMU,
# and testing by using DIR C: or something like that.
#
# If you want DOSEMU to be able to access a DOS partition, the
# safer type of access is ".partition" access, because "wholedisk"
# access gives DOSEMU write access to a whole physical disk,
# including any vulnerable Linux partitions on that drive!
#
# !!! IMPORTANT !!!
#
# You must not have LILO installed on the partition for dosemu to boot off.
# As of 04/25/94, doublespace and stacker 3.1 will work with wholedisk
# or partition only access. Stackter 4.0 has been reported to work with
# wholedisk access.
#
# Please read the documentation in the "doc" subdirectory for info
# on how to set up access to real hard disk.
```

```

#
# "image" specifies a hard disk image file.
# "partition" specifies partition access, with device and partition number.
# "wholdisk" specifies full access to entire hard drive.
# "readonly" for read only access. A good idea to set up with.
# "bootfile" to specify an image of a boot sector to boot from.
#
#disk image "/var/lib/dosemu/hdimage"      # use diskimage file.
disk partition "/dev/hdal"          # 1st partition on 1st IDE.
# disk partition "/dev/sdal"          # 1st partition on 1st IDE.
#disk partition "/dev/hdal" bootfile "/var/lib/bootsect.dos"
                                         # 1st partition on 1st IDE
                                         # booting from the specified
                                         # file.
#disk partition "/dev/hda6" readonly      # 6th logical partition.
#disk partition "/dev/sdbl" readonly      # 1st partition on 2nd SCSI.
#disk wholdisk "/dev/hda"                # Entire disk drive unit

```

A következő beállítás azt frja elő, hogy a rendszer betöltése lemez image fájlból menjen végbe.

```

***** DOSEMU BOOT *****
#
# Use the following option to boot from the specified file, and then
# once booted, have bootoff execute in autoexec.bat. Thanks Ted :-).
# Notice it follows a typical floppy spec. To create this file use
# dd if=/dev/fd0 of=/var/lib/dosemu/bdisk bs=16k
#
#bootdisk heads 2 sectors 18 tracks 80 threeinch file /var/lib/dosemu/bdisk
#
# Specify extensions for the CONFIG and AUTOEXEC files. If the below
# are uncommented, the extensions become CONFIG.EMU and AUTOEXEC.EMU.
# NOTE: this feature may affect file naming even after boot time.
# If you use MSDOS 6+, you may want to use a CONFIG.SYS menu instead.
#
EmuSys EMU
EmuBat EMU

```

A hajlékonylemez-meghajtókat is definiálni kell, hogy a hozzáférés lehetséges legyen.

```

***** FLOPPY DISKS *****
#
# QuickStart:
#   This part is fairly easy. Make sure that the first (/dev/fd0) and

```

```

# second (/dev/fd1) floppy drives are of the correct size, "threeinch"
# and/or "fiveinch". A floppy disk image can be used instead, however.
#
# FOR SAFETY, UNMOUNT ALL FLOPPY DRIVES FROM YOUR FILESYSTEM BEFORE
# STARTING UP DOSEMU! DAMAGE TO THE FLOPPY MAY RESULT OTHERWISE!
#
# floppy device /dev/fd0 threeinch
# floppy device /dev/fd1 fiveinch
#floppy heads 2 sectors 18 tracks 80
#      threeinch file /var/lib/dosemu/diskimage
#
# If floppy disk speed is very important, uncomment the following
# line. However, this makes the floppy drive a bit unstable. This
# is best used if the floppies are write-protected.
# Use an integer value to set the time between floppy updates.
#
#FastFloppy 8

```

A nyomtatóhoz való hozzáférés a DOS-emulátorból a megfelelő eszközre, de még egy Linux parancshoz is átirányítható.

```

***** PRINTERS *****

#
# QuickStart:
# Printer is emulated by piping printer data to a file or via a unix
# command such as "lpr". Don't bother fiddling with this configuration
# until you've got DOSEMU up and running already.

#
# NOTE: Printers are assigned to LPT1:, LPT2:, and LPT3: on a one for
# one basis with each line below. The first printer line is assigned
# to LPT1:, second to LPT2:, and third to LPT3:. If you do not specify
# a base port, the emulator will setup the bios to report 0x378, 0x278,
# and 0x3bc for LPT1:, LPT2:, and LPT3: respectively.

#
# To use standard unix lpr command for printing use this line:
#printer options "-s" command "lpr" timeout 20
#
# And for any special options like using pr to format files,
# add it to the options parameter:
#printer options "-p %s" command "lpr" timeout 10      # pr format it
#
# To just have your printer output end up in a file, use the following line:
#printer file "lpt3"

```

```

# If you have a DOS application that is looking to access the printer
# port directly, and uses the bios LPT: setting to find out the port to use,
# you can modify the base port the bios will report with the following:
#
#printer options "%s" command "lpr" base 0x3bc
#
# Be sure to also add a port line to allow the application access to
# the port:
#
#ports 0x3bc 0x3bd 0x3be
#
# NOTE: applications that require this will not interfere with applications
# that continue to use the standard bios calls. These applications will
# continue to send the output piped to the file or unix command.
#

```

A betöltőlemez image-ben használt autoexec.bat fájl tartalma a következő lehet:

```

@ECHO OFF
PROMPT $p$g
PATH c:\dosemu;c:\dosemu\win;C:\wfw;c:\wfw\system;C:\DOS;c:\nc
SET TEMP=C:\DOS
lredit d: linux\fs\dosd
lredit e: linux\fs\dose

```

Az autoexec.bat fájl futtatása után a behúzó fájl már nem érhető el, mivel a bootoff parancs letiltja a lemez image-t, és a hozzáférést valós lemezmeghajtóhoz állítja vissza. Ha a későbbiek során a betöltési konfiguráció módosítása válik szükséges, akkor a betöltő fájl engedélyezése a booton parancssal végezhető el.

Az autoexec.bat fájlból érdemes megfigyelni, hogy a !blank sorok nem tartalmaznak üres sorokat a bootoff parancs után. Mivel a továbbiakban az autoexec.bat fájl már nem létezik, ez egy hibaüzenetet eredményezhet.

### **Novell szerverek (kiszolgálók) elérése**

A DOS-emulátor Novell szervereket is el tud érni. Mivel azonban ez a terület még fejlesztés alatt van, az eljárás maga is gyorsan változik. Az emulátor alapvetően úgy is konfigurálható, hogy már eleve biztosítson egy IPX interfész, de a pdipx kezelőprogram adaptált verziója is elindítható a DOS-emulátorból. Mindkét esetben egy normál netx shellt indít el. Ez teremti meg a kapcsolatot a szerverrel és új virtuális Novell meghajtó biztosít. Ezután átvált a Novell meghajtóról és elindítja a szerver login parancsát.

Az eljárásról szóló konkrét és aktuális információ az adott DOS-emulátorcsomag readme fájljában található meg.

### **Indítás és befejezés**

A DOS-emulátor a dos vagy az xdos parancsokkal indítható el. Az xdos egy olyan dos-ra utal, amely az emulátor X11-es verzióját indítja el. A dos parancsot a -x paraméterrel elindítva ugyanez

az eredmény érhető el. A többi lehetséges opcióval a DOS-emulátor konfigurációs fájljában lehet felülírást végezni. Ezeket az opciókat a `dos -help` parancssal lehet megjeleníteni.

A DOS-emulátor működésének befejezéséhez az `exitemu` programot kell elindítani vagy másik virtuális konzolról a `kill` parancsot végreghajtani.

## 4.2 WINE

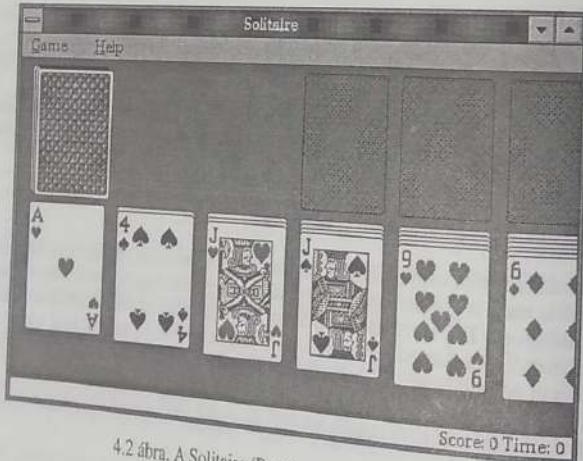
Egy másik fejlesztés, amely várhatóan nagy hatással lesz a PC-s UNIX rendszerek elterjedésében, a Windows Application Binary Interface (WABI).

A interfész a Sun Microsystems fejlesztette ki és számos UNIX gyártó megvásárolta a felhasználás jogát, így ez várhatóan az összes ismert platformon rendelkezésre áll majd. Ez a interfész a Microsoft Windows programok futtatását teszi lehetővé közvetlenül a UNIX-ból. Az interfész a Windows API hívásait megfelelő X11 és operációs rendszerbeli hívásokká konvertálja, amelynek érdekes mellékhatása az, hogy az ilyen programok más X-szervereken is használhatóvá válnak, az X11 hálózati átlátszósága miatt.

1993 júniusának elején a Linux DOS-emulátorával foglalkozó levelezési listára olyan levelezések kerültek, amelyek a WABI Linuxra készített új fejlesztésről szóltak. Néhány napon belül önellő levelezési listát állítottak fel kifejezetten erre a témaéra, és a konkrét fejlesztés is elkezdődött.

Elsőször két viszonylag független fejlesztés létezett. Az egyik csoport az MS Windows programok betöltőjét fejlesztette, míg a másik az MS Windows rendszer X11-re vonatkozó hívásainak API konverterével foglalkozott. A két csoport azonban hamarosan közös fejlesztésbe kezdett, melynek eredménye a ma hozzáférhető WINE, amelyben a kisebb Windowsos alkalmazások már sikeresen futthatók.

A WINE legújabb verziója a `tsx-11.mit.edu` FTP kiszolgáló `/pub/linux/ALPHA/Wine/development` alkönyvtárából tölthető le.



4.2 ábra. A Solitaire (Pasziański) a WINE alatt.

### 4.3 AZ IBCS2 EMULÁTOR

A fejlesztés az iBCS2 emulátor esetében tovább jutott el. Ez az emulátor a Linux alatt lehetővé teszi számos, a kereskedelemben terjesztett, Unix operációs rendszerre készített alkalmazás futtatását is, mint a WordPerfect vagy az Oracle.

Az előfeltétel csak az, hogy a program a PC UNIX által alkalmazott tárgykód-formátumok egyikében hozzáférhető legyen. A UNIX System V 3-as változata a COFF-ot használja, a 4-es az ELF-et, az SCO és az interaktív UNIX egy COFF változatot.

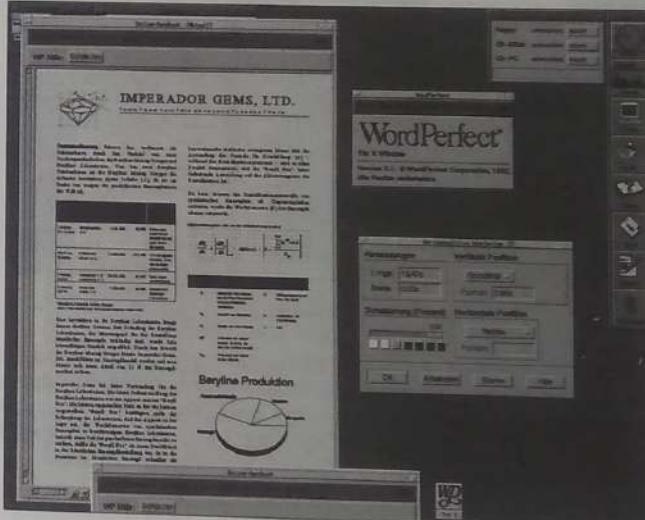
Az SCO szoftversomagok jelenleg a legdérekesebbek, mivel ezek általában statikusan vannak szerkeszve (link), így nem igényelnek további osztott tárgykód-könyvtárakat.

#### Telepítés

Mivel az iBCS2 emulátor betölthető modulként áll rendelkezésre (lásd a 3.4 szakaszt), a telepítés viszonylag egyszerű. Mindössze a forráskódban át kell váltani a kitömörített könyvtárra, és végrehajtani a make parancsot. A fordítás után új object fájl keletkezik iBCS néven, amelyet az insmod parancsval lehet betölteni:

```
nicetry:~# insmod iBCS
```

Érdemes az emulátort az egyik indító szkriptben (`rc.local`) elhelyezni, hogy a behúzás során ez is automatikusan betöltsön.



4.3 ábra. A WordPerfect 5.1 a Linux alatt futtatva

A teljes telepítés speciális kezelőfájlok létrehozását követeli meg. Érdemes megfigyelni a csatolás megváltoztatását /dev/null-ról /dev/XOR-ra.

```

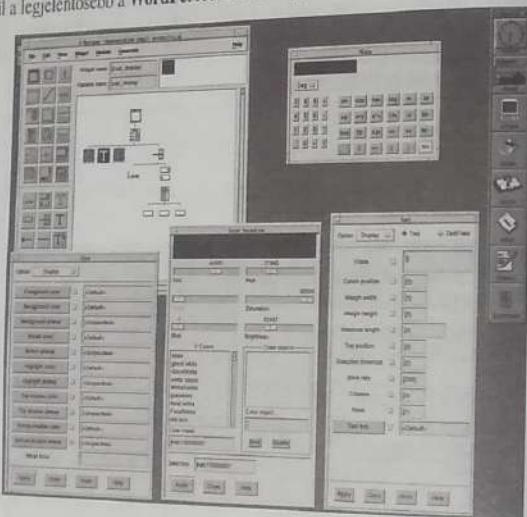
nicetry:-# mknod /dev/socksys c 30 0
nicetry:-# ln -s /dev/socksys /dev/nfsd
nicetry:-# ln -s /dev/null /dev/XOR
nicetry:-# mknod /dev/spx c 30 1

```

Ha a felhasználó engedélyel rendelkezik (megvásárolta) az SCO megosztott könyvtárakhoz, akkor ezeket a fájlokat érdemes az /shlib könyvtára másolni, mivel így lehetővé válik olyan SCO programok futtatása, amelyek nem statikus szerkesztéssel rendelkeznek.

### Alkalmazások

Az iBCS2 emulátornak köszönhetően számos érdekes SCO alkalmazás használható a Linux alatt. Ezek közül a legjelentősebb a WordPerfect 5.1-es vagy 6.0 verziója.



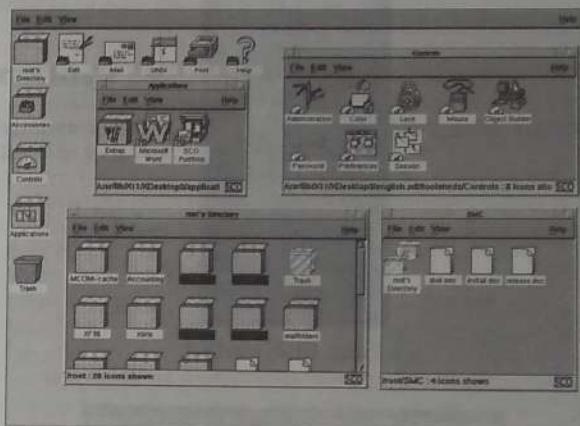
4.4 ábra. Az X-Designer

Ha a felhasználó SCO megosztott könyvtárral rendelkezik, akkor az SCO Open Desktop 3-mat és számos ehhez kapcsolódó segédprogramot is problémamentesen futthat a Linux alatt. Az SCO és a Linux alatt futtató alkalmazások teljesítményének összehasonlítva kiderül, hogy ezek jelentősen gyorsabban futnak a Linux alatt. Ennek oka a Linux kernel hatékonyságának és a korábbinál sokkal gyorsabb X-szervernek (Xfree86) köszönhető.

## 4.4 HP48 EMULÁTOR (X48)

Azok, aik szeretik a HP48 kalkulátorokat, az X48 emulátort is szeretni fogják. Ez utóbbi ugyanis minden megjelenésben, mint funkcionálisan felülmúja az eredeti. Szerzői jogi okok miatt az emulátor működtetéséhez az eredeti kalkulátor EPROM-jának tartalmára is szükség van. A külső HP48 a Linux-os számítógép soros interfészéhez csatolható. Az X48 fő előnye az eredetivel szemben a

jelentős mértékben megnövekedett RAM, és a HP48 szoftverek fejlesztésének lehetősége. A komoly alkalmazásokon kívül ez az emulátor olyan játékprogramok futtatását is lehetővé teszi, mint a *Lemmings*.



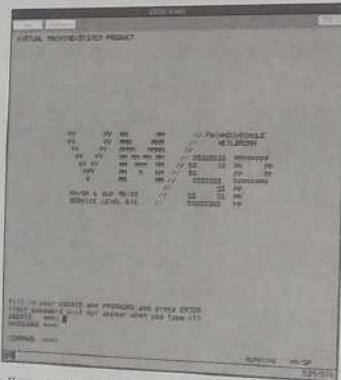
4.5 ábra. Az SCO Open Desktop 3 programja a Linux alatt futtatva



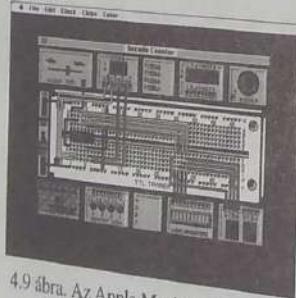
4.6 ábra. SCO ODT alkalmazások



4.7 ábra. HP48 emulátor



4.8 ábra. Az IBM 3270 emulátor az X11 alatt futtatva



4.9 ábra. Az Apple Macintosh emulátor

## 4.5 AZ IBM 3270 EMULÁTOR

Az X3270 emulátor a Linux csatlakoztatását teszi lehetővé IBM nagyszámítógépekhez. A csatlakozáshoz azonban a megfelelő nagyszámítógépen a TCP/IP verem elérhetőségére van szükség. A 3270 terminál tökéletes emulációjának érdekében speciális karakterkészleteket is telepíteni kell.

## 4.6 A MACINTOSH EMULÁTOR

Az Apple Macintosh emulátort az Abacus Research and Development cég mutatta be. A szoftvert kereskedelmi úton terjesztik.

# TELEPÍTÉS

**E**bben a fejezetben a Linux legfontosabb forrásait tárgyaljuk és a telepítését ismertetjük. Először azok a legáltalánosabb összetevők szerepelnek, amelyek majdnem minden Linux disztribúcióra vonatkoznak, majd pedig egy-egy adott telepítés részletei következnek.

## 5.1 LINUX DISZTRIBÚCIÓK

Mivel a Linux, mint szabadon hozzáérhető szoftver számos olyan összetevőből áll, amelyeket a világ különböző részein különböző emberek fejlesztenek, nem létezik olyan *hivatalos* Linux telepítőcsomag, amely a szoftverhez rendelkezésre álló összes programot tartalmazná.

Azban számos szoftverterjesztő létezik, aki saját, egyéni leg összeállított programcsomagokat ajánl. Ezek a Linux kernel, egyéb segédprogramokat, alkalmazásokat és a telepítőprogramot tartalmazzák. Az ilyen csomagot a Linux terjesztett változatának, vagy Linux **disztribúcióinak** nevezik. Vannak olyan gyártók is, akik az általuk összeállított disztribúciókat **FTP szervereken** teszik elérhetővé, illetve hajlékonylemezen vagy CD-n terjesztik bizonyos téritési díj ellenében. Az FTP szervereken hozzáérhető Linux disztribúció részei általában több könyvtárban foglalnak helyet, amelyek tartalma azonban egy-egy hajlékonylemezen egyesével elfér.

A disztribúciók mindegyike tartalmazza a kernelt és azokat a fontos segédprogramokat, amelyek a telepítéshez szükségesek, valamint a GNU C fordítóprogramját és a X11 grafikus felhasználói interfészét is.

A Linux telepítésén rendszerint menü-vezérelt telepítőprogram vezeti végig a felhasználót, biztosítva a telepítendő részegységek kiválasztását. A legtöbb telepítőprogramra az is jellemző, hogy lehetővé teszi a rendszer interaktív konfigurálását.

A hajlékonylemezekről való telepítésen kívül a legtöbb disztribúciónál a telepítés közvetlenül a CD-ről, mágnesszalagról vagy a másik számítógépről is végrehajtható. A felhasználó gyakorlatáttól és a kiválasztott programösszetevőktől függően a telepítés egy-két, vagy több óráig is eltarthat. (Egy mai „csúcsgépen” egy gyakorlott linuxos a telepítési kb. 20 perc alatt elvégzi CD-ről, míg hajlékonylemezről bizony órákat vehet igénybe – a lektor.)

### Az első disztribúciók egyike: az SLS

Az SLS (Softlanding System) volt az első disztribúciók közül az, amelyik jelentős szerepet játszott a Linux elterjedésben. Hosszú ideig ugyanis ez volt az egyetlen olyan disztribúció, amely a C fordítóprogram és a grafikus felhasználói kezelőfelület teljes telepítését lehetővé tette. Azonban az olyan disztribúciók megjelenésével, mint a Slackware az SLS csomag fontossága csökken. Ajelenlegi verzió a [tsx-11.mit.edu](http://tsx-11.mit.edu) FTP szerveren a /pub/linux/packages/SLS könyvtárban található meg.

## Az MCC Interim disztribúció

Az MCC Interim disztribúciót a University Of Manchester egyetemen állították össze. A többi disztribúcióhoz képest az MCC eléggyé lesoványított verzió, mivel a kevésbé fontos programok, mint az **XII** és a TeX hiányoznak belőle. Az M<sup>C</sup>C disztribúció elsősorban azon felhasználók számára megfelelő, akik a további programok telepítését maguk szeretnék elvégezni és pontosan tudni akarják, hogy a merevlemez, amelyet használnak, mit tárol. Kezdő felhasználóknak ez a programcsomag nem ajánlható.

## A Slackware programcsomag

A Slackware egy újabb, de nagy hírnévre rendelkező disztribúció. Eredetileg erősen támaszkodott az SLS csomagra, azonban időközben kifejlesztették saját telepítő rutinjait. A Slackware **színes** ablakai, párbeszéd- és listapaneljai a felhasználó számára már a telepítés elején lehetővé teszik a programegységek, a merevlemez, a partició, és a nyelvnek megfelelő billentyűzetkiosztás kiválasztását. A Slackware disztribúciót hajlékony- vagy merevlemeandről, illetve CD-ről vagy NFS-en keresztül egyszerű lehet telepíteni.

A Slackware disztribúcióba tartozó **programcsomagok** köre meglehetősen széles. Az alaprend-szeren kívül az XII, XView, alkalmazói programok, TCP/IP, UUCP hálózati programok, játék-programok, és a legújabb GNU Emacs telepítését is lehetővé teszi. A Slackware programcsomag összeállítói azt is garantálják, hogy a csomag minden a legújabb programverziókat tartalmazza, amelyek megfelelni nem kis követelmény, tekintve a Linux-rendszer rendkívül gyors fejlődését.

## CD disztribúciók

Sok programcsomag ma már CD-n áll rendelkezésre. Léteznek olyan **CD-előfizetések** is, amelyek évente többször CD-n szállítják a Linux kernel és az egyéb programok legújabb verzióját, de még az FTP szerverről letölthető teljes Linux könyvtárfa másolatát is. Az egyik ilyen a Yggdrasil Computing-tól származó CD, amely futtatható és telepíthető Linux rendszert tartalmaz, továbbá a MIT XII/R6 forráskódját és sok egyéb GNU segédprogramot.

A CD-előfizetés egyszerű és gazdaságos abban az esetben, ha a felhasználó nem fér hozzá az Internet hálózathoz.

A CD-n terjesztett változatoknak egyedül az a hátránya, hogy a Linux sokkal gyorsabban változik annál, mint amit a CD-verziók kibocsátása követni tud. Ugyanis szinte hetente jelennek meg a hálózaton új verziók, amelyek számos fejlesztést és bővítést tartalmaznak.

## A Unifix disztribúció

A németországi Unifix Software of Braunschweig cégtől származó CD disztribúció különösen érdekes. A CD-n kívül ugyanis a megrendelő még egy kis kézikönyvet és egy behúzó lemezt is kap, amellyel elvégezheti a rendszer első behúzását. A meglevő DOS particióra vagy egy új particióra a 8 Mb-os minimális konfigurációt telepítve a rendszer már használatra kész.

Az alkalmazói programok közvetlenül a CD-ről futtathatók. Ha a számítógépnek elég memória van (8-16 Mb), akkor a rendszer a CD legfontosabb adatait memóriában tartja, amely az elérési időt jelentősen csökkenti. A teljes Linux szoftver időigényes telepítése így feleslegessé válik.

A Unifix disztribúció a többitől tartalmában is különbözik. A szoftverlista, amely WWW klients-ként olvasható, az összes rendelkezésre álló programot minősíti. A lemezen található segédprogramok és alkalmazások konfigurációja jól átgondolt.

## 5.2 A FORRÁSOK

A Linux mostanában tapasztalt népszerűségének köszönhetően nyilvánvalóan sok forrásból lehet a csomagot és az egyéb Linuxos programokat beszerezni. Az összes lehetőség közül azonban a leggyorsabb és a legközvetlenebb az Interneten egy FTP szerverhez csatlakozni.

### FTP szerverek

Európában a Linuxszal foglalkozó legfontosabb FTP szerver az `ftp.funet.fi` Finnországban, amely az eredeti Linux szerver volt. A kernel legújabb verziója valamint az alfa verziói is itt találhatók meg. A MIT FTP szervere a `tsx-11.mit.edu` a legújabb GNU C **fordítóprogramot** és a Linux C könyvtárakat tartalmazza. A University of North Carolina egyetemen ugyancsak megtalálható a Linux szoftverek széles paleitája. Ez a szerver elsődlegesen olyan szoftversorok archívuma, amelyeket általában Linuxra.

E kiszolgálók tartalma rendszeres időközökben a világ számos egyéb FTP szerverére is leképződik, tükrözödik, aminek az a célja, hogy az Internet általános forgalmát csökkentsse. (Például az európai felhasználók számára a csatlakozás egy európai szerverrel sokkal gyorsabb és megbízhatóbb, mint az amerikai szerverekkel létesített közvetlen kapcsolat.) A következő lista a legismertebb tükörszervereket tartalmazza:

Név	IP cím	Linux könyvtár
<code>tsx-11.mit.edu</code>	18.172.1.2	<code>/pub/linux</code>
<code>sunsite.unc.edu</code>	152.2.22.81	<code>/pub/Linux</code>
<code>ftp.funet.fi</code> 128.214.248.6	<code>/pub/Linux</code>	
<code>ftp.cdfom.com</code>	192.153.46.2	<code>/pub/linux</code>
<code>net.tamu.edu</code>	128.194.177.1	<code>/pub/linux</code>
<code>ftp.mcc.ac.uk</code>	130.88.203.12	<code>/pub/linux</code>
<code>src.doc.ic.ac.uk</code>	146.169.2.1	<code>/packages/linux</code>
<code>ftp.informatik.tu-muenchen.de</code>	131.159.0.110	<code>/pub/Linux</code>
<code>ftp.informatik.rwt-aachen.de</code>	137.226.112.172	<code>/pub/Linux</code>
<code>ftp.ibp.fr</code> 132.227.60.2	<code>/pub/linux</code>	
<code>kirk.bond.edu.au</code>	131.244.1.1	<code>/pub/OS/Linux</code>
<code>ftp.uu.net</code> 137.39.1.9	<code>/systems/unix/linux</code>	
<code>ftp.win.tue.nl</code>	131.155.70.100	<code>/pub/linux</code>
<code>ftp.stack.ure.ue.nl</code>	131.155.2.71	<code>/pub/linux</code>
<code>ftp.denet.dk</code> 129.142.6.74	<code>/pub/OS/Linux</code>	
<code>nictucca.edu.tw</code>	140.111.1.10	<code>/Operating-Systems/Linux</code>
<code>nic.switch.ch</code>	130.59.1.40	<code>/mirror/linux</code>
<code>monu1.cc.monash.edu.au</code>	130.194.1.101	<code>/pub/linux</code>
<code>cnuce-arch.cn.it</code>	131.114.1.10	<code>/pub/Linux</code>
<code>ftp.linux.org</code>	198.182.196.129	<code>/pub</code>

Az `FTP-Roadmap.table.z` Linux fájl a németországi FTP szerverekről nyújt széleskörű áttekintést. Ez a fájl a többi helyek között a University of Erlangen (`FTP.uni-erlangen.de`) szerveren található meg a `pub/Linux/LOCAL/Roadmaps` könyvtárban. (Gyakori, hogy az FTP szerverek könyvtáráit átszervezik, így előfordulhat, hogy egy, a fenti lista alapján keresett könyvtár már nem található az adott helyen – a lektor.)

### **FTP mail szerverek (kiszolgálók)**

Az felhasználó, aki nem rendelkezik közvetlen Internet csatlakozással, azonban tud elektronikus postát küldeni és fogadni, az FTP mail szerverek egyikén keresztül is elérheti az Internet FTP szervereinek egyikét, amely programokat tud küldeni. Egyszerűen csak egy elektronikus postát kell küldeni a megfelelő parancssal együtt a mail szervernek, amely aztán megcsinál az FTP szervert, felbontja a programot kisebb egységekre, majd a uuencode parancssal kódolva elküldi a választ elektronikus postán.

Az FTP mail szerver által értelmezett parancsok listáját a felhasználó úgy kaphatja meg, ha olyan üzenetet küld a szervernek, amelynek első sorában a help parancs áll. Ilyen mail szerverek például a következők:

FTP-mailer@informatic.tu-muenchen.de

ftpmail@decwrl.dec.com

Fontos megjegyezni azonban, hogy ezek a szerverek általában nem alkalmasak arra, hogy teljes Linux disztribúciókat átküldjenek, mivel általában kis adatmennyiségek továbbítására vannak korlátozva.

### **Kereskedelmi disztribúciók**

A Linuxhoz és a Linux programokhoz a kereskedelmi hálózatban is hozzá lehet jutni lemezeken, CD-ken vagy **postai** megrendelés útján. A címeket általában a számítógépes szaklapok tartalmazzák. A kereskedők közül vannak olyanok is, aki teljes merevlemezeket vagy PC-ket árulnak telepített és futó Linux környezettel.

### **Postaládák (Mailboxes)**

Egy lelkes felhasználó, Matthias Gmelch listát készített, amely a világban elérhető Linuxos postaládák áttekintését tartalmazza. Maga a lista a tsx-11.mit.edu FTP szerveren található a /pub/linux/docs alkönyvtárban, azonban sok tükröszerben is elérhető.

## **5.3 HARDVER**

A Linuxszal kapcsolatos egyik leggyakoribb kérdések egyike az, hogy milyen hardvert támogat és mire van szüksége. Általános előfeltétel a PC-kompatibilis számítógép 30386-os vagy ennél korszerűbb processzorral. A Linux a régebbi XT-ken vagy 80286-as processzorral rendelkező AT-ken egyáltalán nem futthatató, mivel szüksége van az ún. taszkkezelésre, amelyet csak azok a PC-k biztosítják, amelyek 80386-os vagy ennél korszerűbb processzorral rendelkeznek.

### **RAM**

A Linux futtatásához minimálisan **2 Mb RAM**-ra van szükség. A telepítési folyamat azonban 2 Mb RAM esetén nehézségekbe ütközhet, mivel ez a RAM-méret a telepítőprogramok számára nem elegendő. Ilyen esetben a rendszernek **csererepartíciót** vagy cserefájlokat kell létrehozni, annak érdekében, hogy el tudja indítani a telepítéshez szükséges szövegszerkesztő és egyéb programokat.

A normális munkavégzéshez még szöveges üzemmódban is legalább 4 Mb RAM szükséges. Azonban a normál verzió esetén az X Windows Systemmel való együttműködéshez legalább 8 Mb RAM kell. Ha azonban a számítógépen 16 Mb RAM van, akkor ezzel sokkal nagyobb teljesítőképesség érhető el az X11 esetén.

### Merevlemez

Ha a felhasználó úgy dönt, hogy a grafikus felhasználói interfész feláldozza annak érdekében, hogy a rendszer a lehető legkevesebb helyet foglalja el a merevlemezen, akkor a méretet kb. 40 Mb-ra csökkentheti. A teljes telepítéshez, amely magában foglalja az X11-et, a C fordítóprogramot, az összes segédprogramot és eszközt, 150 Mb hely szükséges. (Ez az alkalmazások nélküli értendő, az egyik legújabb terjesztés, a Debian teljes telepítéséhez minimum 400-500 Mb lemezterület szükséges. Ilyenkor persze sok olyan csomag is felkerül, amit ritkán vagy soha nem fog használni – a lektor.) Nehéz azonban meghatározni azt a felső értéket, amelyre egy Internethez csatlakozó hálózati gép esetén gondolni kell. Az innen letöltött különféle programok ugyanis könnyen elérhetik még a 2 Gb méretet is.

Gazdaságos első telepítéshez IDE vagy EIDE interféssel rendelkező merevlemez használata javasolt. A merevlemezeknek több Gb-ig igen sok változata kapható, amely megfelelő teljesítőképességet biztosít egyfelhasználós üzemmódban. Azonban a nagyobb méret és professzionális használat esetén inkább az SCSI (Small Computer System Interface) illesztő alkalmazása javasolt.

A Linux kernel (rendszermag) már a kezdettől fogva támogatta az IDE interfészét, és így ennek használatahoz külön kezelőprogramra nincs szükség. Még olyan kernel patch is van, amely lehetővé teszi két IDE kontroller párhuzamos használatát, feltéve, hogy azok I/O portja és megszakítási meglefelőn beállíthatók. (Az újabb linux kernelek már beépítve tartalmazzák a második IDE vezérlő meghajtóját. IDE változatban, kiegészítve IDE/ATAPI CD meghajtók, szalagegységek meghajtó programjaival, kihangsúlyosan sok speciális áramkörkészlet adta gyorsítási lehetőséget. Ilyen áramkörök pl. az Intel Triton I és II, az Ali, a CMD 640 alapú laplápi vezérlők – a lektor.)

Nagyobb és gyorsabb merevlemezek esetén általában az SCSI-t alkalmazzák, amely maximálisan hétszeres párhuzamos működtetését teszi lehetővé. Ezek az eszközök merevlemezek, mágnesszalagos adatrögzítők (streamerek), mágneses-optikai meghajtók, CD-ROM-meghajtók vagy szkennerek lehetnek.

A SCSI alrendszer speciális SCSI host adapter vezérlőt igényel a Linux kerneljében, de ezt már az összes újabb Linux rendszermag tartalmazza. Különleges host adapterek használata nem javasolt, mivel ezekhez a kezelőprogramok csak ritkán állnak rendelkezésre. (Itt mutatkozik meg a linux hatámas előnye a többi rendszerről szemben: mivel megvan minden forráskódban, ha nincs meg egy eszközmeghajtó, megírjak – a lektor.) Azok a kezelőprogramok viszont, amelyeket DOS-hoz vagy a UNIX-hoz készítettek, a Linuxhoz nem használhatók.

Az olyan jól ismert cégek, mint az Adaptec, NCR, Future Domain vagy a Seagate által gyártott kontrollerek nem okozhatnak problémát. Kétséges esetekben érdemes megtekinteni a támogatott hardverek listáját, amely az FTP szervereken a „Hardware HOWTO” részben található.

### Videokártyák

A videokártyák esetén érdemes a legegyszerűbbet használni, S3-as processzorkészlettel. Az X11 R6-os csomag Linuxhoz készített változatához speciális szerver-optimalizált meghajtókat is készítettek, amelyek a 32 vagy 64 bites, Local Bus vagy PCI változatainál az S3 vagy a Mach processzorkészletek esetében különösen gyorsak.

Más gyártók által készített és támogatott hardverek listája az FTP szervereken a HOWTO részekenél az „Xfree86 HOWTO” feliratnál található meg.

Különleges vagy kevés memóriával rendelkező videokártya esetén vagy egy általános VGA szerver használata lehetséges, amely csak 16 szint és 640x480-as felbontást támogat, vagy mono szerver, amely azonban a Hercules Kártya használatait is lehetővé teszi.

A Linux esetében a soros porthoz csatlakoztatott egér, vagy a PS/2-vel kompatibilis pozícionáló gömb ugyanések alkalmazható.

### **Busz-rendszer (ISA/EISA/PCI)**

A Linux a régi AT buszos (ISA) alaplapokat ugyanúgy támogatja, mint a rugalmasabb és gyorsabb EISA buszt, vagy a Local Bus bővítésekét és az újabb PCI buszokat.

A PCI busz processzor független és sokkal gyorsabb mint az EISA rendszer. 32 bites módon még az 130 Mbytes/s -os adatátvitelre is képes. Sok PCI alaplapnak nagyon gyors SCSI chipje van (NCR 53c810), amelyhez a kernel tartalmazza a kezelő programot.

Bár a PCI szabványokra vonatkozó specifikációt különös gondossággal állították össze, vannak olyan PCI összetevők, amelyek mégis elérnek eztől. Ez viszont kompatibilitási problémákhoz vezethet az alaplap és a perifériakártyák között. A HOWTO PCI-re vonatkozó részében ezekről a problémákról továbbiakat lehet olvasni, valamint PCI vásárlásához tanácsokat lehet kapni.

### **Perifériák**

A hálózati kártyákhoz meglehetősen nagy a kezelőprogramok választéka. Az ismert gyártók – pl. a Novell, 3Com és SMC – által készített kártyákon kívül számos HP és Dlink kártya használata is támogatott. Még az Aronet kártyához is vannak kezelőprogramok. Léteznek olyan általános kártyák is, amelyek az előzőekkel kompatibilisek és ugyancsak használhatók.

A UNIX tradícióinak megfelelően a Linux is lehetővé teszi a soros porthoz csatlakoztatott ASCII terminálok használatát. Ezekhez a kártyákhoz kezelőprogramok is rendelkezésre állnak.

Az egyik fontos jellemző, ami a Linuxot sok más rendszertől megkülönbözteti, az ISDN kártyák közvetlen kernelen keresztüli támogatása. Az *isdn4linux* kezelőcsomag lehetővé teszi aktív ISDN kártyák hálózati interfésként való alkalmazását a Linux alatt. Ezt a szoftvert az Interneten keresztül az alábbi FTP szerveren lehet beszerezni: <FTP.to.com:/pub/isdn4linux>

A Linux alatt multimédia alkalmazások is használhatók. Az olyan perifériák, mint a hangkártyák (Soundblaster, Soundblaster 16, Adlib, Gravis Ultra Sound vagy ProAudio Spectrum 16) és a CD-ROM-meghajtók a megfelelő kezelőprogrammal használhatók. Ebben az esetben is fontos ellenőrizni a támogatott hardverek listáját, hogy az adott CD-ROM-meghajtónak van-e kezelőprogramja. Az SCSI-meghajtókat különösen könnyű csatlakoztatni.

Még az olyan különleges hardverek esetében is, mint a transzputer kártyák, léteznek kezelőprogramok, amelyek ezek használatát a Linux alatt lehetővé teszik.

**Javasolt hardver:** a hardver és a szoftver folyamatos fejlődése miatt 80486-os számítógép 33 MHz vagy nagyobb órajellel és 16 Mb RAM ad megfelelő hardver konfigurációval. A merevlemez tárolókapacitásának legalább 200 Mb-osnak kell lennie, mivel a hatékony munkát az ennél kevesebb hely már nem teszi lehetővé. Ha várhatóan CD-ROM-meghajtó vagy streamer használataira is szükség lesz, akkor már a kezdetektől az SCSI konfiguráció írdemes választani.

A 14 "-es monitor mérete nem elég akkor, ha több ablakban párhuzamosan futó környezetben kell dolgozni, vagyis ha egyszerre több a nyitott ablak. A Linux X képes virtuális képernyőt kezelni (amelynek a mérete nagyobb, mint a monitor felbontása, ezen minden görgeti a képet) és vannak olyan monitorok, amelyek a 800x600-as felbontásban is üzemeltethetők, így a kezdéshez a 17 "-es monitor méret nem feltétlenül szükséges.

## **5.4 A TELEPÍTÉS**

Ez a rész Slackware disztribúciót feltételezve a telepítési eljárást tárgyalja. (Ez a disztribúció meglehetősen széles körben terjedt, mivel szabadon másolható és a kereskedők által terjesztett CD-kön is általában megtalálható.)

### A Linux telepítésének alapvető lépései a következők:

1. A minimális Linux rendszer betöltése a boot lemezről
2. Partíciók létrehozása a Linux számára
3. Fájlrendszer és csereterületek létrehozása
4. A rendszer merevlemezre másolása
5. A legfontosabb rendszerrögzítők konfigurálása
6. A boot manager telepítése
7. A grafikus felhasználói kezelőfelület telepítése
8. Felhasználók létrehozása

### Betöltő

A telepítési eljárás a minimális Linux rendszer betöltésével kezdődik, amely csak a kernelt, a legfontosabb segédprogramokat és a telepítő programot tartalmazza. A Slackware disztribúció erre a cédra két lemezzel rendelkezik, az egyik a betöltő (boot), a másik a gyökér (root) lemez, amelyek mindegyike  $3\frac{1}{4}$ -es és  $5\frac{1}{2}$ -es méretekben áll rendelkezésre.

A behúzó lemezek a kernelben tárolt **kezelőprogramokban** térhetnek el egymástól, míg a gyökérlemezek minimális fájlrendszer tartalmaznak a legfontosabb **segédprogramokkal** és a telepítő szkripttel.

Ha az adott disztribúció lenyomtsomagból áll, akkor a lemezekről a rendszer behúzható. Egyébként a telepítő lemezek **image** fájlokat tartalmazznak, amelyeket a behúzó lemezekre egy speciális segédprogrammal át kell vinni. A behúzó és a gyökér lemezekre vonatkozó **image** fájlok a Slackware disztribúció esetében a következő könyvtárakban találhatók:

- bootdks.12
- bootdks.144
- rootdks.12
- rootdks.144

A **bootdks.144** könyvtár a következő fájlokat tartalmazza:

README	cd535.gz	old1118.gz
WHICH.ONE	loaded.gz	sbped.gz
alpha.gz	mitsumi.gz	scsi.gz
barc.gz	nec260.gz	scsinet.gz
cdu31a.gz	net.gz	xt.gz

A  $3\frac{1}{4}$ -es hajlékony lemezekhez az **install/1.44meg/bootdisks** és az **install/1.44meg/rootdisks** könyvtárak tartalmazzák a megfelelő adatokat. Az  $5\frac{1}{2}$ -es lemezre vonatkozó fájlokat az **install/1.2meg** könyvtár tartalmazza.

A Linux NFS-en keresztüli távoli telepítéséhez a kernelnek tartalmaznia kell a hálózati kártya kezelőprogramját is. Ha a számítógépnek olyan hálózati kártyája van, amelyre a betöltő lemezen levő kernel nem tartalmaz kezelőprogramot, akkor egyedi betöltő lemezt kell létrehozni (lásd az 5.5 szakaszban).

Ugyanez vonatkozik azokra a rendszerekre is, amelyek SCSI merevlemezzel és különleges SCSI host adapterrel rendelkeznek, amelyekhez a betöltő lemezek egyike sem tartalmazza a megfelelő kezelőprogramot. Ha a helyzet nem egyértelmű, akkor az **image** fájlokkal azonos könyvtárban levő **README** fájlt érdemes elolvasni, amelyből megállapítható, hogy mely fájlok milyen hardver konfigurációhoz szükségesek.

A gyökérlemezekhez más fájlok tartoznak, de csak abban különbözik, hogy a lemezről levő telepítőprogram színes-e vagy monokróm. A legtöbb esetben a színes változatot használják, amely a 3 1/4-es meghajtó esetén a `rootdsks.144/colorl44.gz` fájl. A további könyvtárak egyedi betöltőlemezek készítését teszik lehetővé vagy egyéb eszközökkel és információkkal tartalmaznak.

A betöltő- és gyökérlemezekre vonatkozó megfelelő fájlok kiválasztása után azokat formázott lemezre kell átmásolni vagy a DOS `rawrite.exe` vagy a UNIX `dd` parancsával. Először azonban ki kell tömörítetni a `gzip` programmal. A következő példa ezt az eljárást szemlélteti:

```
nicetry:/tmp# gzip -d bare.gz
nicetry:/tmp# dd if=bare of=/dev/fd0 bs=8k
180+0 records in
180+0 records out
nicetry:/tmp#
```

A lemezek létrehozása után a betöltő lemezt a megfelelő meghajtóba kell helyezni, és a számítógépet újraindítani. Az első üzenet, amely megjelenik, a **LILO**-tól származik, a Linux boot-menedzserről és betöltőjéről, amely a betöltő lemezre telepítve van. A következő képernyőképen a betöltés során megjelenő első üzenet látható:

```
Welcome to the Slackware Linux 3.1.0 bootkernel disk!

If you have any extra parameters to pass to the kernel, enter them at the
prompt below after one of the valid configuration names (ramdisk, mount,
drive2)

Here are some examples:

    ramdisk hd=cyl,hds,secs (Where "cyl", "hds", and "secs" are the number of
                           cylinders, sectors, and heads on the drive. Most
                           machines won't need this.)

In a pinch, you can boot your system with a command like:
    mount root=/dev/hd1a

On machines with low memory, you can use mount root=/dev/fd1 or
mount root=/dev/fd0 to install without a ramdisk. See LOWMEM.TXT for details.

If you would rather load the root/install disk from your second floppy drive:
    drive2 (or even this: ramdisk root=/dev/fd1)

DON'T SWITCH ANY DISKS YET! This prompt is just for entering extra parameters.
If you don't need to enter any parameters, hit ENTER to continue.
```

A Linux Loader most adatbevitelre vár. A szokásos telepítés során, amikor a betöltő meghajtót a gyökérlemez beolvasására is használják, elegéndő a **<Enter>** billentyű lenyomása.

Bizonyos esetekben előfordul, hogy a CD-ROM-meghajtót, a merevlemezét vagy a hálózati kártyát a rendszer nem megfelelően azonosítja be, így a kernel számára különleges paraméterek áta-

dása szükséges. Ez a paraméterek felsorolása a betöltő opció specifikációja után található meg. A betöltő készüléti jelhez a következők szerint kell az adatbevitelt végrehajtani:

```
Boot choice Parameter=Value,Value,...  
Parameter=Value,Value,...
```

Ezt a példát az előző Linux Loader üzenet tartalmazza. De lássunk egy másik példát is:

```
ramdisk ether=5,0x320
```

Ez azt közi az Ethernet kártya kezelőprogramjával a kernelben, hogy a kártya az 5-ös **megszakítás** tást és a 0x320 I/O címet használja.

Ha a választás helyes és megfelelő volt, ha minden szükséges paraméter átadásra került, akkor a Linux Loader beölti a kernelt. A kernel először kitömöríti magát, majd inicializálja az egyedi eszközmeghajtókat és végül megjeleníti a megfelelő üzeneteket. A kezelőprogramokra vonatkozó üzenetek jelzik, hogy a rendszer mely eszközöt ismerte fel és melyeket sikerült inicializálnia. A betöltési eljárás során megjelenő üzenetek sorrendje a következőhöz hasonló:

```
Apr 19 20:55:37 darkstar kernel: Console: colour EGA+ 80x25, 1 virtual  
console (max 63)  
Apr 19 20:55:37 darkstar kernel: bios32_init : BIOS32 Service Directory  
structure at 0x000facd0  
Apr 19 20:55:37 darkstar kernel: bios32_init : BIOS32 Service Directory  
entry at 0xfb190  
Apr 19 20:55:37 darkstar kernel: pcibios_init : PCI BIOS revision 2.10  
entry at 0xfb1c0  
Apr 19 20:55:37 darkstar kernel: Probing PCI hardware.  
Apr 19 20:55:37 darkstar kernel: Unknown PCI device. PCI Vendor id=3.  
PCI Device id=8880.  
Apr 19 20:55:37 darkstar kernel: PLEASE MAIL POTTER@CAO-VLSI.IBP.FR your  
hardware description and /proc/pci.  
Apr 19 20:55:37 darkstar kernel: Calibrating delay loop.. ok - 40.18  
Bogomips  
Apr 19 20:55:37 darkstar kernel: Serial driver version 4.11 with no serial  
options enabled  
Apr 19 20:55:37 darkstar kernel: tty00 at 0x03f8 (irq = 4) is a 16550A  
Apr 19 20:55:37 darkstar kernel: tty01 at 0x02f8 (irq = 3) is a 16550A  
Apr 19 20:55:37 darkstar kernel: lpt at 0x0378, using polling driver  
00228000  
Apr 19 20:55:37 darkstar kernel: ftape: allocated 3 buffers aligned at:  
Cache, LBA, CHS=787/64/63, MaxMult=16  
Apr 19 20:55:37 darkstar kernel: ide0: primary interface on irq 14  
Apr 19 20:55:37 darkstar kernel: Floppy drive(s): fd0 is 1.44M  
Apr 19 20:55:37 darkstar kernel: FDC 0 is a post-1991 82077  
Apr 19 20:55:37 darkstar kernel: scsi-ncr53c7,8xx : at PCI bus 0, device 8,  
function 0
```

```
Apr 19 20:55:37 darkstar kernel: scsi-ncr53c7,8xx : warning : revision  
of 2 is greater than 1.  
Apr 19 20:55:37 darkstar kernel: scsi-ncr53c7,8xx : NCR53c810 at memory  
0xf2000000, io 0x6000, irq 10  
Apr 19 20:55:37 darkstar kernel: scsi0 : using io mapped access  
Apr 19 20:55:37 darkstar kernel: scsi0 : using initiator ID 7  
Apr 19 20:55:37 darkstar kernel: scsi0 : using level active interrupts  
Apr 19 20:55:37 darkstar kernel: scsi0 : burst length 8  
Apr 19 20:55:37 darkstar kernel: scsi0 : using 40MHz SCSI clock  
Apr 19 20:55:37 darkstar kernel: scsi0 : NCR code relocated to 0x2404f0  
Apr 19 20:55:37 darkstar kernel: scsi0 : test 1 started  
Apr 19 20:55:37 darkstar kernel: scsi0 : NCR53c7,8xx (rel 4)  
Apr 19 20:55:37 darkstar kernel: scsi : 1 host.  
Apr 19 20:55:37 darkstar kernel: Vendor: FUJITSU Model: M2694ES-512  
Rev: 811F  
Apr 19 20:55:37 darkstar kernel: Type: Direct-Access ANSI SCSI  
revision: 02  
Apr 19 20:55:37 darkstar kernel: Detected scsi disk sda at scsi0, id 1,  
lun 0  
Apr 19 20:55:37 darkstar kernel: scsi : detected 1 SCSI disk total.  
Apr 19 20:55:37 darkstar kernel: SCSI Hardware sector size is 512 bytes  
on device sda  
Apr 19 20:55:37 darkstar kernel: Memory: 14684k/16384k available  
(816k kernel code, 384k reserved, 500k data)  
Apr 19 20:55:37 darkstar kernel: This processor honours the WP bit even  
when in supervisor mode. Good.  
Apr 19 20:55:37 darkstar kernel: Swansea University Computer Society  
NET3.019  
Apr 19 20:55:37 darkstar kernel: Swansea University Computer Society TCP/IP  
for NET3.019  
Apr 19 20:55:37 darkstar kernel: IP Protocols: ICMP, UDP, TCP  
Apr 19 20:55:37 darkstar kernel: PPP: version 0.2.7 (4 channels)  
NEW_TTY_DRIVERS OPTIMIZE_FLAGS  
Apr 19 20:55:37 darkstar kernel: TCP compression code copyright 1989 Regents  
of the University of California  
Apr 19 20:55:37 darkstar kernel: PPP line discipline registered.  
Apr 19 20:55:37 darkstar kernel: SLIP: version 0.8.3-NET3.019-NEWTTY  
(4 channels) (6 bit encapsulation enabled)  
Apr 19 20:55:37 darkstar kernel: CSLIP: code copyright 1989 Regents  
of the University of California  
Apr 19 20:55:37 darkstar kernel: Checking 386/387 coupling... Ok,  
fpu using exception 16 error reporting.  
Apr 19 20:55:37 darkstar kernel: Checking 'hlt' instruction... Ok.  
Apr 19 20:55:37 darkstar kernel: Linux version 1.2.13 (root@bigkitty)  
(gcc version 2.7.0) #1 Wed Aug 23 03:23:21 CDT 1995  
Apr 19 20:55:37 darkstar kernel: Partition check:  
Apr 19 20:55:37 darkstar kernel: sda: sdal sda2  
Apr 19 20:55:37 darkstar kernel: hda: multiple mode turned off  
Apr 19 20:55:37 darkstar kernel: hda: hdal hda2 hda3 hda4
```

```

Apr 19 20:55:37 darkstar kernel: VFS: Mounted root (ext2 filesystem)
readonly.
Apr 19 20:55:37 darkstar kernel: Adding Swap: 42332k swap-space
Apr 19 20:55:37 darkstar kernel: Unable to identify CD-ROM format.
Apr 19 20:55:37 darkstar kernel: end_request: I/O error, dev 2100,
sector 64
Apr 19 20:55:37 darkstar kernel: isofs_read_super: bread failed, dev 0x2100
iso_blknum 16
Apr 19 20:55:37 darkstar kernel: Unable to identify CD-ROM format.
Apr 19 20:55:39 darkstar kernel: iBCS: socks sys registered on character
major 30
Apr 19 20:56:20 darkstar login: ROOT LOGIN ON ttys1

```

Miután a kernel sikeresen betölödött és elindult, egy üzenet jelenik meg a képernyőn, amely arra kéri a felhasználót, hogy a meghajtóban cserélje ki a betöltő lemezt a gyökérlemezkel, amely az alap fájlrendszeret tartalmazza.

```

Please remove boot kernel disk from your floppy drive,
insert a
root/install disk (such as one of the Slackware colorl44,
coirlite,
ttyl44, or ttyl2 disks) or some other disk you wish to load
into a
ramdisk and boot, and then press ENTER to continue.

```

Amikor a gyökérlemez berakta, ennek tartalmát a rendszer beolvassa a RAM-ba, teljesen betölve a fájlrendszer és a telepítőprogramot, így a meghajtó felszabadul, mivel a lemezre a továbbiakban már nem lesz szükség.

```

VFS: Disk change detected on device 2/28
RAMDISK: Minix filesystem found at block 0
RAMDISK: Loading 1440 blocks into RAM
disk.
done

```

A behúzási folyamat befejezésként egy üdvözlő üzenet jelenik meg a képernyőn, amely a további lépésekkel is megadja:

```

Welcome to the Slackware Linux installation disk, (v. 3.1.0)
#####
# IMPORTANT! READ THE INFORMATION BELOW CAREFULLY. #####
- You will need one or more partitions of type "Linux native" prepared.
It is
also recommended that you create a swap partition (type "Linux swap")
prior

```

to installation. Most users can use the Linux "fdisk" utility to create and tag the types of all these partitions. OS/2 Boot Manager users, however, should create their Linux partitions with OS/2 "fdisk", add the bootable (root) partition to the Boot Manager menu, and then use the Linux "fdisk" to tag the partitions as type "Linux native".

- If you have 4 megabytes or less of RAM, you MUST activate a swap partition before running setup. After making the partition with fdisk, use: mkswap /dev/<partition> <number of blocks>; swapon /dev/<partition>
- Once you have prepared the disk partitions for Linux, and activated a swap partition if you need one, type "setup" to begin the installation process.
- If you want the install program to use monochrome displays, type: TERM=vt100 before you start "setup".

You may now login as "root".

A folytatáshoz a **login** készenléti jelhez a **root** felhasználói nevet kell beírni, amely egy további üzenet megjelenését eredményezi:

Linux 2.0.0. (Posix).

If you're upgrading an existing Slackware system, you might want to remove old packages before you run 'setup' to install the new ones. If you don't, your system will still work but there might be some old files left laying around on your drive.

Just mount your Linux partitions under /mnt and type 'pkgtool'. If you don't know how to mount your partitions, type 'pkgtool' and it will tell you how it's done.

To start the main installation, type 'setup'.

Az összes további lépést már a Linux alatt kell végrehajtani.

## **Particionálás**

A Linux telepítéséhez legalább egy szabad partícióra van szükség. Ésszerűbb megoldást nyújt azonban **három partíció** használata: egyiket a gyökér fájlrendszer, a másikat a felhasználói fájlrendszer számára, a harmadikat pedig cserepartícióként. Ezután a rendszer új telepítése a /home partíciót sértetlenül hagyja, készen a telepítés utáni logikai beillesztéshez. Ezzel viszont az összes felhasználói fájl is megmarad.

Ha a merevlemez nem tartalmaz egyetlen Linux partíciót sem, akkor azokat először létre kell hozni a Linux **fdisk** parancsával. A következő példa a partíciók létrehozásának műveletét szemlélteti. Kiindulásként azt feltételezzük, hogy a DOS számára már létezik egy partíció a merevlemezen.

Az új partícióként létrehozandó partíció típusa 83 (Linux native), míg a cserepartíció típusa 82 (Linux swap).

A partíció típusát a telepítőprogram beolvassa, és innen tudja meg, hogy a telepítéshez milyen partíciók használhatók. Ha a Linux számára már létezik partíció, amelyet a DOS fdisk parancsval hoztak létre, akkor a Linux fájlrendszer által használendő partíció típusát a Linux fdisk parancsával 83-ra kell beállítani.

A következő képernyőkép az fdisk parancs Linux alatti futtatását szemlélteti:

```
Command (m for help): p
Disk /dev/sda: 34 heads, 61 sectors, 1012 cylinders
Units = cylinders of 2074 * 512 bytes
Device Boot Begin Start End Blocks Id System
Command (m for help): n
Command action
  e extended
  p primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1012): 1
Last cylinder or +size or +sizeM or +sizeK ([1]-1012): +100M

Command (m for help):
Command (m for help): n
Command action
  e extended
  p primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (100-1012): 100
Last cylinder or +size or +sizeM or +sizeK ([100]-1012): +8M

Command (m for help): p
Disk /dev/sda: 34 heads, 61 sectors, 1012 cylinders
Units = cylinders of 2074 * 512 bytes
Device Boot Begin Start End Blocks Id System
/dev/sda1      1      1     99 102632+  83 Linux native
/dev/sda2    100    100   107     8296  83 Linux native
Command (m for help): n
Command action
  e extended
  p primary partition (1-4)
p
partition number (1-4): 3
```

```

First cylinder (108-1012): 108
Last cylinder or +size or +sizeM or +sizeK ([108]-1012): 1012

Command (m for help): p

Disk /dev/sda: 34 heads, 61 sectors, 1012 cylinders
Units = cylinders of 2074 * 512 bytes

      Device Boot  Begin    Start     End   Blocks  Id System
/dev/sda1            1        1     99  102632+  83 Linux native
/dev/sda2         100     100    107     8296  83 Linux native
/dev/sda3         108     108    1012   938485  83 Linux native

Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): L

 0  Empty          9  AIX bootable    75  PC/IX           b7  BSDI fs
 1  DOS 12-bit FAT  a  OS/2 Boot Manag 80  Old MINIX       b8  BSDI swap
 2  XENIX root     40  Venix 80286   81  Linux/MINIX     c7  Syrinx
 3  XENIX usr      51  Novell?      82  Linux swap       db  CP/M
 4  DOS 16-bit <32M 52  Microport    83  Linux native     e1  DOS access
 5  Extended       63  GNU HURD     93  Amoeba          e3  DOS R/O
 6  DOS 16-bit >=32 64  Novell Netware 94  Amoeba BBT     f2  DOS
secondary
 7  OS/2 HPFS      65  Novell Netware a5  BSD/386        ff  BBT
 8  AIX

Command (m for help): p

Disk /dev/sda: 34 heads, 61 sectors, 1012 cylinders
Units = cylinders of 2074 * 512 bytes

      Device Boot  Begin    Start     End   Blocks  Id System
/dev/sda1            1        1     99  102632+  83 Linux native
/dev/sda2         100     100    107     8296  82 Linux swap
/dev/sda3         108     108    1012   938485  83 Linux native

Command (m for help): w

The partition table has been altered!
Calling BLKPART ioctl() to re-read partition table
Syncing disks
Reboot your system to ensure partition table is updated

```

Az összes partíció, beleértve a logikai meghajtóhoz tartozó külső partíciókat is, a Linux alatt szekvenciálisan számozódik és elsődleges partícióként használható. Az összes partíció fájlokra képeződik le a /dev könyvtárban, és a nevük a hd karakterekkel kezdődik a normál merevlemezek esetén és az sd karakterekkel az SCSI merevlemezek esetén.

## Fájlrendszerek létrehozása

A partició csak akkor képes fájlokat tárolni, ha azon a fájlrendszer már létrehozták. Ezt a műveletet rendszerint a telepítési szkript megfelelő menüpontjának kiválasztásával lehet elvégezni.

A fájlrendszerit azonban kézzel is létre lehet hozni a `mkfs` parancsal, amely csak arra szolgál, hogy a megfelelő programot elindítja a számára átadott fájlrendszer-paraméternek (`-t`) megfelelően. Az Extended-2 fájlrendszer esetében például ez a parancs az `mk2fs`.

```
nicetry:~# mkfs -t ext2 -c /dev/hda4
```

A `-c` paraméter hatására a fájlrendszer számára használt blokkok ellenőrzése a merevlemezén végbemegy.

## Cserepartíció létrehozása

A cserepartíciót az `mkswap` parancssal lehet létrehozni.

```
nicetry:~# mkswap /dev/hda3
Setting up swap space, size = 43347968 bytes
```

Az új cserepartíció aktivizálásához a `swapon` parancsot a partició nevével, mint paraméterrel kell végrehajtani.

A memória aktuális állapotát a `free` parancssal lehet lekérdezni:

```
nicetry:~# swapon /dev/hda3
nicetry:~# free
              total        used        free      shared      buffers
Mem:       14892       14036        856       5804        892
-/+ buffers:          13144       1748
Swap:      42332           0       42332
MemTotal:    14
MemFree:     0
MemShared:   5
Buffers:     0
Cached:      9
SwapTotal:   41
SwapFree:    41
```

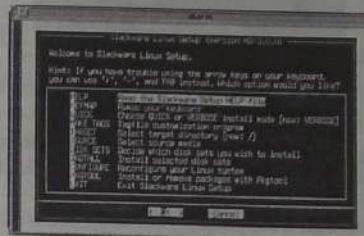
8 MB, vagy ennél több memóriával rendelkező számítógépek esetében a cserepartíció a fájlrendszerrel együtt a telepítőprogrammal létrehozható és aktivizálható. Az ADDSWAP menüpont kiválasztására a rendszer a merevlemezről végignézi, hogy van-e rajta Linux cserepartíció. Ha van, akkor az a `mkswap` parancssal inicializálható és a `swapon` parancssal aktivizálható.

Ha a számítógép 4 MB vagy ennél kevesebb RAM-mal rendelkezik, akkor a cserepartíciót még a telepítőprogram elindítása előtt kell létrehozni és aktivizálni. Ellenkező esetben ugyanis a rendelkezésre álló memória nem lesz elegendő.

### Másolás a merevlemezre

A csererepartíció és a fájlrendszer elköszítése után a Linux rendszert a merevlemezre kell másolni. Ezt a telepítőprogram `setup` parancssal történő indítássával teheti meg.

A fómenü legfontosabb pontjai a **forráseszköz** és a **célpártíció kiválasztása**, a telepítendő komponensek meghatározása és az aktuális telepítés elindítása.

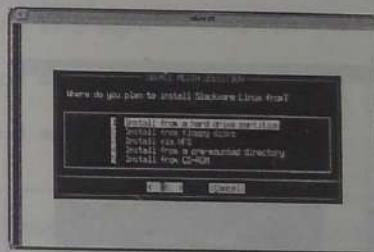


5.1. ábra. A telepítőprogram fómenüje

Ezek a menütételek automatikusan csatolva vannak egymáshoz, ami azt jelenti, hogy az első pont kiválasztására a telepítőprogram automatikusan a következő menütételekre ugrik, ha az első tételelhet tartozó műveletek elvégzésével végzett. A telepítési módoknál a `quick` és a `verbose` beállítások között ugyan választani lehet, azonban a beállítást jobb `verbose` állásban hagyni.

Forráseszközök körül szerepelhet **merevlemez**, ha a telepítendő fájlokat előzetesen ide másoltuk, **hajlékonylemezek**, ha a hálózaton egy másik számítógép, amely logikailag be van illesztve a rendszerbe az **NFS-sel**, **CD-ROM** vagy **sztrímer**. Ha a telepítés sztrímről megy végbe, akkor a művelet összetettebb, mint az egyéb esetek, így az erre vonatkozó tudnivalókat egy külön **README** fájl tartalmazza.

Az **NFS**-en keresztül végrehajtott telepítés számos hálózati **paraméter** megadását igényli, mint például a célszámítógép és az **NSF** szerver IP címe, a hálózati maszk, a hálózati cím és az **NFS** szerveren a disztribúció könyvtár-útvonala. Ezeket a fogalmakat a 9. fejezet ismerteti. Az adott eszközre vonatkozó telepítési adatokat a **hálózati adminisztrátor**tól lehet megtudni.



5.2 ábra. A forráseszköz kiválasztása



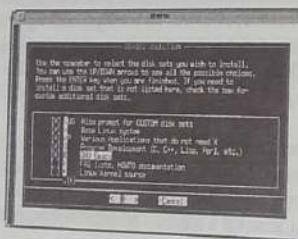
5.3 ábra. A célpártíció kiválasztása

A célpártíció kiválasztását ugyancsak a megfelelő menüpont kiválasztásával lehet végrehajtani. A telepítőprogram végignézi a rendszert az összes Linux típusú pártíciót és a kiválasztáshoz ezeket megjelenítő.

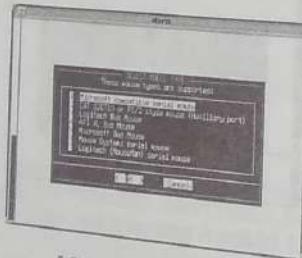
A gyökérpartíció létrehozása után a telepítőprogram megkérdezi, hogy további partíciók logikai beillesztésére is sor kerül-e. Ha a `/home` és a `/usr` rendszereket önálló fájlrendszerbe szeretné telepíteni, akkor azt ítt kell előírnia.

Ezután meg kell adnia, hogy az adott disztribúcióról mely programcsomagokat szeretné telepíteni.

Ezután következik az `INSTALL` menüpont kiválasztása, amellyel a másoláshoz a legmegfelelőbb eljárás írható el.



5.4 ábra. A programcsomagok kiválasztása



5.5 ábra. Egérkonfigurálás

Első telepítésnél a `NORMAL` módot érdemes választani, mivel ez az alapvető rendszerösszetevőket automatikusan telepít és a választható összetevőkről külön bekéri a felhasználó válaszát. Ugyanez vonatkozik a programcsomagok telepítésére is.

Amint a programcsomagok telepítése is befejeződött, elkezdődik a rendszer konfigurálása, ami különböző paraméterek beállítását és kapcsolatok létrehozását jelenti.

A menürendszerben a Linux Loader telepítésére is lehetőség van.

A telepítés befejeződése után a program automatikusan visszatér a főmenübe, ahonnan a felhasználó kiléphet. Ezzel tehát a Slackware disztribúció telepítése befejeződött és a számítógép újraindítható.

## 5.5 BETÖLTŐLEMEZ KÉSZÍTÉSE

Ha a Linux rendszer telepítéséhez a behúzó kernelben speciális kezelőprogramra van szükség, például egy SCSI merevlemez-meghajtót kell elérni egy különleges host adapterrel vagy hálózati kártyával, akkor egyedül betöltő lemezt kell készíteni. Az előfeltételek egyike, hogy az adott kezelőnek már léteznie kell, valamint az, hogy már futó Linux rendszerhez kell hozzáérni azért, hogy a megfelelő kernelt elő lehessen állítani. A következő fejezet (6.2 szakasz) a kernel konfigurálását és fordítását tárgyalja, ezért itt csak a legfontosabb tudnivalókat foglaljuk össze tömörén.

Először egy olyan kernelt kell összeállítani, amely az összes szükséges kezelőprogramot tartalmazza. Azokra a perifériákra, amelyek nincsenek jelen, nem szabad kezelőprogramot fordítani, mivel ezzel nő a veszélye annak, hogy a kezelőprogramok összeütközésbe kerülnek. A Linux forráskódját tartalmazó könyvtárban a make config parancsot kell futtatni. (Az újabb linux kernelekben a make menuconfig parancs is használható, ezzel egy menüs rendszeren keresztül konfigurálható a kernel – a lektor.)

Ezután a kernel már újrafordítható. A make dep parancs először meghatározza a kernel önálló forráskód fájljai közötti függőséget, majd minden olyan régi object fájlt, amely még jelen van, töröl a make clean parancssal. A kernel aktuális fordítása a kernel forráskódjának főkönyvtárában (pl. /usr/src/linux) levő make zImage parancssal indítható el. Eredményként a z Image nevű fájl keletkezik, amely a /usr/src/linux/arch/i386/boot alkonyvtárban van. A fordítás a hardvertől függően 15 percetől óráig is eltarthat. (Ma még igen jónak mondható az a gép, amely 15 perc alatt végez a kernel fordításával, pl. Pentium 100MHz, 32 MB memória, 1-2 óra várható el egy 486-os géptől, 8 MB memóriával, míg 4 MB memórián egy 386 SX gépkárak 8-10 órát is fordít – a lektor.)

```
nicetry:~# cd /usr/src/linux
nicetry:/usr/src/linux# make dep
...
nicetry:/usr/src/linux# make clean
...
nicetry:/usr/src/linux# make zImage
...
```

Ezután a rdev parancssal néhány kernel paramétert módosítani kell.

```
nicetry:/usr/src/linux# rdev zImage /dev/fd0
```

Ez a parancs a lemezen a gyökér fájlrendszert írja elő.

```
nicetry:/usr/src/linux# rdev -R zImage 0
```

Ez a parancs a rendszerbe logikailag beillesztendő gyökér fájlrendszer írható/olvashatónak deklarálja.

```
nicetry:/usr/src/linux# rdev -r zImage 1440
```

Ez a parancs a virtuális meghajtó (RAM disk) méretét 1.44 MB-ra állítja be (lásd még a 6.2 részt is), amely 3 1/4-es dupla sűrűségű lemeznek felel meg. 5 1/2-es lemez használatához az 1440 helyet 1200-at kell megadni.

```
nicetry:/usr/src/linux# rdev -v zImage 1
```

Ez a parancs biztosítja azt, hogy a kernel a videokártyát normál módban (80x25) inicializálja. Következő lépésként ezzel a kernel image fájllal egy betöltő lemezt kell elkészíteni. Ennek a leggyakoribb módja, hogy a Slackware behúzó lemezről másolatot készítünk, azt logikailag a rendszerbe kapcsoljuk (mount), majd a zImage fájlt vmlinuz néven rámásoljuk, majd a lemezt a rendszerről az umount parancssal leválaszjuk.

```
nicetry:/usr/src/linux# zcat net.gz | dd of=/dev/fd0 bs=8k
0+360 records in
0+360 records out
nicetry:/usr/src/linux# mount /dev/fd0 /mnt
nicetry:/usr/src/linux# cat arch/i386/boot/zImage >/mnt/vmlinuz
```

Ezután már a rendszer betöltése a lemezről végrehajtható. Amint az alábbi üzenet megjelenik, a gyökérlemez behelyezhető:

```
VFS: Insert root floppy and press ENTER
```

A <Return> billentyű lenyomása után a betöltés folytatódik. A képernyőn több hibaüzenet is megjelenhet, azonban ezek ebben a fázisban figyelmen kívül hagyhatók.

Rövid idővel ez után a bejelentkezési (login) prompt lesz látható:

```
(none) login: root
```

Ezután a lilo és a sync parancsokat kell beírni:

```
# lilo
Added ramdisk
Added drive2
Added mount
# sync
```

Amint a sync parancs végrehajtódik és a készenléti jel is megjelenik, a lemez a meghajtóból kivehető. Ez tehát a Slackware disztribúcióhoz készített új behúzó lemez.

## 5.6 A BETÖLTÉSI MŰVELET KEZELÉSE (BOOT MANAGER)

A Linux Loader (LILO) lehetővé teszi a Linux betöltését. Ha a merevlemezén több operációs rendszer is van, akkor a LILO az elindítandó rendszer kiválasztását is biztosítja, valamint a Linuxon kívül be tudja tölteni a DOS-t, az OS/2-t, vagy egyéb PC-UNIX változatot is, de ugyanezt a műveletet még egy második merevlemezről is képes elvégzni.

### Működtetés

A betöltő először a LILO karakterekkel jelentkezik be a képernyőn és meghatározott ideig lehetővé teszi, hogy az <Alt>, <Ctrl> vagy az <AltGr> billentyűk bármelyikének a lenyomásával a felhasználó elkerülje a standard konfigurálást, és egy másik partíciót vagy konfigurálást válasszon. Ezután a betöltő a boot : üzenetet jeleníti meg, amellyel a behúzandó rendszer megadását várja. A felhasználó a <Tab> billentyű lenyomásával választhat a képernyőn megjelenített lehetőségek közül:

```
LILO boot:  
linux linux-old dos  
boot: linux  
Loading linux
```

Ha a felhasználó a megadott időn belül nem választ, akkor a program az elsőként felsorolt operációs rendszert tölti be automatikusan.

### Konfigurálás

Az alábbiakban egy olyan jellemző telepítést mutatunk be, amelyben egy (E)IDE merevlemez-meghajtó szerepel és a Linux gyökér fájlrendszer a második partícióra lesz telepítve. Először az /etc/lilo.conf fájlt kell módosítani:

```
# LILO configuration file  
# generated by 'liloconfig'  
#  
# Start LILO global section  
boot = /dev/hda2  
#compact      # faster, but won't work on all systems.  
delay = 50  
vga = normal    # force sane state  
ramdisk = 0      # paranoia setting  
# End LILO global section  
# Linux bootable partition config begins  
image = /vmlinuz  
root = /dev/hda2  
label = linux  
read-only # Non-UMSDOS filesystems should be mounted read-only for  
checking  
# Linux bootable partition config ends  
# Linux bootable partition config begins
```

```

image = /vmlinuz.old
root = /dev/hda2
label = linux-old
read-only # Non-UMSDOS filesystems should be mounted read-only for
          # checking
# DOS bootable partition config begins
other = /dev/hd1
label = dos
table = /dev/hda
# DOS bootable partition config ends

```

Az első sor a LILO telepítésének helyét adja meg. A lehetséges választék az MBR (Master Boot Record) vagy a partíció kezdete. A legbiztosabb megoldás a LILO telepítése a Linux gyökér fájlrendszer elejére, majd ezen partíció aktiválása.

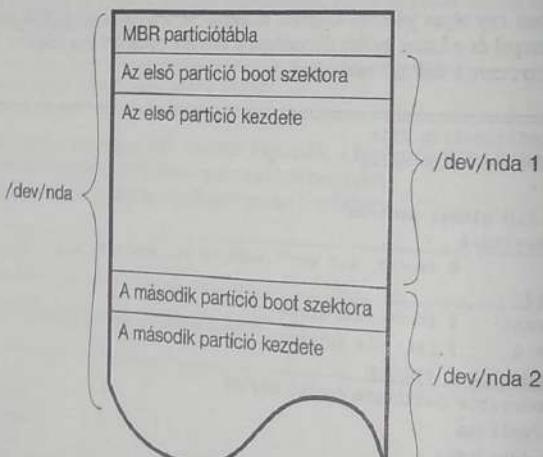
A LILO telepítése a MBR-be az eredeti DOS-os MBR felülírását jelenti. Ez azonban az fdisk /mbn parancssal visszadítható.

A root = /dev/hda2 azt írja elő, hogy a kernelnek a gyökér fájlrendszerét a /dev/hda2 partíció kell keresnie.

A compact optimalizálja a LILO működését megakadályozva, hogy a szektorokat egyesével olvassa be.

A install = /boot/boot.b egy opcionális beállítás, amely a boot szektor telepítését írja elő. Az alapítelmezés szerinti beállítás /boot/boot.b.

A message = /boot/message ugyancsak opcionális, és annak a szövegfájlnak a nevét írja elő, amelynek a tartalma még a prompt előtt jelenik meg. Ez a fájl a betöltésre vonatkozó lehetségekkel és az image fájlokkel kapcsolatosan tartalmaz megjegyzéseket.



5.6 ábra. A merevlemez logikai szerkezete

Másik leképező fájl előírásához a map = /boot/map definiálására van szükség. Itt a telepítés során a LILO a lehetséges image fájlokról és opcionális információt tárol. Az alapértelmezett érték /boot/map.

A delay tizedmásodpercekben azt az időintervallumot írja el, amíg a LILO-nak az első lehetőség automatikus kiválasztásáig egy billentyű lenyomására várnia kell.

Az image = /vmlinuz és a további sorok az első boot lehetőséget írják el. A betöltendő kernel image fájl a vmlinuz, amely a gyökérkönyvtárban van. Az elnevezés, amelyet a betölteshez be kell írni a linux read-only.

A read-only felülírja azt a kernel beállítást, amely megszabja, hogy a logikailag beillesztett gyökér fájlrendszer frontható/olvasható, vagy csak olvasható legyen.

A második búzási lehetőség a /vmlinuz.old fájl. Ez egy régebbi kernel fájl, amelyre akkor lehet szükség, ha az aktuális kernel nem működne megfelelően.

A harmadik behúzási alternatíva az első IDE merevlemezen levő DOS partíció. Az other címke biztosítja azt, hogy a LILO ne kísérélje meg az elindítást ebből a partícióból, hanem egy másik operációs rendszert töltön be.

A LILO konfigurációs fájl hasznos opcionális paramétere a append, amely után – a LILO boot készenléti jeléhez hasonlóan – a kernel vagy az init eljárás számára lehet paramétereket adni. Erre hangkártyák vagy Ethernet adapterekkel rendelkező rendszerek esetén van általában szükség. A következő példában egy olyan /etc/lilo.conf fájlrészlet látható, amely átírányítóként működő, és két Ethernet kártyával rendelkező számítógéphez készült:

```
boot = /dev/hda2
compact
delay = 50
vga = normal
ramdisk = 0
image = /vmlinuz
  root = /dev/hda2
  label = linux
  read-only
image = /vmlinuz.fw
  label = firewall
```

Az append sor az Ethernet kártyákat leképezi az eth0 és eth1 eszközökre, valamint megszakításra és az I/O portbeállításokra vonatkozó értékeket definiálja. A 6.2 szakasz a kernel paramétrek lehetséges beállítási lehetőségeit tekinti át.

## Kernel image fájlok

A kernel image egy olyan fájl, amely az operációs rendszer aktuális kerneljét tartalmazza az inicializáló programmal és a betöltővel. Ez a fájl, amelynek forráskódja a /usr/src/linux alkönyvtárban van (6.2 szakasz), a kernel fordítása során jön létre.

Annak érdekében, hogy a behúzás a kernel image fájlból lehetséges legyen, a nevét a LILO konfigurációs fájlból be kell írni, és a betöltőt újra kell telepíteni.

Az új kernel fordítása után a kernel image régi verzióját is be kell írni a konfigurációs fájlból. Ugyanis abban az esetben, ha a kernel konfigurálása és fordítása hibás, akkor még mindig van arra lehetőség, hogy a rendszert a régi kernellel betöltve a hibát ki lehessen javítani.

Ezt a megközelítést a kernel makefile funkciója is támogatja. A kernel /usr/src/linux forráskód könyvtárban a make zlilo zlilo hívással a komplálás után az új kernel image fájl a /vmlinuz könyvtárba másolódik és a régi fájl a /vmlinuz.old nevet kapja.

### A betöltő aktiválása

A konfigurációs fájlból levő bejegyzések aktiválására a LILO-t újra kell telepíteni, és el kell indítani az /etc/lilo alkonyvtárban levő install szkriptet, amely az aktuális betöltőt a hozzá tarozó aktuális konfigurációval a boot szektorba vagy az MBR-be írja.

```
nicetry:# lilo
Added linux
Added linux.old
Added DOS
nicetry:#
```

### A LILO eltávolítása

Ha a LILO telepítésére az előzőekben leírtaknak megfelelően a Linux gyökér fájlrendszer particiójában került sor, akkor az előzőekben aktiv partíció visszaállításához elegendő a LILO telepítésének visszavonása.

Ha a LILO telepítésére az MBR-ben került sor, és valamilyen oknál fogva innen ezt el kell távolítani, akkor az MBR előző tartalmát vissza kell állítani. Ha a DOS egy másik merevlemezen vagy particióban is rendelkezésre áll, akkor a legegyszerűbb megoldást a DOS fdisk programjának elindítása adja az /mbr kapcsolóval. Ez a parancs telepíti az új MBR-t és felülírja a LILO-t.

### Alternatív boot manager

Ha a LILO-tól eltérő boot manager telepítése szükséges, például az OS/2 boot manager, akkor a LILO-t nem szabad telepíteni az MBR-be, különben mint első betöltési lehetőséget, a rendszer mindenképpen elindítja.

A particiók felállításához az fdisk OS/2 verzióját kell használni. Ezután az OS/2 boot manager már inicializálható és a boot partició aktiválható. Ezután a Linux particiót, ahová a Linux sa is megtörénik. Innen a felhasználó már kiválaszthatja az OS/2-öt vagy a LILO-n keresztül elindíthatja a Linux rendszert.

# KONFIGURÁLÁS

**A**mint a fájlrendszer merevlemezre másolása valamint a Linux Loader telepítése megtörtént, néhány konfigurációs fájl megfelelő beállítására is szükség van, amellyel az operációs rendszer a rendelkezésre álló hardverhez a legmegfelelőbb módon állítható be.

## 6.1 ÁLTALÁNOS KONFIGURÁLÁS

A konfigurációs fájlok legtöbbjét a `/etc` könyvtár tartalmazza. E fájlok közül soknak állandó a tartalma és rendszerint meg sem kell változtatni. E könyv függeléke a fájlok tömör leírását tartalmazza. Ez a fejezet viszont azokat a konfigurációs módosításokat tárgyalja, amelyeket a Linux Loader PC-re való telepítése után végre kell hajtani.

### Fájlrendszerek

Amint a rendszer elindul, az `rc` szkriptek (lásd a 7.2. részt) közül az egyik elindítja a `mount` parancsot, amely az összes fájlrendszert logikailag a rendszerbe illeszti. A parancsot rendszerint a `-a` opcióval hajtják végre, amely az `/etc/fstab` fájlból előírt összes fájlrendszert automatikusan bekapcsolja a könyvtárfába. E fájlnak az összes rendelkezésre álló fájlrendszer fel kell sorolnia, beleérte azokat is, amelyeket a felhasználó nem szeretne automatikusan a rendszerbe illeszteni. Azokhoz a fájlokhoz, amelyeknél nincs szükség a rendszerbe való automatikus beillesztésre, a `noauto` opciót kell megadni, amely a `mount -a` parancsot automatikusan felülírja. Tehát így védhető ki, hogy az automatikus beillesztés (`mount`) az összes fájlrendszerrre automatikusan vonatkozzék.

A következő példa, amely az `/etc/fstab` fájlból származó részlet, olyan bejegyzéseket tartalmaz, amelyeknél nemesak a `/home`, hanem az `/usr` és a `/var` is önálló partícióra voltak telepítve:

<code>/dev/hda2</code>	<code>/</code>	<code>ext2</code>	<code>defaults</code>	1	1
<code>/dev/hda3</code>	<code>none</code>	<code>swap</code>	<code>defaults</code>	1	1
<code>/dev/hda4</code>	<code>/usr/local</code>	<code>ext2</code>	<code>defaults</code>	1	1
<code>/dev/sda2</code>	<code>/opt</code>	<code>ext2</code>	<code>defaults</code>	1	1
<code>/dev/hdc</code>	<code>/cdrom</code>	<code>iso9660</code>	<code>defaults,user,noauto,ro</code>	0	0
<code>none</code>	<code>/proc</code>	<code>proc</code>	<code>defaults</code>	1	1

A bejegyzések nagyrészt a `mount` parancs paramétereire vonatkoznak. Az első oszlop a logikailag rendszerbe illesztendő eszközt tartalmazza. A második oszlop jelzi az **útvonalat** ahhoz a beillesztési ponthoz (**mountpoint**), vagyis ahhoz a meglevő könyvtárhoz, amelyhez az új fájlrendszert logikailag be kell illeszteni. A harmadik oszlop a fájlrendszer típusát jelenti.

A legtöbb fájlrendszer további opciók előírását is lehetővé teszi, amelyeket az utolsó oszlophoz kell megadni. Ha nincs szükség különleges paraméterekre, akkor a szabványos beállítások érvényesítéséhez itt a `defaults` opciót kell szerepeltetni. A lehetséges további beállításokat a könyv Referencia része tartalmazza.

A /proc fájlrendszerbe beillesztő bejegyzés fontos, mivel ettől a könyvtártól sok parancs – például a ps változatai – függ.

### A csereműveletek helye

Ha a célszámítógép 8 Mb vagy ennél kevesebb RAM-mal rendelkezik, akkor – ha erre a telepítés során még nem került sor – cserepartíció vagy cserefájl létrehozására van szükség. Ellenkező esetben ugyanis a számítógépen nem valósítható meg több program párhuzamos futtatása grafikus felhasználói környezetben. 4 Mb memória még arra sem elég, hogy egy szövegszerkesztő futtatása mellett a kernelt fordítani lehessen. Még 16 Mb memória esetén is érdemes a virtuális memóriát cserepartícióval bővíteni.

A swapon parancs aktivizálja a csereművelethez felhasználható helyet (lásd 5.4 részt). A cserepartíciót az /etc/fstab fájban elhelyezve az a rendszer indításakor automatikusan létrejön. Az előző példa egy ilyen bejegyzést is tartalmazott.

Bizonyos esetekben cserepartíció helyett érdemesebb cserefájlt használni, amelyet a megfelelő méretekben a dd parancs felhasználásával kell először létrehozni. A következő példa egy ilyen cserefájl létrehozását és aktivitását mutatja:

```
nicetry:~# free
              total        used         free       shared      buffers
Mem:      14892       14664        228        3788       9104
-/+ buffers:          5560       9332
Swap:          0          0          0
MemTotal:     14
MemFree:      0
MemShared:    3
Buffers:      8
Cached:       1
SwapTotal:    0
SwapFree:     0
nicetry:~# mkswap /swapfile
bash: mkswap: command not found
nicetry:~# mkswap /swapfile
Setting up swapspace, size = 8384512 bytes
nicetry:~# swapon /swapfile
nicetry:~# free
              total        used         free       shared      buffers
Mem:      14892       13580       1312        3892       7188
-/+ buffers:          6392       8500
Swap:          8188          0       8188
MemTotal:     14
MemFree:      1
MemShared:    3
Buffers:      7
Cached:       2
SwapTotal:    7
SwapFree:     7
nicetry:~#
```

A cserefájlokat a `/etc/fstab` fájlban is meg lehet adni, amelyeket aztán a `swapon -a` parancssal lehet automatikusan aktiválni. A következő példában a cserefájl megadására egy eszköz helyén kerül sor:

<code>/swapfile</code>	<code>none</code>	<code>swap</code>	<code>defaults 1 1</code>
------------------------	-------------------	-------------------	---------------------------

## Bejelentkezés

Azoknál a rendszereknél, amelyek támogatják az árnyék jelszót, a felhasználói jelszóra vonatkozó általános beállításokat az `/etc` könyvtárban levő `login.defs` fájl tárolja. A fájlból elvégzett beállítástól függ például, hogy a bejelentkezési prompt letiltására mennyi sikertelen próbálkozás után kerüljön sor.

Az ide vonatkozó megjegyzések az `/etc/login.defs` fájlból, vagy a megfelelő kézirány-oldalon találhatók meg. A következő példa egy rövid részletet mutat ebből a fájlból.

```
# Delay in seconds for which the login prompt is disabled
# after entry of a wrong password
FAIL_DELAY 2

# Devices from which a login as root is permissible
# CONSOLE      /etc/consoles
# CONSOLE      console:tty01:tty02:tty03:tty04
CONSOLE        ttym:tty2:tty3:tty4:tty5:tty6:tty8

# Files to be output after a login
MOTD_FILE     /etc/motd
```

Az olyan egyéb csomagok, amelyek nem támogatják az árnyék jelszót, általában nem tartalmazzák ezt a fájlt. Ezek a megfelelő paramétereket a fordítás során állítják be.

## A billentyűzetkiosztásra vonatkozó konfigurálás

A Linux kernel 0.99 verziójától kezdődően a billentyűzetre vonatkozó beállítások nem kötődnek már a kernel fordításához. A beállítások ugyanis futási időben is megváltoztathatók. Ezt a célt a `loadkeys` parancs szolgálja, amely a billentyűzetkiosztást tartalmazó táblázatot betölti. A rendszer indításakor ezt a parancsot az `rc` fájlok egyikéből kell elindítani. Például a német nyelvterületen dolgozó felhasználók a megfelelő billentyűzetkiosztáshoz a `gr-latin1.map` fájlt használják, amely a szokásos német billentyűzetet adja olyan umlauttal, amelyet az ISO Latin-1 szabvány definiál. Hasonló billentyűzet kiosztás létezik a spanyol, francia és egyéb nyelvek esetében is.

Annak érdekében, hogy a Linux használata minél kényelmesebb legyen, programcsomagtól független a megfelelő ország-specifikus billentyűzetkiosztást kell kiválasztani a programcsomagtól független a `/usr/lib/keymaps`, `/usr/lib/kbd/keymaps` vagy az `/etc/keymaps` könyvtárból. A rendszer betöltésekor a megfelelő billentyűzetkiosztás betöltéséhez az `rc` fájlok egyikében a `loadkeys` parancsot kell kiadni. A következő részlet, amely az `/etc/rc.d/rc.local` fájlból származik, a német billentyűzetkiosztás betöltésére mutat példát:

```
# loading country specific keyboard layout
/usr/bin/loadkeys /usr/lib/kbd/keytables/iso-latin2.map
```

## 6.2 A KERNEL

A parancsokhoz, segédprogramokhoz és egyéb Linux programokhoz hasonlóan a kernel forrás-kódja is ingyenesen hozzáférhető. A kernal főleg C nyelven írták, bizonysos részeit pedig assemblyben. Az ütemezőkön kívül, amely a processzek közötti átkapcsolást kezeli, a kernel-pérférához és a fájlrendszerhez is kezelőprogramokat tartalmaz.

### A kernel konfigurálása

Az operációs rendszer telepítése után a következő lépés a kernel konfigurálása és fordítása, amely bizonysos esetekben elhagyható. Azonban annak érdekében, hogy a rendszer az adott hardveren optimálisan működjön, a **fordítás** mindenkorban javasolt. Ez ugyanis lehetővé teszi az adminisztrátor számára, hogy a felesleges kezelőprogramoktól megszabaduljon, felvegye a megfelelő új kezelőprogramokat és módosítsa az azokra vonatkozó beállításokat. Így csökkentheti a kernel memóriaigényét és gyorsíthatja a rendszer betöltését.

A kernel forráskódját általában a `/usr/src/linux` könyvtár tartalmazza, amelyben egy olyan konfigurációs szkript is van, amelyet a `makefile` indít el, és amely jelentősen leegyszerűsíti a kernel egyedi igények szerinti beállítását. A szkript, amelyet a `make config` parancssal lehet elindítani, a következő beállítások elvégzését teszi lehetővé:

- Támogatott fájlrendszerek
- TCP/IP támogatás
- Társprocesszor emulátor használata
- A tárgykód optimalizálása 80486-os vagy Pentium processzorokra
- SCSI támogatás
- SCSI-re és a hálózati kártyákra vonatkozó kezelőprogramok
- Paraméterbeállítások különleges interfészkártyákhoz
- Hangkártyákra vonatkozó beállítások

Ha a kernel nem képes felismerni olyan speciális paramétereket, mint az I/O cím vagy egy kártya megszakítási száma, és ezeket elő kell irni, van amikor nem érdemes ezeket a vonatkozó kezelőprogram forráskódjában megváltoztatni. Inkább a kernelnek kell átadni a betöltés során. Ezeket a paramétereket a LILO promptnál kezelni kell, vagy az `append` utasítással (lásd az 5.6. részt) az `/etc/lilo.conf` fájlból. A kernelnek jelezni kell, ha a számítógép például két Ethernet-kártyával rendelkezik, mivel szokásos esetben csak egy hálózati kártya jelenlétére számít.

### Fordítás

Amint az összes szükséges beállítás megtörtént, a forráskód különböző részei közötti függőségeit ismét meg kell határoznai. Az eljárás a `make dep` meghívásával kezdődik. Ehhez a kernel forráskódjának fókonyvtárába (`/usr/src/linux`) kell lépni. Mivel a régi object-fájlok még mindig jelen lehetnek akkor, amikor a konfiguráció módosítására sor kerül, ezeket először a `make clean` parancssal el kell távolítani.

Ezután már megkezdődhet az aktuális fordítási eljárás. Itt a `Makefile` központi szerepet játszik, ahogy a legtöbb más program telepítése és fordítása során is: ez tartalmazza ugyanis a függ-

sségi információkat az önálló forráskódú fájlokra vonatkozóan, és segít koordinálni a különböző szkripteket a kernel telepítésekor és konfigurálásakor.

A szokásos művelet során egy tömörített kernel generálódik, amelyet egy beépített rutin a betöltés során automatikusan kibont. A Linux-kernelnek ez a tulajdonsága teszi lehetővé, hogy a Linux viszonylag kis méretű legyen, azonban minden hardverösszetevőt (hálózatot, sztrímet, SCSI eszközök, CD-ROM-ot) támogasson. A tömörítés megszüntetéséhez szükséges idő nem jelentős.

A következőben a ma ke parancs hívásának legfontosabb változatait soroljuk fel.

- **make dep** – Újból meghatározza a függőségeket a forráskódú fájlok között. A műveletet a kernel konfigurációjának módosítása után minden el kell végezni.
- **make clean** – Törli az összes object-fájlt. A következő fordítás során az összes forráskódú fájl fordítása teljesen a kezdettől megvégbe.
- **make** – Csak a kernelt fordítja. Kernel image nem jön létre.
- **make zImage** – Lefordítja a kernelt és létrehoz egy tömörített betölthető **kernel image**-fájlt a `/usr/src/linux/arch/i386/boot` könyvtárban.
- **make zlilo** – Az előzőek szerint létrehoz egy kernel image-fájlt és azt a `/vmlinuz`-ba másolja. Ha ugyanezen a néven létezik már egy régi fájl, akkor azt `/vmlinuz.old` néven tárolja. Ezután elindítja a Linux Loader telepítő szkriptjét így a következő rendszerindításkor már az új kernelt lehet behúzni.
- **make disk** – Az image-fájlt közvetlenül az A meghajtóban levő lemezre írja a komplilálás befejezése után. A parancs végrehajtása előtt egy üres formázott lemezet kell az A meghajtóba helyezni.

### Konfigurálás az rdev programmal

A kernel több paraméterét az image-fájlból még a komplilálás után is be lehet állítani, például a gyökérpartíció helyét vagy a virtuális lemez (RAM disk) méretét. A beállításokat elvégző program az `rdev`, amelyet a `-help` paraméterrel meghívva az összes lehetséges paraméter megjelenik a képernyőn. Az alábbiakban a program hívásának legfontosabb változatait foglaljuk össze:

- **rdev-help** – Megjeleníti a parancsal használható összes paramétert.
- **rdev <image file> <Root device>** pl. `rdev /vmlinuz /dev/hda1` – Előírja azt az eszközt, amelyről a gyökér-fájlrendszer logikai beillesztése történik a megadott kernel image-fájli behúzása során. Ha az előírt image fájlt a LILO tölti be, akkor az itt megadott beállítást a LILO konfiguráció általában felülírja. Az `/etc/lilo.conf` fájlból levő `root=devi-ce` alakú bejegyzés így felülírja a kernelben levő közvetlen bejegyzést. Ez a beállítás a kernelben akkor fontos, ha az image-fájlt a `dd` parancssal közvetlenül lemezre írjuk.
- **rdev <image file> pl. `rdev /dev/fd0`** – Kijelzi a image-fájlból beállított eszközt, mint a gyökér-fájlrendszer eszközét.
- **rdev -R <image file> <Flag>** pl. `rdev -R /vmlinuz 1` – Ez az opció azt határozza meg, hogy a kernel a gyökérpartíciót csak olvashatóként, vagy írható/olvashatóként kapcsolja be a rendszerbe. Sok Linux programcsomag azt feltételezi, hogy a kernel a gyökérpartíciót először csak olvashatóként illeszti a rendszerbe, így a fájlrendszer a betöltés során az `fsck` parancssal lehet ellenőrizni. Az ellenőrzés után a fájlrendszer ismét be kell illeszteni a rendszerbe, de most már írható/olvashatóként. Ha a kernel a betöltés során a gyökérpartíciót nem illeszti be csak olvashatóként, akkor a fájlrendszer ellenőrzése nem lehetséges. Az 1-es érték a csak olvasható, a 0 az írható/olvasható módot jelenti.
- **rdev -r <image file> <ramdiskSize>** pl. `rdev -r /vmlinuz 1440` – A virtuális memória (RAM disk) méretének beállítását teszi lehetővé. Ezt az opciót elsősorban betöltő lemezknél használják, ahol a gyökérpartíció eszközét `/dev/fd0`-ra, a virtuális memória méretét pedig 1440-re állítják, ami a  $3\frac{1}{4}$ -es hajlékony lemez kapacitásának felel meg. A betöltés

során a kernel a betöltő lemez tartalmát betölti a virtuális memóriába, amelyet mint gyökérfájlt rendszert kapcsol logikailag a rendszerbe.

- rdev -v <image file><VideoMode> pl. `rdev -v /vmlinuz -1` – Beállítja azt a video-módot, amellyel a kernel a videokártyát inicializálja. A -1 a normál felbontást jelenti, míg a -3 előírásra a kernel az indítás során a felbontást bekéri a felhasználótól.

### Paraméterek átadása a kernelnek

A behúzás során manuálisan, illetve a Linux Loader konfigurációs fájlján keresztül paraméterek adhatók át a kernelnek, amelyekkel nemesak a kezelőprogramok betöltése, hanem például az SCSI host-adatpter számára szóló paraméterek is megadhatók. Az alábbiakban a legfontosabb paraméreket és azok funkcióját soroljuk fel:

- root=device – Megadja, hogy a kernel melyik eszközről kapcsolja be a gyökérfájlrendszerit.
- ro – Definíálja, hogy a gyökérfájlrendszer csak olvashatóként kapcsolódjon a rendszerhez.
- rw – Megadja, hogy a gyökérfájlrendszer írható/olvashatóként kapcsolódjon a rendszerhez.
- debug – A kernelben belüli hibakeresési szintet 10-re állítja.
- no-hlt – Kikapcsolja a hlt utasítás meghívását a kernel inaktív ciklusában.
- no387 – Előírja, hogy a kernel nem használhatja a matematikai társsprocesszort, csak a saját emulációját, amelyet a kernelbe kell befordítani.
- reserve=IO-addr, length, IO-addr, length ... – Megakadályozza az eszközkezelőket abban, hogy elérjék az előírt I/O portok tartományát és arra készíti, hogy egymástól függetlenül kereszenek a betöltés során adaptereket.
- ramdisk=kilobytes – A virtuális memória méretét az előírta állítja be.
- ether=IRQ, IO-addr, P1, P2, device – A telepített Ethernet-kártyához tartozó kezelőprogramok számára paraméterek átadását teszi lehetővé a kernelben. A P1 és P2 jelentése az adott kezelőprogramról függ. Itt általában 0-át adnak meg. A device annak az Ethernet eszköznek a nevét tartalmazza, amelyre a paraméter vonatkozik. Az alapértelmezés eth0.
- hd=cylinders, heads, sectors – A merevlemez szerekzetét definíálja. Erre az opcióra csak akkor van szükség, ha a merevlemez nem ismerhető fel.
- st=bufferSize, WT, MaxTapeBuffer – Beállítja a paramétereket az SCSI sztrímerhez. A paraméterek pontos jelentése a kernel forráskódjának drivers/scsi könyvtárában a README.st fájlból található meg.
- bmouse=IRQ – Beállítja az IRQ-t a bus-os egérhez.
- max\_scsi\_luns=No – Beállítja a legnagyobb logikai egység számát az SCSI eszközökhöz.
- st0x=ROM-Addres, IRQ – Beállítja a paramétereket a Scagate st01 és st02 host-adapterekhez.
- tmc 8xx=ROM-addr, IRQ – Beállítja a paramétereket a Future Domain TMC8xx adapterhez.
- t128=ROM-addr, IRQ – Beállítja a paramétereket a T128 host-adapterhez.
- pas16=IO-addr, IRQ – Beállítja a paramétereket a PAS 16 host-adapterhez.
- ncr5380=IO-addr, IRQ, DMA – Definiálja a paramétereket az SCSI host-adapterhez, amely az NCR 5380 áramkörű lapkával rendelkezik.
- aha152x=IO-addr, IRQ, SCSI-Id, Reconnect, Parity, Debug – Beállítja a paramétereket az Adaptec 152x kontrollerhez.
- med=IO-addr, IRQ, Mitsumi\_Bug\_Wait – Beállítja a paramétereket a Mitsumi CD-ROM-meghajtóhoz.
- sound=0xTaaalld – Beállítja a paramétereket a hangkártyához. A paramétereket egyetlen hexadecimális szám azonosítja. A számjegyeleknek a következő jelentéstük van: T a kártya típusa, amely a következő értékeket vetheti fel:
  1. FM Synth. (YM3812 vagy OPL3)
  2. Soundblaster 1.0-ig 2.0-ig, Soundblaster Pro and 16

3. Pro Audio Spectrum 16
4. Gravis UltraSound
5. MPU-401 UART Midi
6. SB16 16 bites DMA számmal
7. SB16 Midi (MPU-401 emulációban)

**aaa** – az I/O cím

**I** – IRQ

**d** – DMA csatorna (0,1,3,5,6 vagy 7)

- **sbpcd=IO-Address, Typ** – Beállítja a paramétereket a Soundblaster/Panasonic CD-ROM-meghajtóhoz.
- **cdu31a=IO-Address, IRQ** – Beállítja a paramétereket a Sony CDU-31A CD-ROM-meghajtóhoz.

## 6.3 DÉMONOK

Mivel a démonokhoz nincs terminál hozzárendelve, ezek általában nem küldenek hibaüzeneteket, hanem az összes **hibaüzenetet** és egyéb jelentést a **syslog** démonra küldik. Annak érdekében hogy a démonok konfigurálásában és az egyéb programokban a hibák felderítése minél egyszerűbb legyen, az alábbiakban ismertetett módon kell a **syslog** démont konfigurálni.

(Mint rendesen, ennek a démonnak is számos verziója van a Linux alatt. A következőkben a **syslogd/klogd** kombinációt fogjuk ismertetni, amely sokkal könnyebben konfigurálható, mint a normál BSD **syslogd**.)

### Syslog démon

A syslog démon **syslogd** a másik démontól kapott üzenetet **naplófájlba** tudja írni, el tudja küldeni **elektronikus postán** bizonyos felhasználóknak, vagy közvetlenül meg tudja jeleníteni a konzolon. A beállításokat, amelyek az egyes egységekre önállóan elvégezhetők, az **/etc/syslog.conf** fájlból lehet előírni.

Az egyik lehetséges megoldás az, amely a különlegesen fontos üzeneteket a konzolra küldi, és az összes egyéb üzenetet, a **prioritásuknak** megfelelően egy naplófájlba. Ezen elrendezéshez az alábbi bejegyzésekkel kell az **/etc/syslog.conf** fájlból elhelyezni:

```
# /etc/syslog.conf
# For info about the format of this file, see "man syslog.conf" (the BSD man
# page), and /usr/doc/sysklogd/README.linux.
#
# NOTE: YOU HAVE TO USE TABS HERE - NOT SPACES.
# I don't know why.
#

*.=info;*.=notice          /usr/adm/messages
*.=debug                     /usr/adm/debug
*.warn                       /usr/adm/syslog
*.error                      /usr/adm/syslog

#
# This might work instead to log on a remote host:
# *                         @hostname
```

A bejegyzések az egység specifikációjától és prioritásától valamint az üzenetek helyének megjelöléséből állnak. Az összes definított egység és prioritás áttekintése a kézikönyv *syslogd* és *syslog.conf* fájlakra vonatkozó részében található meg.

A */var/log* könyvtárban a fájloknak már létezniük kell, amikor a *syslog* démon elindul. Üres fájlt a legegyszerűbben a *touch* parancs és a megfelelő fájl paraméterként való megadásával lehet létrehozni.

Annak ellenőrzésére, hogy a konfigurációs fájl beállításai helyesek-e, a *syslog* démont le kell állítani a *kill -1* parancssal, majd újra kell indítani a *-d* opcióval, amely a démont hibakereső módban indítja el. A *syslog* ezután kijelzi egy mátrixot, amely megadja, hogy melyik üzenet melyik fájiba frödött vagy melyik felhasználóhoz jutott el.

A rendszерadminisztrátornak kell gondoskodnia arról, hogy a naplófájlok mérete ne legyen túlságosan nagy. Javasolt egy olyan szkript használata, amely rendszeres időközönként a naplófájlokat egy másik könyvtárba helyezi át, majd azokat újból létrehozza. Ezután a *syslog* démon számára jelezni kell - */etc/syslogd.reload* -, hogy a naplófájlokat újra nyissa meg. Ha később valamilyen probléma adódna, a régi naplófájlohoz még mindig vissza lehet térti.

### Nyomtató démon

A nyomtatók általános definiálása és konfigurálása az */etc/printcap* fájlból történik. Ebben a fájlból adható meg például az, hogy az egyes ijj nyomtatási munkákhoz tartozzon-e kezdőlap, vagy hogy a nyomtatás befejezésekor legyen-e lapdohás. Többszörös nyomtatási sorok is létrehozhatók, mindegyikhez saját szűrőprogramot rendelve.

Annak érdekében, hogy a kliens számítógépek el tudják érni a nyomtató szervereket, amelyek a hálózaton keresztül a nyomtatókat biztosítják, a kliens számítógépet a nyomtató szerver */etc/hosts.lpd* fájljában meg kell adni. Ez a fájl az összes olyan számítógép nevét tartalmazza, amelynek hozzáférése van a nyomtatóhoz. A következő példa egy Linuxos számítógép */etc/printcap* fájliját mutatja, amely nem rendelkezik saját nyomtatóval, hanem egy munkaállomás nyomtatóját éri el:

```

# hosts.lpd      This file describes the names of the hosts which are
#                   allowed to use the remote printer services of this
#                   host. This file is used by the LPD subsystem.
#
# Version:        @(#) /etc/hosts.lpd    2.00   04/30/93
#
# Author:         Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#
# hades.demo.de
# hermes.demo.de
# jupiter.demo.de
#
# End of hosts.lpd.

```

A következő példában egy olyan Linux gép */etc/printcap* nevű fájlja látható, amely nem rendelkezik saját nyomtatóval, azonban a *zeus* nevű munkaállomás nyomtatójához hozzáfér:

```
# typical remote printer entry
lp:lp=:rm=zeus.demo.de:sd=/usr/spool/zeus:lf=/usr/adm/lpd-errs:
```

A printcap fájl minden sora nyomtatási sort definiál. Az önálló opciókat kettőspontok zárfák le, így általa az első attribútum a nyomtatási sor (lp) első attribútumát írja el. Ha a bejegyzés hosszabb egy szornál, akkor azt a sort, amely után folytatás következik, backslash karakterrel kell lezární. Ebben az esetben a kettőspontot a folytatás elejére is ki kell tenni.

A következő táblázat a printcap parancs legfontosabb opciót sorolja fel. A kézikönyv megfelelő oldalán azonban még részletesebb információ található. A szöveges opciók végét = jel, a numerikusaké特 jel zárja le.

lp=	Kijelöli a nyomtató interfészt (az alapértelmezés /dev/1p)
rm=	A távoli host neve, amely nyomtató szerver
rp=	A távoli nyomtatási sor neve (az alapértelmezés lp)
sd=	A lokális spool könyvtár neve
if=	A bemeneti szűrő neve
of=	A kimeneti szűrő neve
mc#	A dokumentum másolatainak maximális száma
mx#	A munka maximális mérete blokkban; az mx#0 tetszőleges méretű munkát tesz lehetővé
sc	Letiltja a dokumentum többpéldányos nyomtatását
sf	Letiltja a lapdóbást az egyes nyomtatási munkák után
sh	Letiltja az azonosító oldalt az egyes nyomtatási munkák előtt

#### A printcap parancs legfontosabb opciói

A nemzeti karakterkészletek (pl. a francia, spanyol, német) esetében a szövegfájlok kivitele a nyomtatóra problémákat okozhat. A német nyelv esetében ilyen problémás karakter az umlaut. A megfelelő szűrő telepítése azonban kielégítő eredményhez vezet. Ilyen szűrőt számos módon meg lehet valósítani. A következő példa egy egyszerű C program forráskódját mutatja, amely az umlautokat konvertálja és a nyomtatóra minden sor után kocsi vissza (CR) karaktert küld. A tr UNIX parancs is ilyen konverziós célokat szolgálhat.

A /etc/printcap fájlba ilyen szűrőt az if (input filter) vagy az of (output filter) szűrőkön keresztül lehet csatolni. Az output szűrő inicializálására csak egyszer kerül sor a nyomtatási munkákra várakozáskor, míg az input szűrő minden egyes nyomtatás esetén újra indul.

```
# Generic printer:
lp:lp=/dev/lp1:sd=/usr/spool/lp1:sh:sf:mx#0:
# text queue
txt:lp=/dev/lp1:sd=/usr/spool/txt:sh:sf:mx#0:\n
      :if=/usr/spool/lp/epson:
# Postscript queue
ps:lp=/dev/lp1:sd=/usr/spool/ps:sh:sf:mx#0:\n
      :if=/usr/spool/lp/Postscript
```

Szűrőn kereszttüli csatolás a /etc/ printcap fájlba

Több nyomtatási sor nyilvántartása lehetővé teszi a szűrők közötti szükség szerinti átkapcsolást. A megfelelő sor előirását az lpr parancs paramétereiben lehet előírni. A következő parancs ékezeti szövegek nyomtatását teszi lehetővé, amennyiben a fenti példa szerint a /usr/spool/ /lp/epson szűrő megfelelően konvertál:

```
nicetry:-$ lpr -Ptxt ekezet.txt
```

A megfelelő szűrő alkalmazva még a hagyományos mátrix- és tintasugaras- vagy egyszerű lézernyomtatóval is megoldható olyan feladat, amelyhez egyébként PostScript nyomtatóra lenne szükség. A Ghostscript nevű PostScript interpreter szűrként van regisztrálva, amelyet közvetlenül nem, csak a shell szkripten keresztül lehet alkalmazni:

```
#!/bin/sh
#
# Postscript printer filter
#
DEVICE=epson

exec /usr/bin/gs -q -dPAPERSIZE=a4 -dSAFER \
      -sDEVICE=$DEVICE -sOutputFile=- -
```

### PostScript szűrő

Az előbbi példák azt feltételezik, hogy a nyomtató Epson kompatibilis. Egy másik nyomtatótípus behelyettesítésre az sDevice paraméter megfelelő beállításával érhető el. A Ghostscript – gs – az alábbi lehetőségeket biztosítja:

```
nicetry:-$ gs -help
Aladdin Ghostscript 4.01 (1996-7-10)
Copyright (C) 1996 Aladdin Enterprises, Menlo Park, CA. All rights
reserved.
Usage: gs [switches] [file1.ps file2.ps ...]
Most frequently used switches: (you can use # in place of =)
-dNOMPAUSE          no pause after page | -q     'quiet', fewer messages
-g<width>x<height>  page size in pixels | -r<res>  pixels/inch resolution
-sDEVICE=<devname>   select device    | -sOutputFile=<file>
-c quit              |           select output file:
                     (as the last switch) | - for stdout, !command for pipe,
Input formats: PostScript PostScriptLevel1 PostScriptLevel2 PDF
Available devices:
x11 x11alpha x11cmyk x11mono deskjet djet500 laserjet ljetplus ljet2p
ljet3 ljet4 cdeskjet cdjcolor cdjmono cdj550 pj pjl pjl1300 bj10e bj200
bjc600 bjc800 faxg3 faxg32d faxg4 epson eps9high pcxmono pcxgray pcx16
pcx256 pcx24b phm phmraw pgm pgmraw pgm pgnmraw pnm pnmlraw ppm ppmraw
```

```
tiffcrle tiffg3 tiffg32d tiffg4 tifflw tiffpack tiff12nc tiff24nc psmono
bit bitrgb bitcmyk pngmono pnggray png16 png256 png16m pdfwrite nullpage
Search path:
. : /usr/local/share/ghostscript/4.01 :
/usr/local/share/ghostscript/fonts
For more information, see /usr/local/share/ghostscript/4.01/doc/use.txt.
nicetry:-$
```

Az **aps** szűrőcsomag a többszörös nyomtatási sor létrehozására további lehetőséget nyújt. Ebben az esetben egyetlen szűrő szkript nyilvántartását végzi, amely aztán automatikusan felismeri a dokumentum típusát, amelyet nyomtatni kell, majd aktiválja a megfelelő szűrőt. Így egyetlen soron keresztül tetszőleges adattípus nyomtatása oldható meg:

```
nicetry:-$ lpr postscript.ps tex.dvi text.txt
```

Az **aps** szűrőre vonatkozó printcap fájl HP Deskjet-re vonatkozó része a következőhöz hasonló:

```
ascii|lp1|PS_300dpi-a4-ascii-mono|PS_300dpi ascii mono:\n
:lp=/dev/lp1:\n
:sd=/usr/spool/PS_300dpi-a4-ascii-mono:\n
:lf=/usr/spool/PS_300dpi-a4-ascii-mono/log:\n
:af=/usr/spool/PS_300dpi-a4-ascii-mono/acct:\n
:if=/usr/local/apsfilter/filter/aps-PS_300dpi-a4-ascii-mono:\n
:mx#0:\n
:sh:\n\n#\nlp|lp2|apple|PS_300dpi-a4-auto-mono|PS_300dpi auto mono:\n
:lp=/dev/lp1:\n
:sd=/usr/spool/PS_300dpi-a4-auto-mono:\n
:lf=/usr/spool/PS_300dpi-a4-auto-mono/log:\n
:af=/usr/spool/PS_300dpi-a4-auto-mono/acct:\n
:if=/usr/local/apsfilter/filter/aps-PS_300dpi-a4-auto-mono:\n
:mx#0:\n
:sh:\n\n#\nraw|lp3|PS_300dpi-a4-raw|PS_300dpi auto raw:\n
:lp=/dev/lp1:\n
:sd=/usr/spool/PS_300dpi-raw:\n
:lf=/usr/spool/PS_300dpi-raw/log:\n
:af=/usr/spool/PS_300dpi-raw/acct:\n
:if=/usr/local/apsfilter/filter/aps-PS_300dpi-a4-raw:\n
:mx#0:\n
:sh:
```

## 6.4 BEJELENTKEZÉS SOROS INTERFÉSZEN KERESZTÜL

Soros interfészben kerestüli bejelentkezés (login) konfigurálásához a szokásos getty helyett a mgetty parancsot érdemes használni, amely lehetővé teszi több szolgáltatás egyidejű működtetését egyenlő porton keresztül. A szokásos bejelentkezésen kívül faxok is küldhetők, illetve fogadhatók. A getty elnevezésű továbbfejlesztett változat pedig lehetővé teszi a Zyxel modem csatlakoztatásával is üzenetrögzítőként.

Az mgetty parancsot az init eljárásból szokás elindítani, amelyhez viszont az /etc/inittab fájlban a következő bejegyzés szükséges:

```
# start mgetty on port /dev/ttyS2
s2:456:respawn:/usr/local/sbin/mgetty -s 38400 ttyS2
```

## 6.5 FAX

A faxok küldése és fogadása az mgetty/sendfax csomagon keresztül konfigurálható.

### Fogadás

Az mgetty a bejövő faxot a /var/spool/fax/incoming könyvtárba menti G3 fájlként, majd a faxadmin nevű felhasználónak elektronikus postán keresztül üzenetet küld a fax beérkezéséről:

```
Subject: fax from (anonymous sender)
To: faxadmin
From: root (Fax Getty)

Pages received: 1

Communication parameters: +FCS:0,3,0,2,0,0,0,0
  Resolution : normal
  Bit Rate   : 9600
  Page Width : 1728 pixels
  Page Length: A4 (297 mm)
  Compression: 0 (id mod Huffman)
  Error Corr.: none
  Scan Time  : 0

Reception Time : 0:42

Spooled G3 fax files: /usr/spool/fax/incoming/fnfl3dbfc50-+49-9344-1636.01
regards, your modem subsystem.
```

Ha a `/usr/local/bin` könyvtár tartalmazza a `new_fax` nevű szkriptet, akkor az a fax sikeres fogadása után végrehajtódik. Itt más grafikus formátumba konvertálás vagy nyomtatási műveletek is végrehajthatók. A ppm programcsomag a G3 tetszőleges formátumba való konvertálását teszi lehetővé.

```
#!/bin/sh
# new_fax: convert incoming g3 faxes to GIF
#
shift 3

for i in $*
do
    /usr/local/netpbm/g3topbm $i |
    /usr/local/netpbm/pbmtogif $i >$i.gif
    rm $i
done
```

A beérkező faxok megjelenítésére szolgáló szkript például a következő lehet:

```
#!/bin/sh
# faxview - display incoming faxes
#
xloadimage -geometry 1000x720+10+10 -xzoom 50
/var/spool/fax/incoming/*.gif
```

## Küldés

A faxok küldését a `sendfax` parancssal lehet elvégezni. A küldeni kívánt fájlnak azonban G3 formátumban kell lennie. PostScript fájlokat a Ghostscript segítségével lehet a legkönnyebben ilyen formátumra konvertálni:

```
#!/bin/sh
# psfax - converts and sends Postscript files via fax
#
echo 'Converting PS to G3 fax ...'
gs -q -sPAPERSIZE=a4 -sDEVICE=dfaxhigh
-sOutputFile=/tmp/$2.fax - <$2

echo 'Sending fax to' $1
sendfax $1 /tmp/$2.fax
rm /tmp/$2.fax
```

Probléma esetén érdemes megtekinteni a /var/spool/fax/Faxlog fájl tartalmát, amely a sendfax parancs által elvégzett összes művelet protokollját tartalmazza.

## 6.6 SZTRÍMEREK ÉS CD-ROM

Ma már sok olyan PC konfiguráció van, amely sztrímet és/vagy CD-ROM-ot is tartalmaz. A Linux ezen adattáról eszközök használatát is támogatja. Ha a sztrímer vagy a CD-ROM vásárlásakor SCSI eszközt választ, akkor ezzel a Linuxba való telepítést is legyiszerűsíti. Az SCSI eszközök parancskészletenek szabványosítása következetében a konfigurálás viszonylag egyszerű. Ha az SCSI kezelőprogram már be van fordítva a kernelbe, akkor lehetővé válik a mknod parancson keresztül a megfelelő bejegyzések elhelyezése a /dev könyvtárba. A legtöbb Linux programcsomag ezeket az eszközkezelő fájlokat már a telepítés során automatikusan létrehozza. Például a CD-ROM-meghajtók (scd?) és az SCSI sztrímek (st?) konfigurálásához az alábbi bejegyzésekkel kell létezniük:

```
nicetry:/dev$ ls -l scd* rmt* st*
lrwxrwxrwx 1 root root 3 Apr 19 22:52 rmt0 -> st01
rwxrwxrwx 1 root root 3 Apr 19 22:52 rmt1 -> st1
crw-rw----
```

	szám	szolgáltató	lemez	szám	szolgáltató	lemez
1	root	root	12,	8	Jul 18	1994 rmt16
2	root	disk	12,	6	Jul 18	1994 rmt8
3	root	disk	11,	0	Jul 18	1994 scd0
4	root	disk	11,	1	Jul 18	1994 scd1
5	root	disk	9,	0	Jul 18	1994 st0
6	root	disk	9,	1	Jul 18	1994 st1

```
nicetry:/dev$
```

Ha a fentie bejegyzések nem léteznek, akkor a rendszeradminisztrátor a következő parancsokkal tudja ezeket létrehozni:

```
nicetry:/dev# mknod /dev/rmt0 c 9 0
nicetry:/dev# mknod /dev/scd0 b 11 0
```

Mivel az egyéb kezelőprogramok esetében nincsenek szabványok, egyéni AT busz-kontrollerrel rendelkező CD-ROM-meghajtó vagy sztrímer konfigurálása sokkal több problémát okozhat. Ez idő szerint uzonban a leggyakoribb modellek kezelőprogramjai már a Linux alatt is rendelkezésre állnak, azonban használataukhoz ezeket ugyancsak a kernelbe kell befordítani.

# ADMINISZTRÁTORI FELADATOK

**A** Linux rendszer telepítése és konfigurálása után a következő lépés a felhasználói hozzáférés megteremtése, illetve olyan alkalmazások iránti igények kielégítése, amelyeket nem tartalmaz a telepítési programcsomag. Az idő műlásával a rendszer összetevőiből is új változatok jelenhetnek meg, amelyeket telepíteni kell, ha a fejlődéssel lépést akarunk tartani. Ezeket a feladatokat együttesen rendszeradminisztrációnak nevezik. Mivel azonban a rendszeradminisztrátori teendők az összes UNIX rendszer esetében hasonlóak, és ezzel a témaval kapcsolatban bőséges irodalom áll rendelkezésre, itt csak azokat az ötleteket és észrevételeket tárgyaljuk, amelyek a Linuxra egyedileg jellemzők.

## 7.1 AZ ADMINISZTRÁTOR

A rendszer konfigurációs fájljait csak a rendszeradminisztrátor – `root` superuser – módosíthatja. E fájlokhoz tartozó hozzáférési jogok ugyanis megakadályozzák a jogosulatlan felhasználók hozzáférését. A rendszeradminisztrátor általában az összes fájlhoz hozzáférhet, és azokat akarata szerint módosíthatja.

Ez egyben azt is jelenti, hogy az adminisztrátor az egész rendszert tönkreteheti vagy törlheti. Például tegyük fel, hogy egy adminisztrátor gondatlanul beírja a következő parancsot amellyel a gyökérkönyvtártól kezdődően az összes fájlt kitörli (a megadott opciók következtében a parancs rekurzív módon az összes alkönyvtárat is kitörli és ehhez a felhasználótól még jóvahagyást sem kér):

```
nicetry:/# rm -rf *
```

A parancs tehát a teljes rendszert azonnal kitörli. Ha azonban ezt a parancsot `root` engedélyek nélkül hajtjuk végre, akkor a rendszerfájlokhoz és -könyvtárakhoz tartozó hozzáférési jogok garantálják, hogy csak az adott felhasználó fájljai törlődjenek. A parancsnak tehát nem lenne hatása a többi felhasználóra illetve magára a rendszerre sem.

Ebből következően tehát a rendszeradminisztrátoroknak különleges gondossággal kell eljárnia, és csak akkor szabad `root`-ként bejelentkeznie, ha a rendszerfájlokat szeretné módosítani, vagy új programot akar telepíteni.

## 7.2 A RENDSZER BETÖLTÉSE

Annak érdekében, hogy a rendszer működése minél érthetőbb legyen, ebben a részben a Linux rendszer betöltésének lépésein, illetve ezen művelet során feldolgozott programokat és szkripteket ismertetjük. Az operációs rendszertől függetlenül a rendszerek a betöltés során minden az MBR-t (*Master Boot Record*) töltik be elsőként. Ebben van ugyanis a partíciótábla és az a betöltőprogram, amely az aktív partíció boot szektorát tölti be.

Ha a Linux Loader (**LILO**) az aktív partíció elejére van telepítve, akkor az indul el elsőként és a különböző Linux kernellek és a DOS választékát kínálja. Ha a felhasználó a Linux kernelt

választja, akkor az ennek megfelelő kernel-image kerül betöltésre és elindításra. A kernel-image kiválasztás után paraméterek is átadhatók a kernel vagy az inicializáló eljárás számára.

A kernel először a videokártyát inicializálja, és kéri a felhasználótól a megfelelő **képernyőfelbonfás kiválasztását** (ha a kernel így konfiguráltuk, ellenkező esetben a kernelben meghatározott videomódot állítja be a kártyán). Ezután telepíti a különböző eszközkezelőket, amelyek mindegyik általában egy megggyezést is megjelenít. Végül logikailag beilleszti a gyökérfájlrendszerét és elindítja az init nevű processzt.

A Linux, a UNIX System V-hez hasonlóan több futtatási szinttel rendelkezik, amelyek jellemzői az /etc/inittab fájl tartalmazza. Ezek olyan különböző konfigurációk, amelyekben csak bizonyos rendszerozzetek aktiválódnak. A rendszer szokásos esetben többfelhasználósként indul el, ami azt jelenti, hogy a konzolra és opcionálisan a soros portakra is több getty processz indul el. Ráadásul ebben a módban az összes hálózati démon is aktiválódik.

Az egyfelhasználós üzemmód elsősorban rendszer-adminisztrációs célokat szolgál. Ez a mód akkor lép életbe, ha a betöltést a single opciónál indítottuk. Ezt az opciót sem a LILO sem a kernel nem értékeli ki, hanem a kernel elindulása után az init processz kapja meg. Egy további futtatási szint például az, amelyik a szokásos terminál bejelentkezés helyett grafikus bejelentkezési promptot (xdm) indít el.

Mivel az init az első processz, amelyet a kernel elindít, ehhez minden az 1-es szám tartozik, és ez az összes további processz szülőjének tekinthető. Az init processz az /etc/rc.d könyvtárban levő különböző szkripteket is elindítja, amelyek neve általában az rc karakterekkel kezdődik (az rc elnevezése a Run Control Script, Futásvezérlő Szkript név rövidítése – a lektor). Ezek a szkriptek újrainicializáláshoz bizonyos rendszerfájlok, és a lokális fájlrendszereket logikailag beilleszik. Az NFS fájlrendszerek logikai beillesztésére még nem kerül sor, mivel a hálózati és az ezzel kapcsolatos démonok még nem indultak el. A különböző szkriptek végrehajtásának pontos sorrendje rendszerrel rendszere változhat. A Slackware disztribúció esetén az /etc/rc.d könyvtár a következő fájlokat tartalmazza:

```
nicetry:~# cd /etc/rc.d
nicetry:/etc/rc.d# ls
rc.08      rc.6*       rc.M*       rc.cdrom*   rc.inet1*   rc.local*
rc.4*      rc.k*       rc.S*       rc.ibcs2*   rc.inet2*   rc.serial*
nicetry:/etc/rc.d#
```

- rc.S – A betöltés során az a szkript kerül elsőként végrehajtásra, amely a rendszert inicializálja.
  - rc.serial – Ez a szkript, amely opcionálisan az rc.S szkriptből is elindítható, a soros vezérlőkártyákat inicializálja.
  - rc.M – A rendszert többfelhasználóra állítja be, és a legfontosabb démonokat indítja el.
  - rc.font – A konzol számára opcionálisan egy másik fontkészletet aktivizál.
  - rc.inet1 – A TCP/IP rendszer alacsonyabb szintjeit inicializálja. Itt az IP cím, a hostnév és az átirányító táblázat beállítása meg vége.
  - rc.inet2 – A hálózati démonokat indítja el.
  - rc.0 – Rendszer leállításakor indul el.
  - rc.K – Többfelhasználóból az egyfelhasználós módba való átkapcsoláskor indul el.
- A következő példában az rc.S szkript részletei láthatók:

```

#!/bin/sh
#
# /etc/rc.d/rc.S: System initialization script.
#
# Mostly written by: Patrick J. Volkerding, <volkerdi@ftp.cdrom.com>
#
PATH=/sbin:/usr/sbin:/bin:/usr/bin

# enable swapping
/sbin/swapon -a
/sbin/kerneld

# Start update.
/sbin/update &

# Test to see if the root partition is read-only, like it ought to be.
READWRITE=no
if echo -n >> "Testing filesystem status"; then
  rm -f "Testing filesystem status"
  READWRITE=yes
fi

# Check the integrity of all filesystems
if [ ! $READWRITE = yes ]; then
  /sbin/fsck -A -a
  # If there was a failure, drop into single-user mode.
  if [ $? -gt 1 ] ; then
    echo
    echo
    echo ****
    echo "fsck returned error code - REBOOT NOW!"
    echo ****
    echo
    echo
    /bin/login
  fi
  # Remount the root filesystem in read-write mode
  echo "Remounting root device with read-write enabled."
  /sbin/mount -w -n -o remount /
  if [ $? -gt 0 ] ; then
    echo
    echo "Attempt to remount root device as read-write failed! This is going to"
    echo "cause serious problems... "
    echo
    echo "If you're using the UMSDOS filesystem, you **MUST** mount the root"
    echo "partition"
    echo "read-write! You can make sure the root filesystem is getting"
    mounted"

```

```

echo "read-write with the 'rw' flag to Loadlin:"
echo
echo "loadlin vmlinuz root=/dev/hd1 rw  (replace /dev/hd1 with your
root device)"
echo
echo "Normal bootdisks can be made to mount a system read-write with
the rdev command:"
echo
echo "rdev -R /dev/fd0 0"
echo
echo "You can also get into your system by using a bootkernel disk
with a command"
echo "like this on the LILO prompt line: (change the root partition
name as needed)"
echo
echo "LILO: mount root=/dev/hd1 rw"
echo
echo "Please press ENTER to continue, then reboot and use one of the above
methods to"
echo -n "get into your machine and start looking for the problem. "
read junk;
fi
else
echo "Testing filesystem status: read-write filesystem"
if [ ! -d /DOS ]; then # no warn for UMSDOS (kind of a bad test, but...)
cat << EOF
*** ERROR: Root partition has already been mounted read-write. Cannot check!
For filesystem checking to work properly, your system must initially mount
the root partition as read only. Please modify your kernel with 'rdev' so that
it does this. If you're booting with LILO, add a line:
read-only
to the linux section in your /etc/lilo.conf and type 'lilo' to reinstall it.
If you boot from a kernel on a floppy disk, put it in the drive and type:
rdev -R /dev/fd0 1
If you boot from a bootkernel disk, or with Loadlin, you can add the
This will fix the problem *AND* eliminate this annoying message. :^)
EOF
echo -n "Press ENTER to continue. "
read junk;
fi

```

```

fi

# remove /etc/mtab* so that mount will create it with a root entry
/bin/rm -f /etc/mtab* /etc/nologin /var/run/utmp

# mount file systems in fstab (and create an entry for '/')
# but not NFS because TCP/IP is not yet configured
/sbin/mount -avt nonfs

# Looks like we have to create this.
cat /dev/null > /var/run/utmp

# Configure the system clock.
# This can be changed if your system keeps GMT.
if [ -x /sbin/clock ]; then
    /sbin/clock -s
fi

# Setup the /etc/issue and /etc/motd to reflect the current kernel level:
# THESE WIPE ANY CHANGES YOU MAKE TO /ETC/ISSUE AND /ETC/MOTD WITH EACH
# BOOT. COMMENT THEM OUT IF YOU WANT TO MAKE CUSTOM VERSIONS.
tput clear > /etc/issue
echo >> /etc/issue
echo Welcome to Linux '/bin/uname -a | /bin/cut -d -f3'. on ''>> /etc/issue
echo >> /etc/issue
echo "'/bin/uname -a | /bin/cut -d -f1,3'." > /etc/motd

# Run serial port setup script:
# (CAREFUL! This can make some systems hang if the rc.serial script isn't
# set up correctly. If this happens, you may have to edit the file from a
# boot disk)
#
# . /etc/rc.d/rc.serial

```

A többfelhasználós módra vonatkozó rc.S szkript után az rc.M szkript végrehajtása következik, amely a legfontosabb démonokat indítja el. Hálózat esetén az rc.inet1 és rc.inet2 szkriptek meghívására is innen kerül sor.

### 7.3 RENDSZERLEÁLLÍTÁS

A többi UNIX rendszerhez hasonlóan ebben az esetben sem szabad a számítógépet egyszerűen kikapcsolni. (Amióta a merevlemez elérésének gyorsítására ún. winchester cache programokat – pl. a smartdrv.exe vagy a Norton Cache programok – használnak Dos/Windows környezetben, azóta ezeket a rendszereket sem szabad csak kikapcsolni – a lektor.) Helyette a shutdown parancs végrehajtásával a rendszert megfelelően le kell zárni. Az eljárás szükségességének az az oka, hogy a kernel cache memoriája miatt a programok által a merevlemezre írt adatok nem biztos hogy fizikailag is rátíródtak a merevlemezre. Továbbá az olyan gyakran igénybe vett információk, mint az i-node táblázat a RAM-ban tárolódnak. Tehát a számítógépet a shutdown

parancs végrehajtása nélkül kikapcsolva ellentmondások keletkezhetnek a merevlemezen, ami adatvesztéshez vezethet.

A shutdown parancs biztosítja, hogy az összes puffer tartalma fizikai tárolóeszközre frödjön, és az összes processz megfelelően fejeződjön be.

A sync parancs a pufferek tartalmát lemezre írja, azonban a számítógépet nem zárja le. Ezt a parancsot viszonylag ritkán használják.

## 7.4 A LINUX KÖNYVTÁRFÁJA

Annak érdekében, hogy a kezdő vagy kevés gyakorlattal rendelkező rendszeradminisztrátorok is könnyen eligazodjanak a rendszerben, ebben a részben a tipikus Linux telepítés legfontosabb könyvtárait ismertetjük. A Linux fájlrendszer szerkezetét a **Linux File System Standard** nevű szabványban rögzítették, amely a többi dokumentummal együtt a szokásos FTP szervereken egy PostScript fájlból megtalálható. Ezt a szabványt a legtöbb disztribúció és programcsomag előállítója betartja. A következőkben egy átlagos disztribúciót tételezünk fel.

A / könyvtár a Unixban és a Linuxban a könyvtárfája gyökere, ezért gyökérkönyvtárnak nevezik. Ez a könyvtár a rendszer betöltséhez szükséges image-fájlok és a legfontosabb alkönyvtárokban kívül más nem tartalmazhat. A rendszeradminisztrátorok a gyökérkönyvtárat gyakran bejelentkezési könyvtárként is használják, azonban erre a célra inkább egy másik könyvtár pl. /root létrehozását javasoljuk, mivel így jobban elkülnöthetők az adminisztrátor konfigurációs fájllai a rendszcráfjuktól. Új bejelentkezési könyvtár létrehozásához egy szövegszerkesztő segítségével kell az /etc/passw fájlból a megfelelő bejegyzést módosítani.

### A gyökér könyvtárai

- **/bin** – Ebben a könyvtárban vannak azok a legfontosabb programok, amelyeknek még akkor is hozzáérhetőknek kell lenniük, ha a /usr könyvtár nem érhető el. Itt olyan programok vannak, mint az mv, cp, cat és rm. Ellentétben az /sbin könyvtárral, amely csak a rendszer betöltésével és az adminisztrációs teendőkkel kapcsolatos programokat tartalmazza, a /bin könyvtár programjai az összes felhasználó számára szólnak. Az összes olyan program, amely vész helyzet esetén nem alapvetően fontos, az /usr/bin könyvtárban (lásd még a /usr könyvtárat) található meg.
- **/boot** – Ez a könyvtár tartalmazza a Linux Loader leképező (map) fájloidjait és a régi boot szektor, valamint particiótábla biztonsági másolatát. Ezeket a fájlokat általában csak a LILO használja, illetve a LILO horzza automatikusan létre.
- **/conf** – Ha ez a könyvtár létezik, akkor kizártolag konfigurációs fájlokat tartalmaz, amelyek egyébként az /etc vagy egyéb könyvtárakban vannak. Ebben az esetben az /etc könyvtár valódi fájlok helyett csak szimbolikus csatolásokat tartalmaz a /conf vagy annak alkönyvtáraiban (pl. /conf/net) tárolt fájlokhoz. Az egyszerűbb telepítések ezt az alkönyvtárat nem hozzák létre.
- **/dev** – Ahogy a neve is sugallja, ez a könyvtár tartalmazza az összes I/O kezelőhöz hozzárendelt speciális fájlokat, amelyek az eszközmeghajtó programok elérési pontjai (lásd még a 2.6. részben.)
- **/dist** – Ez az a hely, ahol a Unifox és Linux Universe disztribúciók logikailag beillesztik a CD-t. A merevlemezre nem telepített programcsomagokhoz és programokhoz a merevlemezben csak csatolások tartoznak, amelyek a /dist könyvtár alkönyvtáraiban levő fájlokhoz mutatnak.
- **/etc** – Az /etc könyvtár konfigurációs fájlokat tartalmaz. Ilyenek például a passwd és a group, amelyek egyéni, illetve csoport információkat tartalmaznak, vagy a TCP/IP démonok konfigurációs fájliai, mint például a services, inetc.conf vagy az exports. A fájlerendszerre vonatkozó szabványok megalkotása előtt ez a könyvtár gyakran tartalmazott olyan

démonokat és rendszerprogramokat, mint az `init` és `update`. Ezek azonban most már az `/sbin` vagy az `/usr/sbin` könyvtárakban vannak.

- **/etc/Isode** – Ez a könyvtár az Isode programcsomag (lásd a 9.16. részt) konfigurációs fájljait tartalmazza.
- **/etc/X11** – A fájlrendszer szabványának megfelelően ez a könyvtár az X11 lokális konfigurációs fájljait tartalmazza. Ezek közé tartozik az `XF86Config`, a szerverre és a monitorra vonatkozó általános beállításokkal, az `Xmodmap` az X11 alatti billentyűzet-kiosztással, az `xinitrc` és az `xdm-re` vonatkozó fájlok.
- **/etc/init.d** – Bizonyos disztribúciókban ez a könyvtár tartalmazza az aktuális `rc` szkripteket amelyeket a rendszer a betöltéskor és a lezáráskor használ. Enzen szkriptek meghívása az `/etc/rc0.d`-/`/etc/rc6.d` könyvtárakban levő szimbolikus csatolásokon keresztül törtenik.
- **/etc/keytables** – A fájlrendszer szabványának megfelelően ez a könyvtár a billentyűzetkiosztás leképezésekét biztosítja, amelyek a rendszer betöltése során töltődnek be. Az amerikai disztribúciók néha erre a célra a `/usr/lib/keytables` vagy az `/usr/liv/kbd/keytables` alkönyvtárat használják.
- **/etc/ppp** – Ebben a könyvtárban a PPP konfigurációs fájlok vannak.
- **/etc/rc0.d**–**/etc/rc6.d** – Ezeket a könyvtárat azok a disztribúciók használják, amelyeknél az indító szkriptek az `/etc/init.d` könyvtárban vannak. Ezen könyvtárak szkriptjeit az `init` hajtja végre a futtatási szint változtatásakor. Az `/etc/rc?.d` könyvtárak csak csatolásokat tartalmaznak az `/etc/init.d` könyvtárakban levő fájlokhoz.
- **/etc/rc.d** – A rendszer indítása során az `init` által meghívott szkriptek az `/etc/init.d` könyvtár helyett ebben vagy az `/etc` könyvtárban is lehetnek.
- **/etc/skel** – Az ebben a könyvtárban tárolt fájlok automatikusan bemásolódnak a új felhasználó saját könyvtárba, ha az új felhasználót a `useradd -m` parancsral létesítik. A könyvtár általában az olyan konfigurációs fájlokra, mint a `.cshrc`, `.bashrc`, `.Xdefaults` és `.emacs` tartalmaz példákat.
- **/FTP** – Bizonyos disztribúciók esetén ezt a könyvtárat az FTP szerver démon használja. A szerverhez `ftp`-ként vagy `anonymous`-ként bejelentkező felhasználók csak az `/FTP` alkönyvtárhoz férhetnek hozzá. Más disztribúciók ugyanerre a célra a `/home/FTP` könyvtárat használják.
- **/home** – Ez a könyvtár tartalmazza az összes felhasználó saját (`home`) könyvtárát a `root` könyvtár kivételével. A saját könyvtárok felhasználóspecifikus konfigurációs fájlokat tartalmaznak. Programok ezekből nem telepíthetők. Mivel ez a könyvtár általában önálló particióban van, a `root` könyvtárat nem tanácsos itt elhelyezni. Ha ugyanis ezt a fájlrendszt valamilyen hiba következtében nem lehet logikailag a rendszerbe illeszteni, akkor még a rendszeradminisztrátor sem tud bejelentkezni a rendszerbe, és így a hibát sem tudja kijavítani.
- **/install** – Bizonyos terjesztések telepítőprogramja ezt az alkönyvtárat a telepítő csomagokon levő információ tárolására használja. Egyéb disztribúciók a `/var` vagy a `/usr` speciális alkönyvtárat használják.
- **/lib** – A rendszer legfontosabb megosztott tárgykódkönyvtárainak fájljai vannak ebben a könyvtárban. Ezek a fájlok részei a megosztott könyvtárnak, amelyek az aktuális rutinokat tartalmazzák. A betöltésüket azok a programok hajtják végre, amelyek ezeket felhasználják. Az egyéb részek, amelyeket `stubs`-nak neveznek, a `/usr/lib` alkönyvtárban találhatók. Ezek csatolva vannak a programokhoz és csak az aktuális rutinokhoz tartalmazzák a csatolásokat. A megosztott könyvtákat, amelyek a betöltéskor és az adminisztrációhoz nem feltétlenül szükségesek, például az X Window rendszer könyvtárait, az `/usr` alatti másik könyvtárban kell tárolni. Az X11 esetén ez a könyvtár a `/usr/X11R6/lib/lib`.

- **/local** – Bizonyos CD-disztribúciók esetén a `/usr/local` könyvtárból egy szimbolikus csatolás elhelyezkedik a könyvtárhoz mutat. Azokat a lokális programokat, amelyeket a CD nem tartalmaz, ide kell telepíteni.
- **/lost+found** – Ez a könyvtár az `ext2` típusú fájlrendszer létrehozásával automatikusan keletkezik, és olyan segédprogramok használják, mint az `fsck`.
- **/mnt** – Ez a könyvtár, amelynek törlesztéshez használatos.
- **/proc** – A proc fájlrendszer általában ide illesztik logikailag a rendszerbe. A proc egy speciális fájlrendszer, amely a kernelben tárolt információt és a processeket minden alkönyvtárákat és fájlokat futtatja. Ezek a fájlok szövegként olvashatók (lásd még a 3.2-es részt). (A kernel belső táblázataiból készített, látszólagos könyvtárrendszert szimulál ez az alkönyvtár; így könnyen lehet információhoz jutni a rendszer legelsőbb részeiről – a lektor.)
- **/root** – Bár ez a könyvtár opcionális, a legtöbb Linux disztribúció létrehozza. Ez a superuser bejelentkezési könyvtára, s általában nincs közös particióban az egyéb felhasználók saját könyvtáraival.
- **/sbin** – Ez a könyvtár csak azokat a legfontosabb programokat és parancsokat tartalmazza, amelyek a rendszer betöltéséhez és az alapvető rendszерadminisztrátori feladatok megvalósításához szükségesek. Ezek közé olyan parancsok tartoznak, mint a `getty`, `init`, `update`, `fdisk`, `fsck`, `ifconfig`, `ping` és `lilo`. Az egyéb olyan programok, amelyek más felhasználók által szükségesek, azonban nem a root könyvtárban vannak, hanem a `/bin` vagy a `/usr/bin` könyvtárban, ha vész helyzet esetén nem feltétlenül szükségesek.
- **/shlib** – Az iBCS2 emulátor által futtattott programok osztott tárgykódkönyvtárai vannak itt.
- **/tftpboot** – Bizonyos disztribúciók ezt a könyvtárat a `tftp` démonhoz használják és amennyiben ez a helyzet, akkor a `tftp`-vel történő hozzáférés erre a könyvtárra korlátozható.
- **/tmp** – Ezt a könyvtárat sok program az ideiglenes fájlok tárolására használja. Ehhez a könyvtárhoz az összes felhasználónak írás/olvasási jogja van. A `/tmp` könyvtárban levő fájlokat rendszerint rendszerindításakor automatikusan törlik, az `/etc/rc.d/rc.M` fájlból levő meglévő bejegyzéssel.
- **/user** – Ez a könyvtár, amíg letezik, általában üres, és logikai **beillesztéshez** használatos.
- **/usr** – Ez a könyvtár tartalmazza az olyan fontos alkönyvtárok legtöbbjét, amelyek a rendszer betöltéséhez nem közvetlenül szükségesek. Elválasztva a gépfüggő konfigurációt, a rendszeradminisztrációhoz alapvetően szükséges programokat és a bejelentkezési vagy spool fájlokat a `/usr` könyvtárban levő programoktól, lehetővé válik a `/usr` használata CD-ről vagy több gép esetén az NFS szerverről. Ebben az esetben azonban a `/usr` könyvtárnak írásvédelemtől kell a rendszerbe logikailag kapcsolódnia. A rendszeradminisztrációra vonatkozó legfontosabb programoknak és könyvtárnak a gyökér-fájlrendszeren kell lenniük, azért, hogy ha a CD vagy az NFS szerver hozzáférhetetlenné válik, a hibát ki lehessen javítani. A gyökér-fájlrendszernek olyan kis méretűnek kell lennie, amennyire ez csak lehetséges, hogy lehetővé tegye minél több program megosztott használatát és minél több helyet biztosítson.
- **/var** – Ez a könyvtár tartalmazza az összes olyan rendszerüzemeléssel kapcsolatos fájlt, amelyeket folyamatosan kell fríni és amelyek mérete gyakran változik (*szinte minden nap*). Ezek között tartoznak a napló és a spool fájlok. Sok olyan alkönyvtár, amely a `/var` alatt helyezkedik el, korábban a `/usr` alatt volt. Annak érdekében, hogy a `/usr` könyvtárat egyidejű és csak olvastatási sorral kapcsolatos alkönyvtárrakkal, a `/var/spool` a levelezéssel és a nyomtatási lock a `/var/lock` fájlokkal.

## A /usr alatti alkönyvtárak

- **/usr/X386** – A régi verziójú X11 programcsomagok könyvtárfájának ez a kezdete, amely helyett a Release 6 esetén az **/usr/X11R6** használatos.
- **/usr/X11R6** – A Release 6-tól kezdődően ez az X Window rendszer könyvtára. A **/usr/lib/X11** és a **/usr/bin/X11** ehhez a könyvtárfához csatolások.
- **/usr/adainclude** – A GNU Ada fordítóprogram include fájljai vannak itt.
- **/usr/adm** – Ez a könyvtár egy csatolás a **/var/adm**-hez.
- **/usr/bin** – Ebben a könyvtárban vannak a felhasználók számára szóló **programok** és UNIX parancsok, valamint azok a rendszeradminisztrátor számára szóló programok, amelyekre nincs feltétlenül szükség akkor, ha a **/usr** könyvtárat nem lehet logikailag beilleszteni. A UNIX parancsoknak a szükséges a **/bin** és **/sbin**, valamint a **/usr/bin** könyvtárak között nem minden egységes. Ha kétséges, hogy az adott parancs melyik alkönyvtárban van, akkor érdemes mindenből megnézni. A PATH környezeti változóban mind a **/bin**, mind a **/usr/bin** könyvtárnak szerepelnie kell.
- **/usr/bin/X11** – Az X11 programokat ebbe az alkönyvtárban szokás telepíteni. Ez azonban rendszerint csak egy szimbolikus csatolás a **/usr/X386/bin** vagy a **/usr/X11R6/bin** alkönyvtárhoz az újabb X11 kibocsátásoknál. A PATH környezeti változónak ezt a könyvtárat tartalmaznia kell.
- **/usr/dict** – Ez a könyvtár eredetileg a **look** parancshoz tartozó angol könyvtárat tartalmazta és olyan parancsokat, amelyek a helyesírás ellenőrzéshez szükségesek.
- **/usr/doc** – Olyan dokumentációt tartalmaz, amely a kézikönyvoldalakon nem hozzáférhető.
- **/usr/etc** – Ennek a könyvtárnak olyan konfigurációs fájlokot kell tartalmaznia, amelyeket több számítógép oszt meg egymás között. Gyakran ez könyvtár csak egy szimbolikus csatolás az **/etc** könyvtárhoz.
- **/usr/g++-include** – Ez a könyvtár fejlibc (header) fájlokat tartalmaz a GNU C++ fordító számára.
- **/usr/games** – Ebben a könyvtárban vannak a játékok és olyan szórakoztató programok, amelyek komoly használat esetén csak másodlagos szerepet töltenek be.
- **/usr/include** – Ez a könyvtára a C könyvtár **inculde fájlljainak**. Rendszerint a **sys** és a **linux** alkönyvtákat tartalmazza, ahol a **linux** egy csatolás a **/usr/src/linux** alkönyvtárhoz.
- **/usr/info** – Ez a könyvtár a GNU Info rendszerhez használatos. Az ebben a könyvtárban lévő fájlok az Emacs szerkesztő info módjában tekinthetők meg, vagy olyan programokkal mint a **tinfo**. A GNU programok elsődleges dokumentációjáit biztosítják. (*Gyakorlatilag a DOS alatti Norton Guide-nak felelnek meg, mind tartalmukban [dokumentációt tartalmaz], mind kialakításban [menüs, hypertext] – a lektor.*)
- **/usr/lib** – A különböző programozási nyelvek statikus könyvtárait és a megosztott könyvtákat, valamint egyéb programok súgó és konfigurációs fájlljainak alkönyvtárait tartalmazza.
- **/usr/lib/X11** – Itt találhatók a konfigurációs adatok, a karaktertáblák, színtáblák és az X Window rendszer egyéb fájlljai. Ez a könyvtár általában a **/usr/X11R6/lib/X11** könyvtárhoz csatolás. Az X szerver lokális konfigurációjával kapcsolatban álló fájlok, mint például az **XF86Config**, a fájrendszer szabványainak megfelelően az **/etc/X11** alatt vannak ténylegesen tárolva. Bizonyos disztribúciók azonban eltérnek eztől.
- **/usr/local** – Ebbe a könyvtárba kell telepíteni az összes **további** olyan programot, amelyek nincsenek a telepítő csomagban. Ez a könyvtár általában teljes könyvtárfát tartalmaz a **bin**, **lib**, **etc**, **include** és a **man** könyvtárkból. A **/usr/local/bin** útvonalat a PATH környezeti változónak, a **/usr/local/man** útvonalat pedig a kézikönyv oldalakra vonatkozó **MANPATH** változónak kell tartalmaznia.
- **/usr/man** – A kézikönyvoldalak a **/usr/man** alkönyvtáraiban vannak.
- **/usr/openwin** – A **/usr/openwin** alkönyvtáraiban olyan programok és adatok találhatók, amelyek a Sun XView csomagjából származnak. A könyvtárak és a konfigurációs fájlok általá-

- ban a `/usr/openwin/lib` könyvtárban vannak. E fájlok közül a legérdekesebbek az OpenLook ablakkezelő (`olwm` és `olvwm`) menüpén definíciós fájljai. A fájlnévek minden az `openwin-menu` karakterrel kezdődnek. Maga az ablakkezelő és más programok a `/usr/openwin/bin` alkonyvtárban vannak.
- `/usr/pkg` – Vannak olyan terjesztések, amelyek ezzel a könyvtárral az egyes programcsomagok önműködő telepítését teszik lehetővé. Innen a programok a telepítőprogram felhasználásával merevlemezre installálhatók.
  - `/usr/sbin` – Az `/sbin` könyvtárhoz hasonlóan itt elsődlegesen olyan fájlok találhatók, amelyek a rendszeralapítási szolgáltatóhoz és nem a rendszer behüzéséhez szükségesek. Ráadásul ez a könyvtár hálózati démonokat is tartalmaz.
  - `/usr/share` – Ez a könyvtár tartalmazza a gépfüggetlen fájlokat. A könyvtár tartalma logikailag beilleszthető számos gépen NFS-sel. Ilyen fájlok tartalmazzák például a kézikönyv oldalakat és a `terminfo` adatbázist.
  - `/usr/src` – A `/usr/src` alatti alkonyvtárak tartalmazzák a rendszerprogramok forráskódjait. Ezek közül a legfontosabb a `/usr/src/linux`, amelyben a Linux kernel forráskódja van.
  - `/usr/tex` – Ebből a könyvtárból a TeX programcsomagot telepítették. A fájlrendszer szabványainak megfelelően a TeX adattárfájloknak a `/usr/lib/Tex` alkonyvtárban kell lenniük.

## 7.5 FELHASZNÁLÓK ÉS CSOPORTOK

Minden felhasználói azonosítóval (user ID) rendelkezik, és egy vagy több csoportba tartozik. Ezt az információt az `/etc/passwd` és az `/etc/group` fájlok tárolják. Ezeket a fájlokat általában nem lehet egy egyszerű szövegszerkesztővel módosítani, helyette a `useradd` programot kell használni.

A program számára az összes fontos információ a parancssorban keresztül adható meg. A program ezután módosítja az `/etc/passwd` és az `/etc/group` fájlokat valamint a `shadow` fájlt is, amely a titkosított felhasználói és csoportjelszavakat tartalmazza, és csak a rendszer adminisztrátora számára olvasható.

Az adminisztrátor a felhasználók és csoportok kezelését egyszerűsítheti, ha a `useradd` parancsot a `-D` opcióval végrehajtja a csoportokhoz alapértelmezett értéket, hosszabb ideig érvényes jelszavakat, és felhasználók számára saját könyvtárat rendel.

A felhasználó-specifikus fájlokat, pl. `.profile`, `.bashrc` és az `openwin-menu` az `/etc/skel` könyvtárban érdemes tárolni, mivel így új felhasználó létrehozása esetén ezeket a rendszer automatikusan bemosolja a felhasználó bejelentkezési könyvtárába.

Az alapértelmezett értékek definíciója és a fájlok `/etc/skel` könyvtárba másolása után az új felhasználót a következő parancssal lehet létesíteni:

```
nicetry:~# useradd -m username
```

A felhasználói azonosító automatikusan a következő szabad szám lesz. Ezután a jelszó hozzárendelését kell elvégezni a `passwd <user name>` parancssal, mivel az új azonosító ellenkező esetben leírásra marad.

A beállítások `usermod` parancssal módosíthatók, a felhasználó pedig a `userdel` parancssal törölhető. A `chsh` a bejelentkezési shellt, a `chfn` pedig a felhasználó teljes nevét állítja be. A kézikönyvoldalak a `useradd` és az egyéb parancsokról további információkat tartalmaznak.

A következő példa az alapértelmezett értékek beállítását és az új felhasználó hozzáadását szemlélteti:

```

nicetry:~# useradd -D -g 6 -b /home -f 3 -e 999
nicetry:~# useradd -m peter
nicetry:~# passwd peter
Changing password for peter
Enter the new password (minimum of 5 characters)
Please use a combination of upperand lower case letters and
numbers.
New password:
Re-enter new password:
nicetry:~#

```

A csoportokat a groupadd, groupmod és a groupdel parancsokkal lehet kezelni. A usermod parancsot a -G opcióval végrehajtva a felhasználó egy másik csoporthoz rendelhető.

### Anónimusz FTP

A root-on kívül az ftp felhasználói névnek is különleges szerepe van. Ha ez a felhasználó létezik, akkor bármelyik felhasználó bejelentkezhet a rendszerbe az ftp parancssal anélkül, hogy azonosítóra lenne szüksége. A felhasználónak egyszerűen ftp-ként vagy anonymous-ként kell bejelentkeznie, majd a rendszer jelszóként az e-mail címét kér be. Az ftp felhasználó saját könyvtára ezután gyökérkönyvtárként lesz használatos, így az a felhasználó, aki ilyen módon szerzett hozzáférést, csak bizonyos fájlokat érhet el.

Ez azonban feltételezi a dev, bin és usr könyvtárak jelenlétéit az ftp felhasználó saját könyvtárában a megfelelő fájlokkal. Az elrendezés pontos magyarázata az ftpd-re vonatkozó kézikönyvoldalakon található meg.

## 7.6 SHELEK

Annak érdekében, hogy a felhasználók maguk is tudjanak shellt váltani, az /etc/shells fájlnak az útvonalával együtt tartalmaznia kell az összes rendelkezésre álló shellt.

Ha az /etc/shells fájlban nincs bejegyzés a shell számára, akkor azt nem lehet használni a felhasználóknak bejelentkezési shellként és a különböző TCP/IP programokkal is problémák lehetnek. A következő részlet egy ilyen /etc/shells fájlrészletet szemléltet:

```

/bin/sh
/bin/bash
/bin/tcsh
/bin/csh

```

A chsh parancs lehetővé teszi a felhasználóknak, hogy a bejelentkezési shelljüket megváltoztassák. Az adminisztrátorok ugyancsak ezt a parancsot kell használnia az azonosító konfigurálásakor. Ebben az esetben a vonatkozó felhasználói nevet paraméterként kell átadni.

```
nicetry:/dev$ chsh
Changing shell for demo.
New shell [/bin/bash]: /bin/tcsh
Shell changed.
nicetry:/dev$
```

## 7.7 A FELHASZNÁLÓK TÁJÉKOZTATÁSA

A bejelentkezési prompt előtt megjelenő információt az `/etc/issue` fájl tartalmazza. Ebben a fájlból általában üdvözlő üzenetek, a számítógép neve és a felhasználó számára szóló utasítások bejegyzései vannak.

Mutat a felhasználó bejelentkezett, általában az `/etc/motd` fájlból származó ún. napi üzenet jelenik meg a számára. Bizonyos rendszerek esetén, amelyek a megfelelő login programárnyék jelezőket használják, ez az `/etc/login.defs` motd fájlból kerül konfigurálásra.

Vannak olyan disztribúciók, beleértve a Slackware-t is, amelyek felülrőjük az `/etc/issue` és az `/etc/motd` fájlokat a rendszer indító szkriptjében. Annak érdekében, hogy a felhasználó kihasználhassa ezeket a fájlokat, azokat a parancsokat, amelyek ezeket felülírják, a szkriptekből el kell távolítani. A Slackware disztribúció esetén ez az `/etc/rc.d/rc`.S fájlból történik.

## 7.8 BIZTONSÁGI MÁSOLATOK

A `tar` parancs a fontos fájlok biztonsági másolatának elkészítéséhez viszonylag egyszerű megoldást kínál. A `tar` egyike azon szabványos UNIX parancsoknak, amelyek a Linux alatt is rendelkezésre állnak.

Például a `/home/stefan` alkönyvtárban levő összes fájl lemezre mentéséhez a `tar` parancsot a M (többkötes mód) opcionál kell végrehajtani. Amikor egy lemez betelt, akkor a `tar` a következő lemez behelyezésére szólít fel. A következő példa a `tar` parancs ilyen célú felhasználását mutatja be:

```
nicetry:~# cd /home/demo
nicetry:/home/demo# tar cvfM ./dev/fd0 *
```

Az archívum során az alkönyvtárak automatikusan létrejönnek. Ha a biztonsági másolatot a mérlemezre kell visszaállítani, akkor ugyancsak a `tar` parancsot kell használni az M opcionál (ellenkező esetben a műveletet már az első lemez után befejeződik). A következő `tar` parancs a mérlemezre mentett fájlokat állítja vissza a mérlemezre:

```
nicetry:~# cd /home/demo
nicetry:/home/demo# tar xvfm ./dev/fd0 *
```

Nagyobb méretű biztonsági másolatokat is ugyan ilyen módon lehet készíteni sztrímerre. Ebben az esetben azonban a `/dev/fd0` helyett a sztrímer cszközkezelőjét (`/dev/rmt0`) kell előírni.

Az M opcionál nem áll rendelkezésre olyan UNIX rendszerekben, amelyek nem használják a GNU tar parancsát. Ennek következtében az olyan biztonsági másolatoknál, amelyeket várhatóan más UNIX gépekre töltének majd át, ezt az opcionál nem szabad alkalmazni.

A `tar` parancs, a többi MTools-hoz hasonlóan feltételezi a lemezek alacsony szintű formázását. Az alacsony szintű formázást az `fdformat` parancssal lehet készíteni, amelyet a könyv referencia része tárgyal.

## 7.9 FÁJLRENDSZEREK KEZELÉSE

A rendszeradminisztrátor egy másik feladata a fájlrendszer kezelése. Szokásos esetben ez a feladat csak a szabad lemezterület méretének rendszeres ellenőrzésére, a napló fájlok figyelésére, és – időről időre – a `/tmp` könyvtárak törlésére korlátozódik.

A rendszer összeomlása esetén azonban a rendszeradminisztrátornak a fájlrendszer egységességenként ellenőrzését is végre kell hajtania. A legtöbb disztribúció az egységesség ellenőrzését minden rendszerbetöltés esetén végrehajtja. A kernel betöltése után a fájlrendszer logikai beillesztése először csak olvashatóként történik meg, majd az ellenőrzés következik. Ezután a beillesztés még egyszer végbernegy az `rc` szkriptben írható olvashatóként. Problémák esetén azonban az egységesség ellenőrzését manuálisan kell elvégezni.

Emiatt az önálló Linux fájlrendszerek egy különleges eszközt is biztosítanak, amelynek neve `fsck`. Az adminisztrátornak biztosítania kell, hogy az ellenőrzendő fájlrendszer logikailag ne legyen a rendszerbe illesztve, és hogy a megfelelő ellenőrzőprogram legyen használva. Jelenleg a legnépszerűbb az `ext2` fájlrendszer, amelynek megfelelő eszköze az `e2fsck`. A következő példa az `ext2` fájlrendszer egységességének ellenőrzését mutatja:

```
nicetry:~# mount
/dev/hda2 on / type ext2 (rw)
/dev/hda4 on /usr/local type ext2 (rw)
none on /proc type proc (rw)
/dev/sdb3 on /hawk type ext2 (rw)
/dev/hdd1 on /mnt type ext2 (rw)
nicetry:~# umount /mnt
nicetry:~# e2fsck -f -y /dev/hdd1
e2fsck 1.40, 16-May-96 for EXT2 FS 0.5b, 95/08/09
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/hdd1: 4123/313344 files (3.8% non-contiguous), 973165/1251904 blocks
nicetry:~# mount -t ext2 /dev/hdd1 /mnt
nicetry:~#
```

Az `fsck` meghívásakor a fájlrendszer vagy a partició paraméterként szerepel. A program szokásos esetben az eltéréseket a konzolon jeleníti meg. Ha viszont hibaüzenet nem jelenik meg, akkor a fájlrendszer nem sérült.

Különleges opciók automatikus hibajavítást is lehetővé tesznek. Az `fsck` program fejlesztői azonban inkább az interaktív javítási módot javasolják.

## 7.10 FRISÍTÉSEK

Mivel – különösen a Linux esetében – a kernel, a C könyvtár vagy a C fordító új verziói szinte minden esetben meg, a rendszерadminisztrátor egyik feladata a frissítések telepítése a rendszernél. Ez a rész a frissítések véghajtásának módjáról ismerteti. (Amennyiben a rendszer stabil verziókkal jól működik, akkor a frissítéseket nem érdemes elvégezni. Frissíteni csak akkor érdemes, ha egy új feladat elvégzéséhez új alkalmazás telepítése szükséges és ez az új alkalmazás javított vagy bővíti szolgáltatásokat vár el a Linuxról. Számítani kell rá, hogy ilyenkor más alkalmazások frissítésére is szükség lehet. Amennyiben ezekből nem áll rendelkezésre újabb verzió, akkor problémák lehetnek a rendszer üzemelésében – a lektor.)

### GCC

A GNU C fordítójának legújabb verzióját a Linuxhoz a prep.ai.mit.edu FTP szerverről lehet letölteni. Ez általában több fájlból áll, amelyiket a tar és a gzip programokkal tömörítetik, és amelyek lefordított programokat tartalmaznak a Linuxhoz.

A régi C fordító lecsatlásáhez a tar fájlokat általában a gyökérkönyvtárban kell kicsomagolni. A folyamat a régebbi verziót felülírja. A tárolási hely megtakarításának érdekében a régi verzió /usr/lib/gcc-lib alatti könyvtárait törölni lehet.

A frissítéssel kapcsolatban részletesebb információt ugyanazon tar archívumhoz tartozó README vagy RELEASE fájlokban lehet olvasni. (A gcc és a hozzá tartozó tárgykód könyvtárak frissítésével csak sok gyakorlat megszerzése után kísérletezzünk – a lektor.)

### Tárgykódkönyvtárak

A Linux C könyvtár a C fordítóhoz hasonlóan a tsx-11.mit.edu FTP szerverről tölthető le a /pub/linux/packages/GCC alkonyvtárból. A könyvtár új verziójának telepítéséhez általában két tar archív szükséges, amelynek nevében az image és az inc karakterek vannak. A fájlok közül az egyik az aktuális könyvtári fájlokat tartalmazza, míg a másik a megfelelő fejlécfájlokat. Ráadásul van még egy olyan fájl is, amelynek nevében az extra karakterek szerepelnek, és amely hibakereséshez és profilozáshoz tartalmaz könyvtárat.

Ezeket az archívokat a gyökérkönyvtárban kell kicsomagolni, míg az include fájlokat az /usr/include, és a könyvtári fájlokat pedig a /lib és a /usr/lib alkonyvtárakban.

```
nicetry:/# tar xvzf libc-5.3.12-bin.tar.gz
drwxr-xr-x root/root      0 Apr 29 23:54 1996 lib/
-rwxr-xr-x bin/bin 583795 Apr 25 07:15 1996 lib/libc.so.5.3.12
-rwxr-xr-x bin/bin 36156 Apr 25 07:15 1996 lib/libm.so.5.0.6
-rwxr-xr-x root/root 25287 Jul 23 23:22 1995 lib/libgdbm.so.1.7.3
-rwxr-xr-x root/root 52641 May 19 05:47 1995 lib/libcurses.so.1.0.0
drwxr-xr-x root/root      0 Apr 29 23:52 1996 lib/libtermcap.so.2.0.8
drwxr-xr-x root/root      0 Apr 29 23:30 1996 usr/
-rw-r--r-- bin/bin    1052 Apr 25 07:15 1996 usr/lib/crti.o
-rw-r--r-- bin/bin    1192 Apr 25 07:15 1996 usr/lib/crtbegin.o
-rw-r--r-- bin/bin   1264 Apr 25 07:15 1996 usr/lib/crtbeginS.o
-rw-r--r-- bin/bin 12462 Dec 22 22:48 1995 usr/include/arpa/nameser.h
-rw-r--r-- bin/bin 10498 Feb 18 06:34 1995 usr/include/arpa/telnet.h
```

-rw-r--r-- bin/bin	2982 Feb 18 06:34 1995	usr/include/arpa/tftp.h
drwxr-xr-x bin/bin	0 Apr 29 23:26 1996	usr/include/bsd/
drwxr-xr-x bin/bin	0 Apr 29 23:26 1996	usr/include/bsd/sys/
-rw-r--r-- bin/bin	90 Feb 18 06:34 1995	usr/include/bsd/sys/ttychars.h
-rw-r--r-- bin/bin	998 Jun 11 02:25 1995	usr/include/bsd/bsd.h

A könyvtárak verziószáma három részből áll, egy fő- és egy alverziószámból valamint a javítási szint, patch level jelzésszámából. A fájlok, amelyek a fenti /lib könyvtárba kerülnek kicsomagolva a 4-es fő és az 5-ös alverziószámmal, 26-os javítási szinttel, például /lib/libc.so.4.5.26, /lib/libc-lite.so4.5.26 és /lib/libm.so.4.5.26-nak nevezhetők. Ezek a fájlok olyan szimbolikus csatolásokon keresztül értehők el, amelyek csak a fő verziószámot tartalmazzák a nevükben. Az előbbi fájlok esetében a /lib/libc.so.4 és a /lib/libm.so.4 helyekhez szimbolikus csatolásoknak kell lenniük, amelyek aktuális fájlokhoz hivatkoznak.

Új verziószámmal rendelkező könyvtár telepítéséhez ezeket a csatolásokat módosítani kell. Az alábbi példa egy új könyvtár telepítését mutatja be az ldconfig program felhasználásával:

```
nicetry:~# ldconfig -nv /lib
ldconfig: version 1.8.0
/lib:
    libncurses.so.3.0 => libncurses.so.3.0
    libe2p.so.2 => libe2p.so.2.1
    libncp.so.1 => libncp.so.1.0
    libproc.so => libproc.so
    libproc.so.0.99 => libproc.so.0.99
    libm.so.4 => libm.so.4.6.27
    libcurses.so.0 => libcurses.so.0.1.2
    libc.so.4 => libc.so.4.7.6
    libtermcap.so.2 => libtermcap.so.2.0.0
    libcurses.so.1 => libcurses.so.1.0.0
    libm.so.5 => libm.so.5.0.6
    libc.so.5 => libc.so.5.3.12
    libdl.so.1 => libdl.so.1.7.3
    libss.so.2 => libss.so.2.0
    libext2fs.so.2 => libext2fs.so.2.0
    libcom_err.so.2 => libcom_err.so.2.0
    libe2p.so.1 => libe2p.so.1.0
    libe2fs.so.1 => libe2fs.so.1.0
    libet.so.1 => libet.so.1.0
    libss.so.1 => libss.so.1.0
nicetry:~# cd /lib
nicetry:/lib# ls -l libc* libm*
lrwxrwxrwx 1 root root 14 Jul 21 18:32 libc.so.5 -> libc.so.5.3.12*
-rw-r-xr-x 1 root root 562683 May 19 1995 libc.so.5.0.9*
-rw-r-xr-x 1 bin bin 583795 Apr 25 07:15 libc.so.5.3.12*
lrwxrwxrwx 1 root root 13 Jul 21 18:32 libm.so.5 -> libm.so.5.0.6*
-rw-r-xr-x 1 root root 35942 May 19 1995 libm.so.5.0.0*
-rw-r-xr-x 1 bin bin 36156 Apr 25 07:15 libm.so.5.0.6*
```

Az előbbi példában a 4-es fő és az 5.24 alverziószámmal és javítási szinttel rendelkező könyvtár egy olyan könyvtárra cserélődik ki, amely ugyanezzel a főverziószámmal és az 5.26 alverziószámmal és javítási szinttel rendelkezik.

A csatolások manuális módosítását, vagyis az `ln` parancssal régi csatolások törlését majd pedig új csatolás létrehozását nem javasoljuk. Az `ln` parancs ugyanis maga is használja a C könyvtárat, így a csatolás törlésekor új csatolást nem lehet létrehozni. Hasonlóan a többi parancshoz, amelyek ugyancsak a C könyvtártól függnek, a továbbiakban ezeket nem lehet meghívni. A csatolás ismételt létrehozásához a rendszert betöltőlemezről kell újraindítani (pl. a disztribúciós cso-magról) a merevlemez gyökérfájlrendszerét újra be kell kapcsolni a rendszerbe, majd ott a csatolást újra létre kell hozni. A betöltőlemez létrehozását az 5.5 rész tárgyalja.

### Kernel

A legújabb kernel forráskódját a legtöbb Linux FTP szerverről le lehet tölteni. Ez azonban először a `ftp.funet.fi`, vagyis a finn szerveren jelent meg. A kernel legújabb verziójának telepítésekor először a `/usr/src/linux` könyvtár teljes tartalmának törlését kell elvégezni, majd a kicsomagolt kernel forráskódját tartalmazó tar archívot a `/usr/src` alkonyvtárba. A folyamat során a `linux` alkonyvtár újra létrejön. Ezután – ahogy ezt a kernel konfigurálásával és fordításával foglalkozó rész (6.2. alfejezet) tárgyalta – a felhasználandó kezelőprogramok a `make config` parancssal definíálhatók, majd a kernel lefordítatható.

## 7.11 BETÖLTŐLEMEZEK

Nemcsak a kezdő felhasználókkal fordulhat elő, hogy a fontos fájlokat letörlik és a rendszert indítás közepénne teszik. Ezen fájlok közé elsősorban a kernel image-fájlok tartoznak, amelyeket a LILO tölt be, shellek és az `/etc` könyvtár fájljai. A fontos fájlok törlése esetén a rendszer kijavítására számos lehetőség nyílik.

Ha a rendszert még egyszerűsítő módban sem lehet elindítani (lásd a 7.2 részt), akkor a helyzet megoldásához betöltőlemezre van szükség. Ha a telepítőlemezek vagy a Slackware, Universe, Lst vagy SLS disztribúció CD-je rendelkezésre áll, akkor a Linux rendszer innen betölthető. A telepítőprogram elindítása helyett a gyökérpartíciót illesztük be logikailag a rendszerbe, és a hiányzó fájlokat állítunk elő vagy másoljuk be. Ha a fontos fájlokról akár lemezen vagy származtatott másolattal rendelkezünk, akkor az ebben a helyzetben különösen hasznos lehet.

Ha a merevlemez csak a kernel sértült meg, akkor egy másik rendszer működőképes kernelje követelni hajlékonylemezre másolható. Ez tehát csak a kernelt tartalmazza és semmilyen más egyéb fájlt nem. A betöltés erről a lemezről végrehajtható. A behúzás után a kernel kísérletet tesz a gyökérfájlrendszer logikai rendszerbe illesztésére. Az az eszköz vagy partició, amelyet a kernel megkísérli logikailag a rendszerbe illeszteni, az `root` programmal állítható be (lásd az 5.5 részt).

Ha az `/etc` könyvtárban levő fontos fájlok sérültek meg, és a betöltés a gyökérpartícióból lehetetlen, akkor olyan lemezre van szükség, amely saját fájlrendszeret tartalmaz. A betöltő kernel akár ezen, vagy egy másik lemezről is lehet. Elegánsabb megoldás, ha egyetlen betöltőlemezen van mind a kernel, minden gyökérfájlrendszer. Az ilyen lemez lehetővé teszi a minimális, azonban független Linux rendszer betöltését, amelyről a merevlemez-partíciók logikailag beilleszthetők a hiba kijavítására.

Ha a lemez gyökérfájlrendszerként használatos, akkor azt a meghajtóból nem szabad kivenni! Ez azonban meglehetősen kényelmetlen, ha további lemezeket is be kell olvasni. Ebben a helyzetben a virtuális lemez nyújtja a megfelelő megoldást. A betöltéskor a lemezről erre kell rámasolni, így a hajlékonylemez meghajtójára felszabadul.

Az ilyen betöltő/gyökér lemez létrehozása nem túlságosan bonyolult. Először is egy formázott lemezre van szükség, amelyet a DOS `format` parancsával, vagy a Linux `fdformat` parancsá-

val is létre lehet hozni, azonban már gyárilag formázott lemezek is léteznek. Általában a lemezén a MINIX fájlrendszer készítünk.

```
nicetry:~# fdformat /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... done
nicetry:~# mkfs -t minix /dev/fd0 1440
480 inodes
1440 blocks
Firstdatazone=19 (19)
Zonesize=1024
Maxsize=268966912
```

Ezután következik a lemez logikai beillesztése és a legfontosabb könyvtárak létrehozása:

```
nicetry:~# mount -t minix /dev/fd0 /mnt
nicetry:~# mkdir /mnt/etc,bin,sbin,boot
nicetry:~# mkdir /mnt/dev,lib,root,mnt
```

Az eszközkezelő fájlokat a /dev könyvtárban kell elhelyezni, amelyhez itt a cp parancs használható. A legfontosabb programokat, könyvtárakat és egyéb fájlokat kell bemásolni és a szimbolikus csatolásokat kell létrehozni.

```
nicetry:~# cp -a /dev/* /mnt/dev/
...
nicetry:~# cp /bin/bash,cp,cat,ln /mnt/bin
nicetry:~# cp /bin/loadkeys,ls,mkdir,rmdir,rm /mnt/bin
nicetry:~# (cd /mnt/bin; ln -s bash sh)
nicetry:~# cp /sbin/fdisk,mke2fs,mkswap,mount /mnt/sbin
nicetry:~# cp /sbin/reboot,shutdown,umount,update /mnt/sbin
nicetry:~# cp /boot/boot.b /mnt/boot
nicetry:~# cp /lib/ld.so,libc.so.5.3.12 /mnt/lib
nicetry:~# (cd /mnt/lib; ln -s libc.so.5.3.12 libc.so.5)
```

A C könyvtár verziószáma eltérhet az előző példában alkalmazott 4.5.26 verziószámtól. Az a fontos, hogy a szimbolikus csatolás – ami csak a fő verziószámot tartalmazza – úgy jöjjön létre, hogy valóságos könyvtárhoz mutasson, ahogy ez a merevlemez esetében is volt.

Ezután az /etc könyvtárban elhelyezendő legfontosabb fájlokat kell létrehozni. A következőkben ezeket a fájlokat soroljuk fel és a tartalmukat is megadjuk:

/mnt/etc/group:

```
root::0:root
bin::1:root,bin,daemon
daemon::2:root,bin,daemon
```

```

sys::3:root,bin,adm
adm::4:root,adm,daemon
tty::5:
disk::6:root,adm
lp::7:lp
mem::8:
kmem::9:
wheal::10:root
floppy::11:root
mail::12:mail
news::13:news
uucp::14:uucp
man::15:man

```

/mnt/etc/passwd:

```

root::0:0:root:/bin/bash
bin:*:1:1:bin:/bin:
daemon:*:2:2:daemon:/sbin:
adm:*:3:4:adm:/var/adm:
lp:*:4:7:lp:/var/spool/lpd:
sync:*:5:0:sync:/sbin:/bin/sync
shutdown:*:6:0:shutdown:/sbin:/sbin/shutdown
halt:*:7:0:halt:/sbin:/sbin/halt
mail:*:8:12:mail:/var/spool/mail:
news:::9:13:news:/usr/lib/news:
uucp:*:10:14:uucp:/var/spool/uucppublic:
operator:*:11:0:operator:/root:/bin/bash
games:*:12:100:games:/usr/games:
man:*:13:15:man:/usr/man:
postmaster:*:14:12:postmaster:/var/spool/mail:/bin/bash
nobody:*:65535:100:nobody:/dev/null:
ftp:*:404:1::/hawk/ftp:/bin/bash
guest:*:405:100:guest:/dev/null:/dev/null

```

/mnt/etc/fstab:

```

/dev/ram   /    minix    defaults    1 1

```

/mnt/etc/rc:

```

# initial path
PATH=/sbin:/bin

```

```
# start update
update &

# create new utmp
cat /dev/null >>/etc/utmp

#create mtab
mount -av
```

/mnt/etc/profile:

```
export PATH=/bin:/sbin
```

Ahhoz, hogy erről a lemezről a behúzás lehetővé váljon, a kernelt rá kell másolni, és a Linux Loader telepíteni kell. Ehhez egy speciális lilo konfigurációs fájlról van szükség (lilo.conf.bd), amelyet a következőkben meg is adunk. A fájlt és a lilo programot is a lemezre kell másolni.

A gyökérfájlrendszer definícióját és kernelben a virtuális lemezt az rdev segédprogram kezeli (lásd még az 5.5 részt).

```
nicetry:/root# cp /vmlinuz /mnt/
nicetry:/root# rdev /mnt/vmlinuz /dev/fd0
nicetry:/root# rdev -R /mnt/vmlinuz 0
nicetry:/root# rdev -r /mnt/vmlinuz 1440
```

lilo.conf.bd:

```
boot = /dev/fd0
install = /mnt/boot/boot.b
map = /mnt/boot/map
compact
image = /mnt/vmlinuz
root = /dev/fd0
label = linux
```

A lilo program az új konfigurációs fájllal kerül meghívásra. Ez felülírja a betöltő szektorit azon a lemezén, amelyen a Linux Loader van és a lemez már készen áll a behúzásra.

```
nicetry:/root# lilo -c lilo.conf.bd
Added linux
```

Ezzel a behúzólemez létrehozása befejeződött, amely az umount parancssal a Linux könyvtárfából is eltávolítható.

# AZ X WINDOW SYSTEM

**A** grafikus munkaállomások tömeges elterjedésekor a grafikus felhasználói felület (graphical user interface, GUI) programozásához semmiféle szabvány sem létezett. A legtöbb gyártó saját GUI rendszerét szállította a számítógépekkel. Ha egy alkalmazást, amelyet az új gépek grafikus képességeinek elterjesztésére szántak, több platformon kívántak futtatni, akkor azt több vállalatban kellett kifejleszteni és karbantartani. Azok a nagyobb intézmények, amelyek különböző gyártók rendszereivel foglalkoztak, különösen éreztek ennek a hátrányait.

Ez a helyzet készítette a Massachusetts Institute of Technology (MIT) intézetben az Athena Project szervezőit, hogy kidolgozzanak egy platformuktól független, egységes környezetet a grafikus alkalmazások fejlesztéséhez. Az első időkben az X Window System fejlesztését csak a DEC és az IBM támogatta anyagilag.

1987 januárjában 12 neves, munkaállomásokat gyártó vállalat összefogásával született az X Consortium. Az intézmény célja az volt, hogy támogassa az X Window System fejlesztését és szabványosítását, valamint lehetővé tegye a kereskedelmi hasznosítását. Ugyanebben az évben megjelent az X Window System Version 11 Release 1 (rövidítve X11 R1 vagy X11). A korábbi verziókkal ellentében a 11-es verzió már kilepett a kutatási stádiumból. Bár az új változat már nem volt kompatibilis a 10-es verzióval, de lényegesen nagyobb rugalmasságot és teljesítményt képviselt. Az X11 kereskedelmi áttörést jelentett, és hamarosan a gyártók széles körében a UNIX rendszerek grafikus felhasználói felületeinek szabványává vált.

## 8.1 JELLEMZŐK

Az X Window System olyan jellemzőkkel rendelkezik, amelyek jól megkülönböztetik a hagyományos grafikus felhasználó felületektől (ilyen például az Apple Finder és az MS-Windows). Az itt következő részeken e hatékony rendszer legfontosabb jellemzőit vessük sorra.

### Nyitottság

A legtöbb egyéb GUI rendszerrel szemben az X Window System a kezdeteiktől nyitott rendszerként fogalmazódott meg. Ez azt jelenti, hogy a fejlesztők megtartották függetlenségüket a gyártók összes egyedi kívánalmaitól, valamint azt, hogy a teljes forráskódot szabadon hozzáérhetővé tették.

Az X Window System a hordozható és hardvertől független szoftverek kényelmes fejlesztési környezetét biztosítja. A programozónak nem kell törölnie a hardver platformokkal. A rendszerben számos bemeneti és kimeneti eszköz használható. A gyártótól függő kiterjesztésekhez rendelkezésre álló interfések speciális hardver-csatlakoztatását teszik lehetővé.

A gyártótól való függetlenség és a teljes körű hordozhatóság alapján az X Window System a munkaállomás szektor magas szintű elismérését és elfogadottságát érte el. Ma már a PC-től a nagy-széless körű elterjedtség előnyt jelent a Linux felhasználói számára is. A Linux alatt futó XFree86 X Window System szerver igen nagy teljesítményre képes a szokásos PC hardveren. Kétségtelen, hogy az X11 egyenértékű a kereskedelmi X szerverekkel, és számos területen elsőrangú tel-

## **Ügyfél (kliens) kiszolgáló (szerver) architektúrák**

Belső felépítésének köszönhetően az X Window System megkülönbözteti az X kiszolgálót (szervert) és az X ügyfeleket (klienseket). A kiszolgáló program a helyi hardver (például monitor, billentyűzet és egér) kezeléséért felelős, és a megfelelő felületet biztosítja a felhasználó és az egyedi X alkalmazások, azaz az X kliensek között.

Szokásos esetben a munkaállomások csak egy szervert futtanak, amely tetszőleges számú kliensnek biztosít bemenetet, és kezeli az ügyfelek által kért megjelenítéseket a képernyőn. Lehetőség van arra is, hogy egyetlen szerver egy munkaállomásra kapcsolt több monitort is kezeljen, ami különösen a CAD rendszereknél lehet hasznos.

### **Az X protokoll**

Az X protokoll biztosítja az egyetlen kapcsolatot a kiszolgáló és az X ügyfelek között. E szabványos protokollnak köszönhetően az ügyfelek akár a hálózat más számítógépein is futhatnak (lásd a Hálózat-átlátszóság című részt).

Fontos tisztázni, hogy az X kiszolgáló és az X ügyfél fogalom jelentése eltér az általában megszokottól. A kiszolgáló (szerver) általában olyan számítógép, amely a hardver szempontjából az ügyfelei felett áll, s azok adatlekérdezéseit és számításait gyorsabban dolgozza fel. Az X Window System alatt ez általában fordítva igaz: a kiszolgáló futtatja az előfeldolgozót (ez a szokásos munkaállomás), míg az ügyfél (kliens) – egy lényegesen nagyobb teljesítményű gép – jelenti a háttoldali processzort. Nagyon gyakran ez a megkülönböztetés fölösleges, mivel a kiszolgáló és az ügyfél ugyanazon a számítógépen fut.

Az ügyfél és a kiszolgáló közötti különböztetések következményei vannak a programozó és felhasználó számára is. Például a programozónak egy olyan grafikát, amellyel az ügyfél rendelkezik bittérkép formátumban, megjelenítése előtt át kell vinnie a hálózaton keresztül. Ezeket a kölcsönös kapcsolatokat figyelembe kell venni az X11 alkalmazások fejlesztéskor, hogy elkerülhessük a szükségtelen hálózati forgalmat, amely a végrehajtási sebesség csökkenését okozhatja.

Mivel az X ügyfél meglehetősen lazán csatolt a kiszolgálóhoz, a gyakorlatban felhasználót időnként meglepetések érhetik. Ha például egy ügyfél a nagy hálózati vagy CPU-terhelés miatt nem válaszol azonnal egy felhasználói kérésre, a legtöbb felhasználó hajlamos megismételni a bevitelt. Mivel azonban az X kiszolgáló az összes műveletet rögzíti, és gondosan továbbítja az ügyfélnek, a művelet ismételten végrehajtásra kerülhet, ami a legritkább esetben egyezik meg a felhasználó szándékával.

### **Hálózat-átlátszóság**

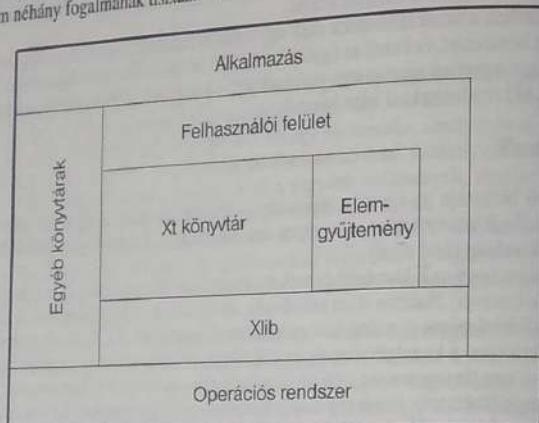
Az X Window System lényeges jellemzője a hálózat-átlátszóság. A UNIX platformon a legtöbb munkaállomás hálózaton keresztül kapcsolódik egymáshoz. Az X Window System egy mechanizmus segítségével lehetővé teszi az alkalmazások grafikus kimenetének a hálózat tetszőleges munkaállomására való átirányítását. Ez a szolgáltatás lehetővé teszi, hogy a számításigényes programokat a legnagyobb teljesítményű gépen futassuk, míg a bemenet és a kimenet kezelését a hálózat kisebb munkaállomására bízhatjuk.

Ha kiélegítően gyors hálózati összeköttetés áll rendelkezésre, akkor a két gép egymástól akár több ezer kilométeres távolságban is elhelyezkedhet. Bár az X Window System jelenleg csak a TCP/IP és a DECnet protokoll használatát teszi lehetővé, elvileg egyetlen hálózati protokollohoz sem kötött.

Meglepő, hogy ez a hálózat-átlátszóság a hagyományos grafikus rendszerekkel összehasonlítva alig vezet a végrehajtási sebesség csökkenéséhez. Az X Window System lehetővé teszi a helyi alkalmazások, valamint a hálózat távoli gépein futó programok megjelenítését egyetlen monitoron. Így az X Window System megfelelő alapja lehet az összes lehetséges alkalmazási terület integrálásának.

## 8.2 A FELÉPÍTÉS

Az X Window System felépítése számos szintből áll (lásd a 8.1 ábrán). Jóllehet a felhasználó általában észre sem veszi ezeket a rétegeket, a felépítés megismerése segítséget jelenthet az X Window System néhány fogalmának tisztázásában.



8.1 ábra. Egy X11 alkalmazás felépítése

Az X Window System alapját az Xlib nevű, szabványos C interfésszel rendelkező kiterjedt grafikus könyvtár adja. Az Xlib könyvtár minden hívását a rendszer megfelelő adatelemek sorozatára fordítja le, amelyet azután a hálózaton keresztül továbbít (X protokollsint), és ez más munkaállomásokon értelmezhető. Ez egyúttal a problémamentes kapcsolatot is biztosítja a különböző gyártók munkaállomásai között.

Az X11 nem tesz lehetővé közvetlen hozzáférést a videohardverhez, és így semmilyen módon sem kerülhető meg a szabványos interfész.

### Belső elemek

Mivel az Xlib csak az alapvető grafikus műveletek végrehajtását teszi lehetővé (például vonalak, körök vagy kitöltött alakzatok rajzolása), magasabb szinteket is bevezettek. Az X Toolkit bevezetésével az MIT kiérte javaslatot tett az ilyen könyvtárak megvalósítására. Grafikus felhasználói felület létrehozásához a fejlesztő általában objektumokat használ az eszközkiészlet szintjén. Ha egy alkalmazás további elemi grafikus kimeneti műveleteket igényel, akkor ezeket közvetlen Xlib hívásokkal lehet megoldani.

Az X Toolkit Intrinsic Library (Xt könyvtár) összetett grafikus objektumok, például parancsgombok, szövegbeviteli mezők és menüálaszték fejlesztését és használatát teszi lehetővé. Ezeket az objektumokat összefoglaló néven *elemgyűjteménynek* (widgets) nevezik.

### Elemgyűjtemény

Az ilyen objektumok megjelenésének és kinézetének további részleteit a belső elemkönyvtárak már nem határozzák meg. Az X Window System fejlesztésének tervezési célja nem az volt, hogy előirja a raja felépített alkalmazások kinézetét, hanem csak az ilyen elemgyűjtemények megvaló-

sításához és használatához kívánt megfelelő kezelőfelületet biztosítja. Ily módon az idők során számos helyen készítettek ilyen elemkönyvtárakat, amelyek időnként megjelenésükben lényeges eltérésekkel mutatnak. Egy idő után azonban ez a sokszínűség zavarni kezdte a felhasználókat.

A legújabb fejlesztések azt mutatják, hogy a legtöbb gyártó megegyezett abban, hogy az Open Software Foundation Motif (OSF/Motif) elemgyűjteményét fogadja el tényleges szabvánnyal (pl. a Sun Microsystems OpenLook elemgyűjteményével szemben). Így a jövőben az várható, hogy az X11 alatt egységes elvek alapján fognak felépülni a grafikus felhasználói felületek. (Ez az eredeti könyv megjelenése óta már eldült, Common Desktop Environment, CDE néven elkészült és szabványos lett az elemkészlet – a lektor.)

### 8.3 AZ X ERŐFORRÁSOK

A tervezés során az X Window System fejlesztői példátlanul rugalmas felületet hoztak létre a rendszer különféle paramétereinek konfigurálására. Az alap az X Toolkit objektumorientált megközelítése és az ezen felépített elemgyűjtemény volt.

#### **Az elemek attribútumai**

Minden vezérlőelem számos egyedi attribútummal rendelkezik, ilyen a pozíció, a méret, az alak és a szín. Ezeket az attribútumokat a programozó, valamint később a felhasználó is módosíthatja. minden grafikus objektum belső alapértelmezett beállításokkal rendelkezik, amelyeket a vezérlőelem fejlesztője határozott meg. Egy X alkalmazás programozója általában csak az egyedi igényekhez feltétlenül szükséges mértékben módosítja ezeket az alapbeállításokat.

#### **Erőforrásfájlok**

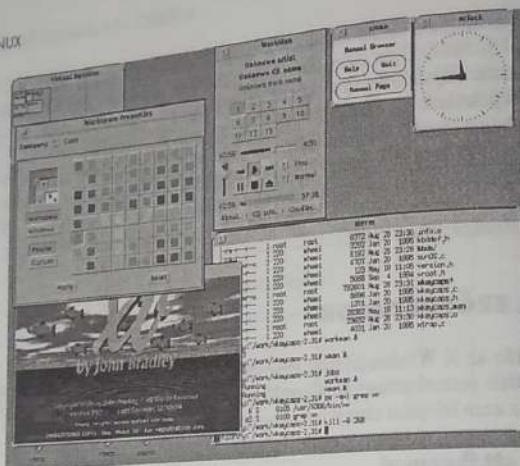
Egy X alkalmazás betöltésekor az **erőforrás-kezelő** az összes új adatot betölti az X erőforrás-adatbázisból, amelynek tartalmát a felhasználó számos helyen módosíthatja.

### 8.4 ABLAKKEZELŐK

Az ablakkezelő egy különleges ügyfél, amely X kiszolgálónak csak egyszer fordul elő. Feladata egy megjelenítő különböző ablakainak kezelése, lehetővé téve, hogy a felhasználó módosíthassa az ablakok helyét és méretét. Az ablakkezelő foglalkozik továbbá a dekorációval is, amely az ablak külső területeit és különböző vezérlőelemeit jelenti. Természetesen az ablak tartalmáért már a megfelelő X ügyfél a felelős.

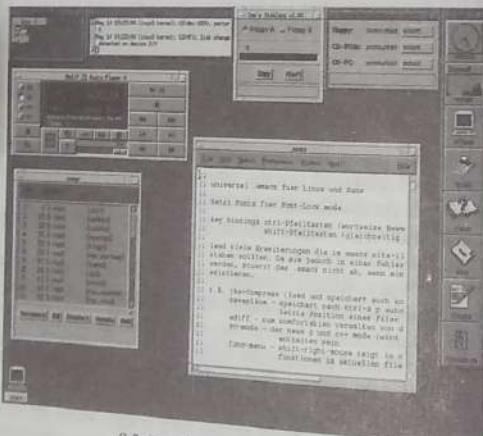
Az X Window System az ablakkezelő megjelenését sem írja elő. Egy adott ablakkezelő és a vezérlése alatti futó ügyfelek közötti problémamentes kapcsolat biztosítására az X Consortium az ICCC (Inter-Client Communications Conventions) rendszert kínálja.

A felhasználóknak a különböző ablakkezelők széles választéka áll rendelkezésre: mindegyiknek van előnye, és van hátránya. A Linux tartalmazz például a `twm` (Tom's Window Manager) ablakkezelőt, amely nem tartozik a legkényelmesebb működtethető rendszerek közé. Az Open Software Foundation (OSF) létrehozta a **Motif** ablakkezelőt (`mwm`), amely jól illeszkedik a Motif elemgyűjteményhez. A Sun Microsystem **OpenLook** Window Manager (`olwm`) forráskódja ingyenesen rendelkezésre áll. Az `olwm` ablakkezelőt különböző fejlesztő bővíti: az így megszületett **OpenLook Virtual Window Manager** (`olvwm`) szinte tetszőleges számú virtuális képernyőt biztosít, és az ezek közötti váltás a rendszerkezelőből végrehajtható (lásd a 8.2. ábrán).



8.2 ábra. Az olvwm ablakkezelő

A legtöbb Linux változat a Robert Nation fejlesztette fvwm ablakkezelőt részesít elönnyben (lásd a 8.3. ábrán). Ez az ablakkezelő is a twm megoldáson alapul, de lényegesen kevesebb memóriát igényel, mint az eredeti. A konfigurációs fájl (fvwmrc) és számos paraméter beállításával a felhasználó lényeges mértékben befolyásolhatja az fvwm megjelenését. Ezzel a lehetőséggel a felhasználó utánozhatja az mwm vagy a Silicon Graphics ablakkezelőjének megjelenését. Az olvwm ablakkezelőhöz hasonlóan itt is több képernyő használható.



8.3 ábra. Az fvwm ablakkezelő

Említtést érdemel az a jellemző is, amely különböző modulok csatolását teszi lehetővé, és így az ablakkezelővel definált interfészen keresztül történik a kommunikáció. Ilyen modul a GoodStuff ikontár, amelyen a felhasználó ikonokat helyezhet el a leggyakrabban használt alkalmazásokhoz, és így

ezek innen hívhatók. Egy másik modul adott felhasználói műveletekhez, például egy ablak megnyitása vagy bezárasa, hangok hozzárendelését teszi lehetővé. További lehetőség a generic window manager (gwm), amelynek felépítését a felhasználó a két szabványos változat (OpenLook és Motif) közül bármelyikhez igazíthatja a beépített Lisp értelmező (interpreter) segítségével.

A twm egy további utódja a ctwm, amely lényegesen kellemebb és jobban konfigurálható környezetet biztosít elődjénél.

## 8.5 KÖNYVTÁRAK ÉS SEGÉDLETEK

Ez a szakasz az X Window System alatt a grafikus felhasználói felületek készítésénél használható legfontosabb eszközöket (könyvtárakat) mutatja be.

### **Az Athena elemgyűjtemény**

Az X Window System számára rendelkezésre álló első elemgyűjtemény az **Athena** gyűjtemény volt, amely azóta is kvázi-szabvány maradt, és része a MIT változatnak. Számos korai X alkalmazás épült az Athena vezérlőelemekre. Még ma is sok szabadon terjeszthető program használja ezeket.

Néhány elemének meglehetősen régies megjelenése a kereskedelmi alkalmazások fejlesztőit korszerűbb eszközök készletének létrehozására készítette. A belső elemkészlet hasonlósága miatt ez nem jelent nagy problémát. Újabban az Athena elemkészlet is változóban van: a szabadon letölthető **forráskód** lehetővé teszi a rajzolórutinok lecserejét. Ezek a módosítások a Motif rendszerhez hasonló háromdimenziós megjelenést eredményeztek. Érdekes, hogy a Linux rendszerben a régi könyvtár (`libXaw`) egyszerűen helyettesíthető az új könyvtárral (`libXaw3D`).

### **OpenLook**

Az OpenLook csak a grafikus felhasználói felület megjelenésének részletes specifikációját tartalmazza. Több éves fejlesztés után az AT&T és a Sun Microsystems jelentette meg azzal a céllal, hogy létrehozzák az X11 alatti grafikus felhasználói felület szabványát.

Az OpenLook specifikáció alapján a Sun eszközök készletet fejlesztett ki az X Window System számára (**XView**), amely majdnem teljesen kompatibilis a Sun korábbi grafikus rendszerével (**SunView**). Ez az eszközök készlet bekerült az X11 MIT változatának 4-es kiadásába. Az XView azonban közvetlenül az Xlib könyvtárra épül, ami azt jelenti, hogy nem valódi elemgyűjtemény. A Sun ezt az elemkészletet beépítette számos alapvető alkalmazásába, és ezeket minden Sun munkaállomással szállítja (**OpenWindows Desktop**).

Az AT&T a UNIX System V rendszerekkel szállítja az OpenLook grafikus felhasználói felületet, így ezzel nagy mértékben hozzájárult széles körű elterjedéséhez. Az XView alternatívjaként, elsősorban az AT&T fejlesztési eredményének köszönhetően, megjelent egy másik X Toolkit alapú elemgyűjtemény is, amely szintén kielégítí az OpenLook specifikációt.

### **OSF/Motif**

Az új technológiák és szabványok kifejlesztésének érdekében a legtöbb híres UNIX gyártó (például a Hewlett-Packard, az IBM és a DEC, de a Sun és az AT&T nem) fokozta erőfeszítéseit az Open Software Foundation (**OSF**) keretében. Ez a csoport felismerte egy egységes felhasználói felület szükségességét az X11 alatt, és kezdeményezte az OSF/Motif kifejlesztését. A fejlesztésben és a megvalósításban elsősorban a DEC és a HP vett részt.

A néhány hónapos munka után megszületett a felhasználói felületet, amely a PC szektor általában GUI megjelenését tükrözte. Központi eleme a Motif ablakkezelő (`mwm`) és a belső X elemeken

alapuló elemgyűjtemény. Emellett létrehozták a User Interface Language (UIL) nyelvet is, amely a Motif alapú grafikus felhasználói felületek egyszerű leírását teszi lehetővé. Ez a leírás egy speciális fordítóprogrammal bináris formátumként alakítható, és egy könyvtár segítségével értelmezhető. A Motif grafikus felhasználói felületek interaktív létrehozására számos gyártó kínál programokat.

Az OSF tevékenységeben részt vevők száma és súlya miatt a Motif lényegében a UNIX világ-szabványává vált. Az OpenLook fejlesztői, a Sun Microsystems és az AT&T (később a UNIX System Laboratories és a Novell) 1993 tavaszán csatlakozott a Motif-trendhez a COSE Initiative (Common Open Software Environment) kezdeményezéssel. Ez az OpenLook végét jelentette, legalábbis a kereskedelmi szektorban. A COSE elsőleges célja a Motif alapú CDE (Common Desktop Environment) környezet szabványosítása. (Ami azóta sikerült – a lektor.)

Mivel az OSF a Motif forráskódját nem bocsátotta ingyenesen rendelkezésre, a felhasználóknak minden egyes futtatási jogért fizetni kell. Éppen ezért kezdetben a Motif nem volt elérhető a Linux alatt. Az OSF/Motif Linux verziójai azonban elfogadható áron kaphatók. Az egyetemek a forráskódot közvetlenül az OSF-től vehetik meg, a munkaállomásain végrehajtják a fordítást, amelyet számos helyen megtettek a Linux rendszereken.

## SUIT

A SUIT (Simple User Interface Toolkit) rendszert a Virginia Egyetemen fejlesztették ki. A legtöbb X eszközökkel szemben ez más általánosan használt felhasználói felületek (például MS-Windows és Apple Finder) számára is rendelkezésre áll. A SUIT ezen operációs rendszerek között átvihető alkalmazások fejlesztését teszi lehetővé. Megjelenése nagyon hasonlít az OSF/Motif környezetre, amely segíti elfogadását a felhasználók körében.

A könyvtár különleges jellemzője az integrált felületszerkesztő, amely lehetővé teszi a felület egyszerű és interaktív kezelését. Például a felhasználó az objektumok helyét és alakját az alkalmazás újrafordítása nélkül módosíthatja. Amikor a felhasználó kilép az alkalmazásból, az összes módosítás automatikusan fájlba íródik.

## InterViews és Fresco

Az InterViews hatékony grafikus környezet, amelyet a Stanford Egyetemen fejlesztettek ki. Az X Toolkit rendszerrel és elemgyűjteményével szemben az InterViews C++ interfésszel rendelkezik, és íly módon az objektumorientált koncepciót követi. A grafikus könyvtár mellett az InterViews rendszer egy interaktív szerkesztőt (ibuild), egy WYSIWG szövegszerkesztőt (doc), egy C++ osztálybongészőt (iclass) és egy hatékony, vektororientált rajzolóprogramot (idraw) is tartalmaz. Éppen ezek miatt az InterViews csak megfelelő hardveren fut.

Az X11 6-os verziója új Fresco eszközökészletet is tartalmaz, amely az InterViews továbbfejlesztése. A Fresco lehetővé teszi a hálózat telszöleges számítógépen futó más programokból grafikus objektumok beágyazását. A felület a COBRA (IDL) nyelvnek megfelelő nyelvvel írható le.

## XView, Slingshot és UIT

Az XView nem az X belső elemeken alapuló, figyelmetlen eszközökészlet. A Sun Microsystems termékét az X Window System 4-es verziójával szállították. A programozási interfész nagymértékben meglehetősen régi Sun View rendszeren alapult, amely a korai Sun munkaállomások hálózati átlát-szóságán nem biztosít, erősen rendszerfüggő grafikus felhasználói felülete. Az XView az OpenLook megjelenését vette át, és programozása az objektumorientált megközelítés következetében meglehetősen egyszerű.

Az XView kiterjesztéseként a Sun Microsystems egy angliai dolgozója kifejlesztette a Slingshot bővítményt. Ez a csomag számos új grafikus objektumot tartalmaz, amelyek harmonikusan illeszked-

nek az alaprendszerbe. A Slingshot az objektumorientált megközelítést elemi grafikus elemekkel bővíti (például vonalak és téglalapok), valamint a programozást is bővíti az elemek áthelyezésében az egér húzásának lehetőségével.

A programozási nyelvek területén az XView (és a Slingshot) beillesztését az új fejlesztésekbe szintén egy Sun Microsystems alkalmazott végezte el egy C++ objektumhierarchia kifejlesztésével. Az UIC lehetővé teszi az összes XView objektum alkalmazását, és használatukat egyszerűsíti. Sajnos a függvények integrálása a származtatott C++ osztályokba némi problémával jár.

## 8.6 AZ X11 KISZOLGÁLÓ (SZERVER)

Minden Linux változat központi eleme az XFree86 csoport X kiszolgálója. Az XFree86 nonprofit szervezet, amely elsősorban az Intel-alapú UNIX rendszereken az X kiszolgáló továbbfejlesztésével foglalkozik. Az XFree86 különös figyelmet fordít a legfrissebb PC-s grafikus kártyák támogatására. Éppen ebből következik, hogy az **XFree86 kiszolgáló** nagyon jól jelzi a lehetséges teljesítményt, amely gyakran meghaladja a RISC munkaállomásoktól elvártat. Az SVGA kiszolgáló (XF86\_SVGA) az összes általánosan használt VGA grafikus kártyával, az összes lehetséges felbontásban és képfrekvenciánál használható. A különleges kiszolgálókkal, amelyek kihasználják a korszerű **gyorsítókártyák** (Mach, S3) lehetőségeit még jobb eredmény érhető el. A korszerűbb kártyákon az Xfree86 képes 16, 24 illetve 32 bit színmelésségen TrueColor grafikát megjeleníteni. (Az Xfree86 X szerverei sok kártyában képesek kihasználni az ún. Windows gyorsítót, ami valójában gyakori grafikus alakzatok megrajzolásához készített hardvereszköz, így a szoftveres megoldással szemben lényegesen nagyobb teljesítményt nyújtanak – a lektor.)

## 8.7 A LINUX MINT X TERMINÁL

A Linux és az XFree86 alatt futó PC kiválóan alkalmas X terminálként való használatra. Ez kiülönösen akkor érdekes, amikor drága szoftvertermék áll rendelkezésre egy munkaállomáson, és azt decentralizáltan kívánják kihasználni.

Számos olyan X kiszolgáló található, amely az MS-Windows alatt fut. Ezek teljesítménye azonban általában lényegesen az XFree86 kiszolgálók teljesítménye alatt marad, mivel az X protokoll parancsait a lassabb MS-Windows hívásokra fordítják le. (Másrészt az MS-Windows pénzbe kerül – a lektor.)

A Linux X terminálként való használatánál jelentkező lehetőségek jobb megértése érdekében egy ilyen alkalmazás példáját mutatjuk be heterogén hálózaton. Egy hálózatba (lásd a 9.3. ábrán) kapcsolt Sun munkaállomáson (**sun**) hatékony grafikus program áll rendelkezésre, amelyet egy Linux munkaállomás (**zeus**) használ. Először a külső hozzáférést kell engedélyezni a helyi X kiszolgáló számára az **xhost** parancssal. A jogosultságot gépenként külön lehet definiálni, de általában elegendő a következő megadás is:

```
nicetry:~$ xhost +
access control disabled, clients can connect from any host
nicetry:~$
```

Ezután a Linux felhasználó a **telnet** vagy az **rlogin** parancssal bejelentkezik a **sun** számítógépre:

```

nicetry:-$ telnet sun
Trying 192.168.1.20...
Connected to localhost.
Escape character is '^}'.

SunOS Unix (sun)

sun login: demo
Last login: Wed Aug 28 22:41:17 on tty2
SunOS Release 4.1.3 (GENERIC) #1: Wed Dec 23 11:02:57 MET 1995
sun:/home/display$ setenv DISPLAY nicetry:0.0

```

Ezt követően a DISPLAY környezeti változó beállítása történik úgy, hogy az X könyvtár az összes grafikus kimenetet a Linux gépre irányítson át. A C shellben ez a `setenv` parancssal történik, míg a Bourne shellben a következő módon:

```

nicetry:-$ export DISPLAY=sun:0.0

```

Ha ezután egy X ügyfél elindul a sun gépen, akkor az összes ablak nem ezen, hanem a Linux gépen (`zeus`) kerül megjelenésre. Igy a különböző munkaállomásokon futó több alkalmazás kimenete egy megjelenítőre küldhető, és ezek működése egy gépről vezérelhető. A Linux esetében a felhasználónak még a DISPLAY környezeti változó beállítását sem kell végrehajtani, mert ezt automatikusan kezeli a `telnetd`.

Alternatív megoldásként egy X alkalmazás a távoli számítógépen az `rsh` parancssal is indítható. Ehhez a megfelelő gépen az `.rhosts` fájlból egy bejegyzés szükséges (további részletek a 10.6. szakaszban találhatók). A következő utasítás elindítja az `xterm` alkalmazást a sun gépen, és a kimenetet átírányítja a zeus gépre. A `-display` kapcsoló szinte az összes X alkalmazásban használható.

```

nicetry:-$ rsh sun "xterm -display nicetry:0.0"

```

Külső alkalmazásból a kimenet átírányításának egyszerűsítésére érdemes egy elemet felvenni az ablakkezelő valamelyik menüjére. Ennek működését az `fvwm` konfigurálásánál, a 8.9. szakaszban mutatjuk be.

## 8.8 AZ X11 KONFIGURÁLÁSA

A Linux alatt az X Window System telepítése gyakorlatilag a programok és a fájlok kibontásából áll a különböző `tar` archívumokból. A legtöbb Linux csomagnál ez az operációs rendszer telepítésénél történik, így nem okoz különösebb nehézséget. A konfigurálás feladata akkor válik nehezebbé, ha az X kiszolgálót adaptálni kell a rendelkezésre álló videokártyához és a monitorhoz. Ekkor a konfigurálás az `/etc` vagy az `/usr/lib/x11` könyvtárban található `XF86Config` központi konfigurációs fájl módosítását teszi szükségesé.

## Az XF86Config fájl

Ez a fájl szakaszokra oszlik. A következőkben ezeket a szakaszokat soroljuk fel, és ismertetjük a szakaszok legfontosabb jellemzőit.

- **Files** – Az X kiszolgálónak szükséges, az RGB színtáblázatra és a betükészlet könyvtáraira mutató elérési útvonalakat definiálja.
- **ServerFlags** – A kiszolgáló általános jelzőit állítja be, például azt, hogy a kiszolgáló leállítható-e a <Ctrl-Alt-Backspace> billentyűkombinációval, és hogy a kiszolgáló hogyan reagál a UNIX jelzésekre.
- **Keyboard** – A csatlakoztatott billentyűzetet és a különleges billentyűk funkciót definiálja.
- **Pointer** – Az egér kezelőprogramját illeszi, megadva az egér típusát és a használt interfést.
- **Monitor** – A monitor(ok) határértékeit és időzítési adatait tartalmazza.
- **Device** – A videokártyát írja le.
- **Screen** – Monitort, definíciókat és videokártyát rendel egy X kiszolgálóhoz.

### Files szakasz

A telepítő csomagok általában megfelelően kezelik az RGB táblázat és a betükészletek elérési útvonalait. A betükészletek külön sorokba vagy vesszővel elválasztva vihetők be. A betükészlet-kiszolgálók megadása átviteli mód/gazdanév:portszám alakú, például `tcp/zeus:7100`. A következő példa egy files szakasz mutat be:

```
Section "Files"
    RgbPath      "/usr/X11R6/lib/X11/rgb"
    FontPath     "/usr/X11R6/lib/X11/fonts/misc/"
    FontPath     "/usr/X11R6/lib/X11/fonts/Typel/"
    FontPath     "/usr/X11R6/lib/X11/fonts/Speedo/"
    FontPath     "/usr/X11R6/lib/X11/fonts/75dpi/"
    FontPath     "/usr/X11R6/lib/X11/fonts/100dpi/"

EndSection
```

### ServerFlags szakasz

Ebben a szakaszban két beállítás lényeges. A `NoTrapSignal` csak a hibakeresésnél fontos, míg a `DontZap` beállításnál a kiszolgáló nem állítható le a <Ctrl-Alt-Backspace> billentyűkombinációval.

```
Section "ServerFlags"
    # Uncomment this to cause a core dump at the spot where a signal is
    # received. This may leave the console in an unusable state, but may
    # provide a better stack trace in the core dump to aid in debugging

    # NoTrapSignals
```

```

# Uncomment this to disable the <Ctrl><Alt><BS> server abort sequence
# This allows clients to receive this key event.

# DontZap

# Uncomment this to disable the <Ctrl><Alt><KP_+>/<KP_-> mode switching
# sequences. This allows clients to receive these key events.

# DontZoom

EndSection

```

### Keyboard szakasz

A billentyűzetkezelő beállítása történik ebben a szakaszban. Mindig szabványos protokollokkal kell használni. Fontos, hogy a nem angol billentyűzeteken a jobb oldali **<Alt>** billentyűt (**<AltGr>**) ModeShift billentyűként kell definiálni, hogy a különleges karakterek, például „@” vagy „!”, beírható legyen. Számos példafájl az amerikai billentyűzetet használja, és a jobb oldali **<Alt>** billentyűt Compose billentyűként definiálja.

```

Section "Keyboard"

Protocol      "Standard"
AutoRepeat   500 5
# Let the server do the NumLock processing. This should only be required
# when using pre-R6 clients
# ServerNumLock

# Specify which keyboard LEDs can be user-controlled (eg, with xset(1))
# Xleds        1 2 3

# To set the LeftAlt to Meta, RightAlt key to ModeShift,
# RightCtl key to Compose, and ScrollLock key to ModeLock:
LeftAlt      Meta
RightAlt     ModeShift
# RightCtl    Compose
# ScrollLock  ModeLock

EndSection

```

### Pointer szakasz

Az egér kezelőprogramjához általában elegendő megadni az egér típusát és a használt interfésszt. A telepítés rendszerint csatolást létesít ehhez az interfészhez (`/dev/mouse`), ami itt is használható.

```

Section "Pointer"
    protocol      "MouseSystems"
    Device        "/dev/cua1"

# When using XQUEUE, comment out the above two lines, and uncomment
# the following line.

#     Protocol      "Xqueue"

# Baudrate and SampleRate are only for some Logitech mice

#     BaudRate     9600
#     SampleRate   150

# Emulate3Buttons is an option for 2-button Microsoft mice
# Emulate3Timeout is the timeout in milliseconds (default is 50ms)

#     Emulate3Buttons
#     Emulate3Timeout     50

# ChordMiddle is an option for some 3-button Logitech mice

#     ChordMiddle

EndSection

```

### **Monitor szakasz**

Ez a szakasz többcélú. Ez tartalmazza a monitor határértékeit és időzítési adatait, és a szakasz többször előfordulhat. minden monitorhoz azonosító rendelhető, amellyel később hivatkozni lehet. A határértékek a vízszintes szinkronizálás, a függőleges frissítés és a sávszélesség maximális értékét tartalmazza.

Ezek az adatok a monitor műszaki dokumentációjában találhatók. Ha nincs másképp definiálva, a sávszélesség mértékegysége MHz, a vízszintes szinkronizálás KHz, míg a függőleges frissítésé Hz. Az indításkor a kiszolgáló ellenőrzi, hogy a kiválasztott üzemmód nem lépi-e túl a monitor határértékeit, és ha igen, az üzemmódot nem állítja be.

A műszaki adatok után a szakaszban az adott monitorhoz alkalmazható üzemmódok felsorolása következik. A videoüzemmódok ismertetését lásd később.

```

Section "Monitor"

Identifier      "My Monitor"
VendorName      "Unknown"
ModelName       "Unknown"

# HorizSync is in kHz unless units are specified.
# HorizSync may be a comma separated list of discrete values, or a
# comma separated list of ranges of values.

```

# NOTE: THE VALUES HERE ARE EXAMPLES ONLY. REFER TO YOUR MONITOR'S  
 # USER MANUAL FOR THE CORRECT NUMBERS.

HorizSync 31.5 - 64.3

# HorizSync 30-64                   # multisync  
 # HorizSync 31.5, 35.2              # multiple fixed sync frequencies  
 # HorizSync 15-25, 30-50            # multiple ranges of sync frequencies

# VertRefresh is in Hz unless units are specified.

# VertRefresh may be a comma separated list of discrete values, or a  
 # comma separated list of ranges of values.

# NOTE: THE VALUES HERE ARE EXAMPLES ONLY. REFER TO YOUR MONITOR'S  
 # USER MANUAL FOR THE CORRECT NUMBERS.

VertRefresh 40-150

# Modes can be specified in two formats. A compact one-line format, or  
 # a multi-line format.

# These two are equivalent

# ModeLine "1024x768i" 45 1024 1048 1208 1264 768 776 784 817 Interlace

```
# Mode "1024x768i"
#   DotClock      45
#   HTimings      1024 1048 1208 1264
#   VTimings      768 776 784 817
#   Flags         "Interlace"
# EndMode
```

# This is a set of standard mode timings. Modes that are out of monitor spec  
 # are automatically deleted by the server (provided the HorizSync and  
 # VertRefresh lines are correct), so there's no immediate need to  
 # delete mode timings (unless particular mode timings don't work on your  
 # monitor). With these modes, the best standard mode that your monitor  
 # and video card can support for a given resolution is automatically  
 # used.

# 640x400 @ 70 Hz, 31.5 kHz hsync	
Modeline "640x400"	25.175 640 664 760 800 400 409 411 450
# 640x480 @ 60 Hz, 31.5 kHz hsync	
Modeline "640x480"	25.175 640 664 760 800 480 491 493 525
# 800x600 @ 56 Hz, 35.15 kHz hsync	
Modeline "800x600"	36 800 824 896 1024 600 601 603 625
# 1024x768 @ 87 Hz interlaced, 35.5 kHz hsync	
Modeline "1024x768"	44.9 1024 1048 1208 1264 768 776 784 817
Interlace	

```

# 640x480 @ 72 Hz, 36.5 kHz hsync
Modeline "640x480"      31.5   640   680   720   864   480   488   491   521
# 800x600 @ 60 Hz, 37.8 kHz hsync
Modeline "800x600"      40     800   840   968  1056   600   601   605   628
+hsync +vsync

# 800x600 @ 72 Hz, 48.0 kHz hsync
Modeline "800x600"      50     800   856   976  1040   600   637   643   666
+hsync +vsync
# 1024x768 @ 60 Hz, 48.4 kHz hsync
Modeline "1024x768"     65    1024  1032  1176 1344   768   771   777   806
-hsync -vsync

# 1024x768 @ 70 Hz, 56.5 kHz hsync
Modeline "1024x768"     75    1024  1048 1184 1328   768   771   777   806
-hsync -vsync
# 1280x1024 @ 87 Hz interlaced, 51 kHz hsync
Modeline "1280x1024"    80    1280 1296 1512 1568 1024 1025 1037 1165
Interlace

# 1024x768 @ 76 Hz, 62.5 kHz hsync
Modeline "1024x768"     85    1024 1032 1152 1360   768   784   787   823
# 1280x1024 @ 61 Hz, 64.2 kHz hsync
Modeline "1280x1024"    110   1280 1328 1512 1712 1024 1025 1028 1054

# 1280x1024 @ 74 Hz, 78.85 kHz hsync
Modeline "1280x1024"    135   1280 1312 1456 1712 1024 1027 1030 1064

# 1280x1024 @ 76 Hz, 81.13 kHz hsync
Modeline "1280x1024"    135   1280 1312 1416 1664 1024 1027 1030 1064

# Low-res Doublescan modes
# If your chipset does not support doublescan, you get a 'squashed'
# resolution like 320x400.

# 320x200 @ 70 Hz, 31.5 kHz hsync, 8:5 aspect ratio
Modeline "320x200"      12.588 320   336   384   400   200   204   205   225
Doublescan
# 320x240 @ 60 Hz, 31.5 kHz hsync, 4:3 aspect ratio
Modeline "320x240"      12.588 320   336   384   400   240   245   246   262
Doublescan
# 320x240 @ 72 Hz, 36.5 kHz hsync
Modeline "320x240"      15.750 320   336   384   400   240   244   246   262
Doublescan
# 400x300 @ 56 Hz, 35.2 kHz hsync, 4:3 aspect ratio
Modeline "400x300"      18     400   416   448   512   300   301   602   312
Doublescan
# 400x300 @ 60 Hz, 37.8 kHz hsync

```

```

Modeline "400x300"      20    400 416 480 528   300 301 303 314
Doublescan
# 400x300 @ 72 Hz, 48.0 kHz hsync
Modeline "400x300"      25    400 424 488 520   300 319 322 333
Doublescan
# 480x300 @ 56 Hz, 35.2 kHz hsync, 8:5 aspect ratio
Modeline "480x300"      21.656 480 496 536 616   300 301 302 312
Doublescan
# 480x300 @ 60 Hz, 37.8 kHz hsync
Modeline "480x300"      23.890 480 496 576 632   300 301 303 314
Doublescan
# 480x300 @ 63 Hz, 39.6 kHz hsync
Modeline "480x300"      25    480 496 576 632   300 301 303 314
Doublescan
# 480x300 @ 72 Hz, 48.0 kHz hsync
Modeline "480x300"      29.952 480 504 584 624   300 319 322 333
Doublescan

EndSection

*****
# Graphics device section
*****
# Any number of graphics device sections may be present

# Standard VGA Device:

Section "Device"
    Identifier      "Generic VGA"
    VendorName     "Unknown"
    BoardName      "Unknown"
    Chipset        "generic"

# VideoRam       256
# Clocks         25.2 28.3

EndSection

```

### Device szakasz

Ez a szakasz a rendelkezésre álló videokártyákat adja meg. A monitor szakaszhoz hasonlóan többször előfordulhat. Számos kártyánál az alkatrészszöveg és az órajel megadása nem szükséges, mivel ezek az adatok az indításnál automatikusan begyűjthetők. Néhány bonyolultabb kártyánál azonban ezeket az adatokat is meg kell adni.

```

Section "Device"
    Identifier "Generic VGA"
    VendorName "Unknown"
    BoardName "Unknown"
    Chipset "generic"

    # VideoRam 256

    # Clocks 25.2 28.3

EndSection

# Sample Device for accelerated server:

# Section "Device"
#     Identifier "Actix GE32+ 2MB"
#     VendorName "Actix"
#     BoardName "GE32+"
#     Ramdac "ATT20C490"
#     Dacspeed 110
#     Option "dac_8_bit"
#     Clocks 25.0 28.0 40.0 0.0 50.0 77.0 36.0 45.0
#     Clocks 130.0 120.0 80.0 31.0 110.0 65.0 75.0 94.0
# EndSection

# Device configured by xf86config:

Section "Device"
    Identifier "My Video Card"
    VendorName "Unknown"
    BoardName "Unknown"
    #VideoRam 2048
    #Ramdac "att20c490"
    Clockchip "s3_sdac"
EndSection

```

Még ha a kiszolgáló automatikusan le tudja is olvasni az órajel-frekvenciát, bizonyos esetekben célszerű beírni ezt, mert az érzékelőt értéket azonosítóként használja a rendszer. A videoüzemmódok definíciója a monitor megadásának ModeLine soraira hivatkozik. Ha az órajel-frekvencia érzékelése bizonytalan, például 50 Hz helyett 49,5 a leolvastott érték, akkor az X kiszolgáló nem tudja azonosítani a videoüzemmódban használt frekvenciát, és hibaüzenettel kilép.

Emellett az órajel-frekvencia automatikus érzékelése bizonyos hardvereknél egyéb problémát is okozhat. A kártyadefiníciónál megadott órajel-frekvenciák tiltják az automatikus érzékelést. A rendelkezésre álló órajel-frekvenciák megállapításához távolítsa el az XF86config fájlból a clocks sorokat, és indítsa újra az X kiszolgálót a -probeonly kapcsolóval. Az X kiszolgáló ekkor az érzékelő órajel-frekvenciákat jeleníti meg szövegüzemmódban, majd kilép.

**Screen szakasz**

Ebben a szakasban az egyes kiszolgálók (SuperVGA, monokróm, S3 kiszolgáló stb.) rendelhetők egy monitorhoz és egy videokártyához. Amikor a kiszolgáló elindul, kiválasztja a megfelelő képernyőt, és így megkapja a videokártya és a monitor adatait. Emellett ez a szakasz a lehetséges videoüzemmódokat is tartalmazza, amelyek a monitorüzemmod megfelelő soraira hivatkoznak. A futás során a felhasználó az üzemmódok között a numerikus billentyűzeten kiadott <Ctrl-Alt- <numerikus\_mínusz>> és <Ctrl-Alt-<numerikus\_plusz>> billentyűkombinációval váltathat.

```

Section "Screen"
    Driver      "svga"
    Device      "Generic VGA"
    #Device     "My Video Card"
    Monitor     "My Monitor"
    Subsection  "Display"
        Depth      8
        #Modes     "640x480" "800x600" "1024x768" "1280x1024"
        ViewPort   0 0
        Virtual    320 200
        #Virtual   1280 1024
    EndSubsection
EndSection

# The 16-color VGA server

Section "Screen"
    Driver      "vga16"
    Device      "Generic VGA"
    Monitor     "My Monitor"
    Subsection  "Display"
        Modes     "640x480" "800x600"
        ViewPort   0 0
        Virtual    800 600
    EndSubsection
EndSection

# The Mono server

Section "Screen"
    Driver      "vga2"
    Device      "Generic VGA"
    Monitor     "My Monitor"
    Subsection  "Display"
        Modes     "640x480" "800x600"
        ViewPort   0 0
        Virtual    800 600
    EndSubsection
EndSection

```

```

# The accelerated servers (S3, Mach32, Mach8, 8514, P9000, AGX, W32, Mach64)

# The accelerated servers (S3, Mach32, Mach8, 8514, P9000, AGX, W32, Mach64)

Section "Screen"
    Driver      "accel"
    Device      "My Video Card"
    Monitor     "My Monitor"
    Subsection  "Display"
        Depth      8
        Modes      "640x480" "800x600" "1024x768" "1280x1024"
        ViewPort   0 0
        Virtual    1280 1024
    EndSubsection
    Subsection  "Display"
        Depth      16
        Modes      "1024x768" "640x480" "800x600"
        ViewPort   0 0
        Virtual    1024 768
    EndSubsection
    Subsection  "Display"
        Depth      32
        Modes      "640x480" "800x600"
        ViewPort   0 0
        Virtual    800 600
    EndSubsection
EndSection

```

### A videoüzemmódok beállítása

A konfigurálás legbonyolultabb és legveszélyesebb része a videoüzemmódok beállítása, mivel ez közvetlenül meghatározza a szinkronizáció frekvenciát (időzítést), és az olcsóbb monitoroknál, amelyek nem tartalmaznak védőáramköröket, a helytelen értékek károsodást okozhatnak. Ennek elkerülésére a monitor határértékeit az induláskor be kell írni a konfigurációs fájlba. (Ez nem jelenti azt, hogy a monitor szükségszerűen tönkretesszi, viszont helytelen értékek megadásával a monitor túlterhelhetők, amely huzamosabb idő után tönkretesz a monitor, szélsőséges esetben komoly balesetet okozhat. Ez az idő minimálisan 15-20 másodperc, tehát, ha látjuk, hogy a monitor képtelen beállni, éles, jó képet produkálni az általunk adott adatok alapján, aminek az ellenőrzése 1-2 másodperc alatt megvan, akkor azonnal kapcsoljunk vissza karakteres üzemmódra. Tehát gyorsan és óvatosan fogjunk ehhez a művelethez, mert megfelelő odafigyelés nélkül balesetet okozhatunk. A korszerű monitorokon, amelyeket VESA vagy Energy Star módszerrel ki lehet kapcsolni, azt fogjuk tapasztalni, hogy a monitor kikapcsol, ha rossz értékeket adunk meg. Az ilyen monitoroknál a károsodás kizárt. A régi monitoroknál előfordulhat az is, hogy magas hangon fútyulni, visítani kezd, ekkor gyorsan kapcsoljunk vissza karakteres üzemmódra – a lektor.)

A videoüzemmód ilyen megadásának az az előnye, hogy a monitor képességeit teljes mértékben ki lehet használni. Például egy 14"-es monitor, amelynek maximális vízszintes szinkronizáció frekvenciája (HSF) túl alacsony a 800x600 képpont felbontású vibrálásmentes megjelenítéshez, használható 800x550-es felbontásban 72 Hz-es képfürészési frekvenciával (RR).

A konfigurációs fájl minden üzemmódhoz megadja a használandó órajel-frekvenciát, valamint a vízszintes és a függőleges szinkronizálás 4-értékét. A megadás történhet egyetlen sorban (mint a ModeLine esetében) vagy több sorba elosztva. A következő példában a két definíció azonos jelentőségű:

```
# These two are equivalent
#
# ModeLine "1024x768i" 45 1024 1048 1208 1264 768 776 784 817 Interlace
#
# Mode "1024x768i"
#   DotClock      45
#   HTimings     1024 1048 1208 1264
#   VTimings     768 776 784 817
#   Flags        "Interlace"
# EndMode
```

Az üzemmód megadásának végén szerepelhet egy Flags sor is, amely az interlace, a +hsync, a +vsync és a csync értékeit tartalmazhatja. Ez az interlace módöt és a szinkronizálás típusát határozza meg.

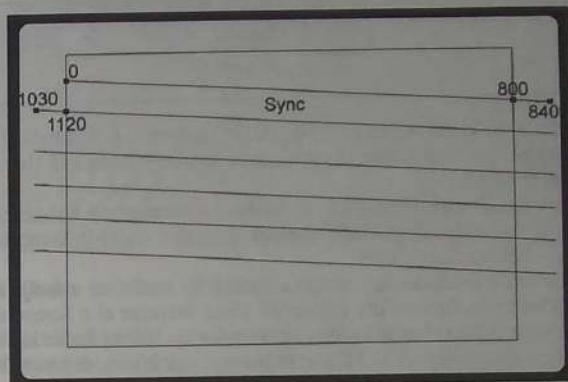
```
Mode "1024x768i"
  DotClock      45
  HTimings     1024 1048 1208 1264
  VTimings     768 776 784 817
  Flags        "Interlace"
EndMode
```

A vízszintes időzítés négyes csoportjainak értékei sorrendben a következők:

- a képpontok maximális száma (ennél nagyobb felbontással már nem jeleníthető meg kép);
- a képpont-órajelek száma a vízszintes szinkronizáló impulzus (sync) kezdeteig (a működés közben mérve);
- a képpont-órajelek száma addig, míg a vízszintes szinkronizáló impulzus lefut és az elektronsgár második védett ideje elkezdődik;
- a képpont-órajelek teljes száma egy ciklus (sor) végéig.

```
ModeLine "800x5" 45 800 840 1030 1120 540 540 546 558
```

A fenti példa olyan 800x550 képpont felbontású üzemmódot definiál, amelyhez a „800x5” nevet rendeljük.



8.4 ábra. A képernyő felépítésének vázlata

A vízszintes felbontás 800 képpont, és a látható sor vége után az elektronsugár számára biztonsági idő kezdődik. Ez az idő a 840-es képpontig tart, majd ezután kezdődik a szinkronizáló impulzus: ez 190 képpont-órajelen át tart, azaz az 1030-as képpontig. Ezután biztonsági idő következik a 1120-as képpontig, majd ekkor kezdődik a második vízszintes ciklus. 540 vízszintes ciklus (letapogatott sor) után a függőleges szinkronizálás következik, amely 6 vízszintes cikluson át tart. Ezután újabb biztonsági idő következik az 558. ciklusig. A biztonsági idő után a következő képernyő kezdődik.

Az /usr/lib/X11/doc könyvtárban található video.tutorial és VideoModes.doc fájl részletesen ismerteti a videoüzemmódok pontos értékeit meghatározására vonatkozó szabályokat.

Gyakran az a legegyszerűbb eljárás, hogy a rendelkezésre álló példafájlokban megkeressük a megfelelő értékeket, és módosítjuk azokat. Az `sc` egyszerű táblázatkezelő programhoz készítettek egy táblázatot, amely leegyszerűsíti a számításokat. Az ezt tartalmazó modegen.taz fájl a sunsite.unc.edu FTP kiszolgálón a /pub/Linux/X11/install könyvtárban található.

Az egyszerűbb monitoroknál a korlátozó tényező általában a vízszintes szinkronizálási frekvencia. Ez az a frekvencia, amellyel az elektronsugár balról jobbra letapogatja a sorokat. Ez a frekvencia az órajel (MHz-ben) és a vízszintes időzítési adatblokk legnagyobb (jobb szélén) értékének hármasaként számítható ki:

$$f_{\text{vízszintes}} = \frac{f_{\text{képpont}}}{N_{\text{képpont}}}$$

A fenti példában a kívánt érték 45 MHz/1120, amely közelítőleg 40 kHz. Példánkban ez az adat jelenti a monitor felső határértékét.

### A képfrissítési frekvencia

A függőleges szinkronizálási frekvenciát (függőleges időzítés vagy képfrissítési frekvencia) úgy számítjuk ki, hogy a vízszintes szinkronizálási frekvenciát elosztjuk a teljes képernyőhöz szükséges sorok (azaz a vízszintes ciklusok) számával. Ez a függőleges időzítési adatblokk jobb szélén száma.

$$f_{\text{függelget}} = \frac{f_{\text{vízszintes}}}{N_{\text{unk}}}$$

Példánkban ez az érték 40 kHz/558, azaz 72 Hz. Ha 540 helyett 600 sort kellene megjeleníteni, akkor a függelges szinkronizálási frekvencia lényegesen kevesebb lenne a 72 Hz-nél, ami a képernyő vibrálását eredményezné.

A jobb minőségű monitoroknál, amelyek vízszi szinkronizálási frekvenciája például 60 KHz, valamint az újabb videokártyákkal, amelyek nagyobb órajel-frekvenciával működnek, nagyobb képrisszeti frekvencia lehető el.

Meglévő videoüzemmóddal módosításához azt javasoljuk, hogy ismétlen másolja és módosítsa az üzemmódot, és minden egyes üzemmódot különböző néven helyezze el a Screen szakasz egy-egy ModeLine sorába. Ezután indítsa el az X kiszolgálót, a numerikus billentyűzetben kiadott <Ctrl-Alt-> és <Ctrl-Alt-> billentyűkombinációval váltsan az üzemmódok között, és hasonlítsa össze működésüket. Ha a monitor „kiesik a szinkronból”, azaz nem képes álló képet megjeleníteni, a monitor károsodásának elkerülése érdekében gyorsan állítsan be másik üzemmódot, vagy a <Ctrl-Alt-Backspace> billentyűkombinációval állítsa le az X kiszolgáló működését.

A kép beállításában a `vga` set program segíthet. Az `xterm` programból történő elindítása lehetővé teszi a képhelyzet interaktív módosítását. Egyetlen billentyűvel megváltoztatható a szegély mérete és módosítható a szinkronizáló jel időtartama. Az `Xconfig` fájlba bevitt aktuális beállításhoz tartozó nyolc értéket folyamatosan megjeleníti a program.

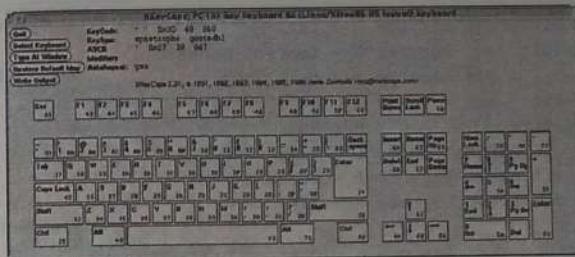
### A billentyűzet konfigurálása

Az X Window System a rendszermagtól függetlenül kezeli a billentyűzetet. Alapértelmezésként amerikai billentyűkiosztást állít be. Az egyes nemzeti billentyűzetek táblázatai az `xmodmap` segédprogrammal töltelhetők be. A szokásos konfigurálással elindított X kiszolgáló az `xmodmap` programot az `.Xmodmap` fájllal indítja el. A program ezt a fájlt először a felhasználó saját könyvtárában, majd az `/usr/lib/X11/xinit` könyvtárban keresi.

Ha egy adott billentyűzet-elrendezést a teljes rendszeren módosítani kell, akkor azt az `/usr/lib/X11/xinit` könyvtárban kell végrehajtani. A szoftver néhány kiadásában kész `.Xmodmap` fájlok találhatók, vagy ezek az FTP kiszolgálókról töltelhetők le, például a `sunsite.unc.edu /pub/Linux/X11/misc` könyvtárból.

Mivel az `xmodmap` hívása általában egy `xinitrc` szkriptből történik, ez a hívás módosítható úgy, hogy az `.Xmodmap` fájlt másik könyvtárban vagy más néven keresse. Probléma esetén érdemes megnézni az X kiszolgálót indító szkriptet, amely általában a `startx` (OpenWindows esetén `openwin`).

Az `xkeycaps` segédprogram még kényelmesebb beállítást tesz lehetővé. Ez megtalálható a legtöbb Linux kiadásban, vagy a szokásos FTP kiszolgálókról is letölthető. A program egy X11 kiosztást, majd azt az egérrel egyszerűen módosíthatja.



8.5 ábra. Az xkeycaps

Az amerikai Linux változatokban általában nem szerepelnek nemzeti billentyűzet-elrendezések. Így példaként itt egy német .Xmodmap fájl szerepel a nemzeti billentyűk definiálásának bemutatására. Érdemes megfigyelni a 12-es számú szimbólumot, amelyet gyakran helytelenül a section szimbólum helyett a paragraph jelre definiálnak.

```

charset "iso-8859-2"
keymaps 0-15

keycode 1 = Escape           Escape
          altgr keycode 1 = Escape
          alt   keycode 1 = Meta_Escape

keycode 2 = one              exclam
          altgr keycode 2 = one
          shift altgr keycode 2 = exclam
          alt altgr keycode 2 = asciitilde
          alt   keycode 2 = Meta_one
shiftalt keycode 2 = Meta_exclam

keycode 3 = two              at
          altgr keycode 3 = two
          shift altgr keycode 3 = quotedbl
          alt altgr keycode 3 = 183

! mi a neve?
control   keycode 3 = nul
shift control keycode 3 = nul
          alt   keycode 3 = Meta_two
          shift alt keycode 3 = Meta_at

keycode 4 = three             numbersign
          altgr keycode 4 = three
          shift altgr keycode 4 = plus
          alt altgr keycode 4 = asciicircum
control   keycode 4 = Escape
          alt   keycode 4 = Meta_three
          shift alt keycode 4 = Meta_numbersign

keycode 5 = four              dollar
          altgr keycode 5 = four
          shift altgr keycode 5 = exclam

```

```

        alt altgr keycode 5 = 162
control      keycode 5 = Control_backslash
shift       alt keycode 5 = Meta_four
keycode 6 = five    percent
                alt keycode 6 = five
shift       alt altgr keycode 6 = percent
                alt altgr keycode 6 = 176

! mi a neve?
control      keycode 6 = Control_bracketright
shift       alt keycode 6 = Meta_five
keycode 7 = six    asciicircum
                alt altgr keycode 7 = six
shift       alt altgr keycode 7 = slash
                alt altgr keycode 7 = 131

! mi a neve?
control      keycode 7 = Control_asciicircum
shift       alt keycode 7 = Meta_six
keycode 8 = seven   ampersand      braceleft
                alt altgr keycode 8 = seven
shift       alt altgr keycode 8 = equal
                alt altgr keycode 8 = grave
control      keycode 8 = Control_underscore
                alt keycode 8 = Meta_seven
keycode 9 = eight   asterisk      bracketleft
                alt altgr keycode 9 = eight
shift       alt altgr keycode 9 = parenleft
                alt altgr keycode 9 = period

! ide biztos, hogy ez kell?
control      keycode 9 = Delete
                alt keycode 9 = Meta_eight
keycode 10 = nine   parenleft      bracketright
                alt altgr keycode 10 = nine
shift       alt altgr keycode 10 = parenright
                alt altgr keycode 10 = apostrophe
                alt keycode 10 = Meta_nine
keycode 11 = zero   parenright      braceright
                alt altgr keycode 11 = +odiaeresis
shift       alt altgr keycode 11 = +odiaeresis
                alt altgr keycode 11 = quotedbl
                alt keycode 11 = Meta_zero
keycode 12 = minus   underscore      backslash
control      keycode 12 = Control_underscore
shift       control keycode 12 = Control_underscore
                alt altgr keycode 12 = +odiaeresis
shift       alt altgr keycode 12 = +odiaeresis
                alt altgr keycode 12 = 168

```

```
!! mi a neve?

keycode 13 = equal          plus
            alt keycode 13 = Meta_equal
            altgr keycode 13 = +oacute
            shift altgr keycode 13 = +Oacute
            alt altgr keycode 13 = 13l

keycode 14 = Delete         Delete
            altgr keycode 14 = Delete
            shift altgr keycode 14 = Delete
            alt altgr keycode 14 = BackSpace
control      keycode 14 = BackSpace
            alt keycode 14 = Meta_Delete

keycode 15 = Tab            Tab
            altgr keycode 15 = Tab
            shift altgr keycode 15 = Tab
            alt keycode 15 = Meta_Tab

keycode 16 = q              alt altgr keycode 16 = backslash

keycode 17 = w              alt altgr keycode 17 = bar

keycode 18 = e
keycode 19 = r
keycode 20 = t
keycode 21 = y
!keycode 21 = z
keycode 22 = u
keycode 23 = i
keycode 24 = o
keycode 25 = p
keycode 26 = bracketleft    braceleft
            altgr keycode 26 = +odoubleacute
            shift altgr keycode 26 = +Odoubleacute
            alt altgr keycode 26 = division
control      keycode 26 = Escape
            alt keycode 26 = Meta_bracketleft
            shift alt keycode 26 = Meta_braceleft

keycode 27 = bracketright   braceright      asciitilde
            altgr keycode 27 = +uacute
            shift altgr keycode 27 = +Uacute
            alt altgr keycode 27 = X

!ide a "szo:gletes X" kell
control      keycode 27 = Control_bracketright
            alt keycode 27 = Meta_bracketright
            shift alt keycode 27 = Meta_braceright

keycode 28 = Return         alt keycode 28 = Meta_Control_m

keycode 29 = Control
```

```

keycode 30 = a
keycode 31 = s
keycode 32 = d
keycode 33 = f
          alt altgr keycode 33 = bracketleft
keycode 34 = g
          alt altgr keycode 34 = bracketright
keycode 35 = h
keycode 36 = j
keycode 37 = k
keycode 38 = l
keycode 39 = semicolon      colon
          altgr keycode 39 = +eacute
          shift altgr keycode 39 = +Eacute
          alt altgr keycode 39 = dollar
          alt keycode 39 = Meta_semicolon
keycode 40 = apostrophe     quotedbl
          altgr keycode 40 = +aacute
          shift altgr keycode 40 = +Aacute
control      keycode 40 = Control_G
          alt keycode 40 = Meta_apostrophe
keycode 41 = grave         asciitilde
          altgr keycode 41 = zero
! shift altgr keycode 41 = paragraph
          altgr keycode 41 = +iacute
          shift altgr keycode 41 = +Iacute
control      keycode 41 = nul
          alt keycode 41 = Meta_grave
keycode 42 = Shift
keycode 43 = backslash      bar
          altgr keycode 43 = +udoubleacute
          shift altgr keycode 43 = +Udoubleacute
          alt altgr keycode 43 = 164
"tastatur"
control      keycode 43 = Control_backslash
          alt keycode 43 = Meta_backslash
shift      alt keycode 43 = Meta_bar
keycode 44 = z
          alt altgr keycode 44 = greater
keycode 44 = y
keycode 45 = x
          alt altgr keycode 45 = numbersign
keycode 46 = c
          alt altgr keycode 46 = ampersand
keycode 47 = v
          alt altgr keycode 47 = at
keycode 48 = b
          alt altgr keycode 48 = braceleft
keycode 49 = n

```

```

        alt altgr keycode 49 = braceleft
keycode 50 = m
keycode 51 = comma           less
            altgr keycode 51 = comma
            shift altgr keycode 51 = question
            alt altgr keycode 51 = semicolon
            alt keycode 51 = Meta_comma
            shift alt keycode 51 = Meta_less
keycode 52 = period          greater
            altgr keycode 52 = period
            shift altgr keycode 52 = colon
            control    keycode 52 = Compose
            alt keycode 52 = Meta_period
            shift alt keycode 52 = Meta_greater
keycode 53 = slash            question
            altgr keycode 53 = minus
            shift altgr keycode 53 = underscore
            alt altgr keycode 53 = asterisk
            control    keycode 53 = Delete
            shift control keycode 53 = Delete
            alt keycode 53 = Meta_slash
keycode 54 = Shift
keycode 55 = KP_Multiply
keycode 56 = Alt
keycode 57 = space            space
            altgr keycode 57 = space
            shift altgr keycode 57 = space
            control    keycode 57 = nul
            alt keycode 57 = Meta_space
keycode 58 = Caps_Lock
keycode 59 = F1                F13           Console_13
            control    keycode 59 = F25
            shift control keycode 59 = F37
            alt keycode 59 = Console_1
            alt altgr keycode 59 = Console_1
            control    alt keycode 59 = Console_1
keycode 60 = F2                F14           Console_14
            control    keycode 60 = F26
            shift control keycode 60 = F38
            alt keycode 60 = Console_2
            alt altgr keycode 60 = Console_2
            control    alt keycode 60 = Console_2
keycode 61 = F3                F15           Console_15
            control    keycode 61 = F27
            shift control keycode 61 = F39
            alt keycode 61 = Console_3
            alt altgr keycode 61 = Console_3
            control    alt keycode 61 = Console_3

```

```

keycode 62 = F4          keycode 62 = F28          F16
control      shift control keycode 62 = F40
shift control keycode 62 = Console_4
alt keycode 62 = Console_4
alt altgr keycode 62 = Console_4
control      alt keycode 62 = Console_4
control      F17
keycode 63 = F5          keycode 63 = F29          Console_16
control      shift control keycode 63 = F41
shift control keycode 63 = Console_5
alt keycode 63 = Console_5
alt altgr keycode 63 = Console_5
control      alt keycode 63 = Console_5
keycode 64 = F6          keycode 64 = F30          F18
control      shift control keycode 64 = F42
shift control keycode 64 = Console_6
alt keycode 64 = Console_6
alt altgr keycode 64 = Console_6
control      alt keycode 64 = Console_6
keycode 65 = F7          keycode 65 = F31          F19
control      shift control keycode 65 = F43
shift control keycode 65 = Console_7
alt keycode 65 = Console_7
alt altgr keycode 65 = Console_7
control      alt keycode 65 = Console_7
keycode 66 = F8          keycode 66 = F32          F20
control      shift control keycode 66 = F44
shift control keycode 66 = Console_8
alt keycode 66 = Console_8
alt altgr keycode 66 = Console_8
control      alt keycode 66 = Console_8
keycode 67 = F9          keycode 67 = F33          F21
control      shift control keycode 67 = F45
shift control keycode 67 = Console_9
alt keycode 67 = Console_9
alt altgr keycode 67 = Console_9
control      alt keycode 67 = Console_9
keycode 68 = F10         keycode 68 = F34          F22
control      shift control keycode 68 = F46
shift control keycode 68 = Console_10
alt keycode 68 = Console_10
alt altgr keycode 68 = Console_10
control      alt keycode 68 = Console_10
keycode 69 = Num_Lock    keycode 69 = AltGr_Lock
control      altgr control keycode 69 = AltGr_Lock
keycode 70 = Scroll_Lock Show_Memory      Show_Registers
control      keycode 70 = Show_State
alt keycode 70 = Scroll_Lock
keycode 71 = KP_7

```

```
        alt keycode 71 = Ascii_7
keycode 72 = KP_8          alt keycode 72 = Ascii_8
keycode 73 = KP_9          alt keycode 73 = Ascii_9
keycode 74 = KP_Subtract
keycode 75 = KP_4          alt keycode 75 = Ascii_4
keycode 76 = KP_5          alt keycode 76 = Ascii_5
keycode 77 = KP_6          alt keycode 77 = Ascii_6
keycode 78 = KP_Add
keycode 79 = KP_1          alt keycode 79 = Ascii_1
keycode 80 = KP_2          alt keycode 80 = Ascii_2
keycode 81 = KP_3          alt keycode 81 = Ascii_3
keycode 82 = KP_0          alt keycode 82 = Ascii_0
keycode 83 = KP_Period
    altgr control keycode 83 = Boot
    control alt keycode 83 = Boot
keycode 84 = Last_Console
keycode 85 =
keycode 86 = less           greater           bar
    ! altgr keycode 86 = +iacute
    ! shift altgr keycode 86 = +Iacute
    alt altgr keycode 86 = less
    alt keycode 86 = Meta_less
keycode 87 = F11            F23                Console_23
    control keycode 87 = F35
    shift control keycode 87 = F47
        alt keycode 87 = Console_11
    control alt keycode 87 = Console_11
keycode 88 = F12            F24                Console_24
    control keycode 88 = F36
    shift control keycode 88 = F48
        alt keycode 88 = Console_12
    control alt keycode 88 = Console_12
keycode 89 =
keycode 90 =
keycode 91 =
keycode 92 =
keycode 93 =
keycode 94 =
keycode 95 =
keycode 96 = KP_Enter
```

```

keycode 97 = Control
keycode 98 = KP_Divide
keycode 99 = Control_backslash
controlkeycode 99 = Control_backslash
altkeycode 99 = Control_backslash
keycode 100 = AltGr
keycode 101 = Break
keycode 102 = Find
keycode 103 = Up
keycode 104 = Prior
    shift     keycode 104 = Scroll_Backward
keycode 105 = Left
    alt keycode 105 = Decr_Console
keycode 106 = Right
    alt keycode 106 = Incr_Console
keycode 107 = Select
keycode 108 = Down
keycode 109 = Next
    shift     keycode 109 = Scroll_Forward
keycode 110 = Insert
keycode 111 = Remove
    altgr control keycode 111 = Boot
    control   alt keycode 111 = Boot
keycode 112 = Macro
keycode 113 = F13
keycode 114 = F14
keycode 115 = Help
keycode 116 = Do
keycode 117 = F17
keycode 118 = KP_MinPlus
keycode 119 = Pause
keycode 120 =
keycode 121 =
keycode 122 =
keycode 123 =
keycode 124 =
keycode 125 =
keycode 126 =
keycode 127 =
string F1 = "33|[A"
string F2 = "33|[B"
string F3 = "33|[C"
string F4 = "33|[D"
string F5 = "33|[E"
string F6 = "33|[7~"
string F7 = "33|[8~"
string F8 = "33|[9~"
string F9 = "33|[20~"
string F10 = "33|[21~"

```

```
string F11 = "33[23~"
string F12 = "33[24~"
string F13 = "33[25~"
string F14 = "33[26~"
string F15 = "33[28~"
string F16 = "33[29~"
string F17 = "33[31~"
string F18 = "33[32~"
string F19 = "33[33~"
string F20 = "33[34~"
string Find = "33[1~"
string Insert = "33[2~"
string Remove = "33[3~"
string Select = "33[4~"
string Prior = "33[5~"
string Next = "33[6~"
string Macro = "33[M"
string Pause = "33[B"
compose "' 'A' to 'R'
compose "' 'a' to 'f'
compose '_A' to 'Á'
compose '_a' to 'á'
compose '^ 'A' to 'A'
compose '^ 'a' to 'a'
compose '~ 'A' to 'Á'
compose '~ 'a' to 'á'
compose "' 'A' to 'A'
compose "' 'a' to 'a'
compose 'o' 'A' to 'í'
compose 'A' 'A' to 'í'
compose 'a' 'a' to 'í'
compose 'A' 'E' to 'é'
compose 'a' 'e' to 'é'
compose ', 'C' to 'ç'
compose ', 'c' to 'ç'
compose "' 'E' to 'ç'
compose "' 'e' to 'ç'
compose '_E' to 'É'
compose '_e' to 'é'
compose '^ 'E' to 'É'
compose '^ 'e' to 'é'
compose "' 'E' to 'É'
compose "' 'e' to 'é'
compose "' 'I' to 'É'
compose "' 'i' to 'é'
compose '_I' to 'í'
```

compose 'i' to 'i'  
compose 'I' to 'I'  
compose 'i' to 'i'  
compose 'I' to 'I'  
compose 'i' to 'd'  
compose 'I' to 'D'  
compose 'd' to 'd'  
compose 'N' to 'N'  
compose 'n' to 'n'  
compose 'o' to 'N'  
compose 'O' to 'N'  
compose 'o' to 'O'  
compose 'u' to 'U'  
compose 'Y' to 'Y'  
compose 'y' to 'y'  
compose 'H' to 'H'  
compose 'h' to 'h'  
compose 's' to 'B'  
compose 's' to 'B'  
compose 's' to 'B'  
compose 'i' to 'j'

Példa egy német billentyűkiosztásra az X11 alatt

#### **8.9 AZ X ALKALMAZÁSOK KONFIGURÁÍÁSA**

A legtöbb X ügyfélhez egy olyan fájl is tartozik, amely az alkalmazások alapértelmezéseit tartal-  
mazza, és ez az X11 rendszerterületre (/usr/lib/X11/app-defaults) töltődik be.  
Ebben a fájlból az alkalmazás legfontosabb alapbeállításai találhatók, így például a grafikus objek-  
tumok mérete, elhelyezkedése és színe, valamint a megfelelő nyelven a hibautzenetek.

Minden alkalmazáshoz a készítői egy osztálynevet rendeltek, amely a forrásfájl nevének felel meg. Az osztálynevek minden nagybetűvel kezdődnek. Például az `xterm` háttérszinét (osztálynév `XTerm`) az `/usr/lib/X11/app-defaults/XTerm` fájlból kell módosítani.

Különböző környezeti változók (`XFILESEARCHPATH`, `XAP-PLERESDIR`) az erőforrás-fájlok keresési útvonalát állítják be. Az `XFILESEARCHPATH` több, egymástól visszszővel elválasztott keresési útvonal megadását teszi lehetővé, és a következő különleges karaktereket kezeli:

<code>%C</code>	az erőforrás értéke (*.customization)
<code>%L</code>	nyelv, helyi kódkészlet
<code>%l</code>	nyelv
<code>%N</code>	osztálynév
<code>%T</code>	fájtípus (app-defaults)

E környezeti változó egy lehetséges beállítása a következő:

```
XFILESEARCHPATH=/usr/X11R6/lib/X11/%T:/usr/local/%T/%N:$HOME/%T/%N
```

Ekkor az erőforrásfájlokat a következő három könyvtárban keresi a rendszer:

1. `/usr/lib/X11/app-defaults/<osztály>`
2. `/usr/local/app-defaults/<osztály>`
3. Home könyvtár/app-defaults/<osztály>

Az X11 alkalmazások konfigurálásnak egy másik módja az `xrdb` parancs, amely az átadott erőforrásfájlt az X kiszolgáló valamelyik tulajdonságába (RESOURCE\_MANAGER vagy SCREEN\_RESOURCES) tölti. A tulajdonság az X kiszolgáló olyan általános memóriatartománya, amelyhez név rendelhető. Egy X alkalmazás elindításakor az erőforrás-kezelő kiértékeli az ezekben a tulajdonságokban lévő erőforrás-definíciókat. Az alkalmazások konfigurálása az erőforrások tulajdonságaival akkor különösen hasznos, ha az alkalmazások indítása más gépen történik, és a helyi megjelenési módot befolyásolni szeretnénk. Az `xrdb` számos kapcsolóval indítható:

<code>-all</code>	a műveletek minden tulajdonságra vonatkoznak
<code>-screen</code>	a műveletek csak a SCREEN_RESOURCES tulajdonságra vonatkoznak
<code>-global</code>	a műveletek csak a RESOURCE_MANAGER tulajdonságra vonatkoznak
<code>query</code>	egy tulajdonság aktuális tartalmát jeleníti meg
<code>merge &lt;fájl&gt;</code>	egy fájl tartalmát egyesíti egy tulajdonsággal
<code>edit &lt;fájl&gt;</code>	egy tulajdonság tartalmát fájlba menti
<code>remove</code>	eltávolít egy teljes tulajdonságot
<code>load &lt;fájl&gt;</code>	egy tulajdonságot egy fájl tartalmával felülír

A felhasználók egyéni `.Xdefaults` fájlokat is készíthetnek a saját könyvtárukban, és így helyettesíthetik a globális alapértelmezett beállításokat.

A következő lista áttekintést ad azokról a fájlokról és elérési útvonalakról, amelyeket a rendszer az aktuális elem attribútumainak meghatározására egy alkalmazás elindításakor egymás után feldolgoz. Ha egy erőforráshoz egnél több helyen történt érték hozzárendelése, akkor csak az utolsóként definiált érték lesz érvényes.

Alkalmazáson belül:

1. Visszalépési erőforrások (Fallback erőforrás: olyan erőforrás, amely kisebb igényt támászt az eredeti erőforráshoz képest, így ha valamilyen feltétel hiányzik egy erőforrás használatához, akkor ezt az erőforrást használja. Pl. egy grafikához 16 bit színmélységre lenne szükség, de az X megjelenítő nem támogatja, akkor itt megoldást adunk 8 bit színmélységen történő megjelenítésre – a lektor.)

- Alkalmazásra vonatkozó adatok:
  1. /usr/lib/SNYELV/app-defaults/<Osztály>
  2. /usr/lib/X11/app-defaults/<Osztály>
- Új keresési útvonal:
  1. \$XFILESEARCHPATH
- Felhasználóra vonatkozó adatok:
  1. XUSERFILESEARCHPATH
  2. \$XAPPLRESDIR/\$LANG/<Osztály>
  3. \$XAPPLRESDIR/<Osztály>
  4. \$HOME/\$LANG/Osztály>
  5. \$HOME/<Osztály>
- Képernyőre vonatkozó adatok:
  1. SCREEN\_RESOURCES tulajdonság (xrdb)
- Megjelenítésre vonatkozó adatok:
  1. RESOURCE\_MANAGER tulajdonság (xrdb)
  2. \$HOME/.Xdefaults fájl
- Gazdaszámítógépre vonatkozó adatok:
  1. \$ENVIRONMENT változók
  2. \$HOME/.Xdefaults-<gazdanév>
- Parancsor:
  1. Parancssori kapcsolók

### Vezérőelemek attribútumai

Ezek az erőforrásértékek ASCII formátumban kerülnek megadásra. Egy erőforrásfájlban a megkülönböztetésükre a programozó minden egyes alkalmazáshoz egy nevet (osztályt) rendel, amely csak ritkán egyszer meg a programfájl nevével. Hasonlóképpen minden egyes kívülről konfigurálható vezérőelem és elemattribútum névvel rendelkezik és egy osztályhoz tartozik. Egy vezérőelem egyedi azonosításához nem elelegendő a név. Ehelyett, a fájlrendszerekhez hasonlóan, az elérési útvonalat kell itt is megadni, amely a vezérőelem hierarchiájának felel meg.

Több vezérőelem attribútumainak egyidejű kezeléséhez az elérési útvonalban megengedett a helyettesítő karakterek (?,\* ) használata. Az erőforrás megadásának pontos szintaxisa a következő:

objektum.alobjektum[.alobjektum...].attribútum: érték

Az egyes elemek jelentése a következő:

objektum	a program osztálya vagy neve
alobjektum	a vezérlőelem osztálya vagy neve
attribútum	az erőforrás neve
érték	érték
.	elválasztó karakter
*	helyettesítő karakter (tetszőleges számú név)
?	helyettesítő karakter (egyetlen név)

Az erőforrásfájl első oszlopa az attribútumot adja meg. Ez általában megfelel a vezérlőelem-erőforrásnak, de a programozó új, az alkalmazáshoz tartozó erőforrásokat is definiálhat. A rendelkezésre álló erőforrások hierarchiája és neve a megfelelő kézikönyvoldalon (Manual) tekinthető meg.

```
Xterm*background:      gray90
Xterm*ScrollBar:       true
Xterm*ForeGround:      white
Xterm*Background:      gray20
Xterm*IconName:        XTerm
Xterm*WaitForMap:       true
Xterm*MarginBell:       false
Xterm*JumpScroll:       true
```

#### Részlet egy erőforrásfájlból

Hasonlóképpen az egyedi vezérlőelemek attribútumai is osztályokba gyűjthetők. Osztályazonosítók használatával az erőforrásfájlok rövidebbé és áttekinthetőbbé tehetők. Például az `xterm` `cursorColor` és `pointerColor` attribútuma a `Foreground` osztályba tartozik. Éppen ezért az

```
XTerm*foreground:      green
XTerm*cursorColor:     green
XTerm*pointerColor:    green
```

a következőképpen rövidíthető:

```
XTerm*Foreground:      green
```

Az X Window System 5-ös és 6-os kiadása tartalmaz egy interaktív erőforrás-kezelőt is (`editres`), amely egy futó program összes erőforrásértéknek kényelmes kezelését teszi lehetővé, valamint ezek az értékek ASCII fájlba is menthetők. Lényeges, hogy ezt a futtatás során kell végrehajtani, mert így a felhasználó közvetlenül megnézheti a módosítások hatásait. Sajnos az `editres` protokoll nem minden elemgyűjteménnyel használható, így alkalmazása korlátozott. A létrehozott

ASCII fájlok egyszerűen integrálhatók a meglévő erőforrásfájlokba, vagy hozzáfűzhetők az .Xdefaults fájlhoz.

### Az ablakkezelő konfigurálása

A felhasználó nemcsak az egyes alkalmazások megjelenését konfigurálhatja, hanem a legtöbb X ablakkezelőjét is. Mivel számos Linux felhasználó dolgozik az fvwm ablakkezelővel, ezt ismertetjük részletesen. Az ablakkezelő paramétereit az /usr/lib/X11/fvwm könyvtárban lévő system.fvwmrc fájl tartalmazza. A felhasználók saját könyvtárukban is létrehozhatnak .fvwmrc fájlt.

Az M4 makrófeldolgozó használata további rugalmasságot biztosít: például lehetővé teszi további konfigurációs fájlok csatolását vagy a beállítások ellenőrzését. A központi fájl (system.fvwmrc) így áttekinthetővé tehető.

```
#####
# system.fvwmrc - fvwm configuration
#
#####
# Paths

ModulePath  /usr/lib/X11/fvwm/modules
PixmapPath  /usr/lib/X11/pixmaps:/usr/local/lib/pixmaps
IconPath    /usr/include/X11/bitmaps

#####
# External configuration files

include(/usr/lib/X11/fvwm/fvwm.options)

include(/usr/lib/X11/fvwm/fvwm.menus)

include(/usr/lib/X11/fvwm/fvwm.functions)

include(/usr/lib/X11/fvwm/fvwm.bindings)

include(/usr/lib/X11/fvwm/fvwm.styles)

include(/usr/lib/X11/fvwm/fvwm.goodstuff)

include(/usr/lib/X11/fvwm/fvwm.modules)

#####
# Initialization and restart function

Function "InitFunction"
  Module   "I"      GoodStuff
  Module   "I"      FvwmPager 0 1
```

```

Exec      "I"      exec xterm -sb -sl 400 -geometry +75+390 &
Exec      "I"      xsetroot -solid LightSlateGray
EndFunction

Function "RestartFunction"
  Module    "I"      GoodStuff
  Exec      "I"      xsetroot -solid LightSlateGray
  Module    "I"      FvwmPager 0 1
EndFunction

```

## A system.fvwmrc fájl

A szín és a betűkészlet beállítása mellett az fvwm.options fájl számos egyéb, a megjelenést beállító paramétert is tartalmazhat.

```

#####
#
# fvwm.options - general options
#
# 

DeskTopSize 2x2
DeskTopScale 32

# Standard colors
StdForeColor     Black
StdBackColor     #d3d3d3

# Window colors
HiForeColor     Black
HiBackColor     #5f9ea0
StickyForeColor Black
StickyBackColor #60c0a0

# Menu colors
MenuForeColor    Black
MenuBackColor   grey
MenuStippleColor SlateGrey

# Fonts
Font           -adobe-helvetica-medium-r----12-----*
WindowFont     -adobe-helvetica-bold-r----12-----*
IconFont       fixed

# Rectangles in which icons are positioned
IconBox 5 -80 -140 -5
IconBox 5 -160 -140 -85
IconBox 5 -240 -140 -165
IconBox 5 -320 -140 -245

```

```

# Motif look and feel
MMFunctionHints
MWMHintOverride
MMDecorHints
MMBorders
MWNButtons

# Moves all windows with contents
OpaqueMove 100

# Suppress automatic desktop change
EdgeScroll 0 0

# Delay in changing desktop section
EdgeResistance 250 50

NoPPosition

# Automatic positioning of new window
RandomPlacement

# Forces decoration in transient shell
DecorateTransients

```

#### Az fvwm.options fájl

A felhasználó új menüket is létrehozhat, és ezekhez műveleteket rendelhet.

```

#####
#
# fvwm.menus - Menu configuration
#
#
Popup "Shells"
    Title      "Shells"
    Exec       "Mxterm"           exec mxterm &
    Exec       "Color Xterm"     exec color_xterm &
    Exec       "Rxvt"             exec rxvt &
EndPopup

Popup "Editors"
    Title      "Editors"
    Exec       "GNU emacs"        exec emacs &
    Exec       "NEdit"             exec nedit &
    Exec       "Textedit"          exec textedit &
EndPopup

```

```

Popup "Graphics"
    Title      "Graphics / Viewer"
    Exec       "XPaint"           exec xpaint &
    Exec       "XV"               exec xv &
EndPopup

Popup "Modules"
    Title      "Modules"
    Module    "GoodStuff"        GoodStuff
    Module    "Identify"         FvwmIdent
    Module    "SaveDesktop"      FvwmSave
    Module    "Pager"            FvwmPager 0 1
    Module    "FvwmWinList"      FvwmWinList
    Module    "FvwmIconBox"      FvwmIconBox
EndPopup

Popup "Window Ops"
    Title      "Window Ops      "
    Move       "&Move Alt+F7"
    Resize    "&Size Alt+F8"
    Lower     "&Lower Alt+F3"
    Raise     "Raise"
    Stick     "(Un) Stick      "
    Iconify   "(Un) Mi&nimize Alt+F9"
    Maximize  "(Un) Ma&ximize Alt+F10"
    Maximize  "(Un) Maximize Vertical 0 100
    Nop      " "
    Destroy   "&Kill Alt+F4"
    Delete    "Delete"
EndPopup

Popup "Window Ops2"
    Move       "&Move Alt+F7"
    Resize   "&Size Alt+F8"
    Iconify  "(Un) Mi&nimize Alt+F9"
    Maximize  "(Un) Ma&ximize Alt+F10"
    Lower    "&Lower Alt+F3"
    Nop      " "
    Destroy   "&Kill Alt+F4"
    Delete    "Delete"
    Nop      " "
    Module    "ScrollBar"        FvwmScroll 2 2
EndPopup

#####
#
# Mainmenu

Popup "Programs"

```

```

Title      "Programs"          exec xterm -sb -sl 400
Exec      "Xterm"
Popup     "Shells"            Shells
Popup     "Editors"           Editors
Popup     "Graphics"          Graphics
Popup     "Modules"           Modules
Exec      "Screen Lock"       exec xlock &
" "
Nop
Restart   "Restart Fvwm"     fvwm
Quit      "Exit"
EndPopup

```

Az fvwm.menus fájl

Az fvwm konfigurációs fájlon belül új függvények definíálhatók, amelyeket rendszerint billentyű-kombinációhoz vagy egérmóuvelethez rendelik.

```

#####
#
# fvwm.functions - function definition
#
Function "Move-or-Raise"
    Move      "Motion"
    Raise    "Motion"
    Raise    "Click"
    RaiseLower "DoubleClick"
EndFunction

Function "maximize_func"
    Maximize  "Motion"  0 100
    Maximize  "Click"   0 80
    Maximize  "DoubleClick" 100 100
EndFunction

Function "window_ops_func"
    PopUp    "Click"  Window Ops2
    PopUp    "Motion" Window Ops2
EndFunction

Function "Move-or-Lower"
    Move      "Motion"
    Lower    "Motion"
    Lower    "Click"
    RaiseLower "DoubleClick"
EndFunction

```

```

Function "Move-or-Iconify"
    Move      "Motion"
    Iconify   "DoubleClick"
EndFunction

Function "Resize-or-Raise"
    Resize    "Motion"
    Raise     "Motion"
    Raise     "Click"
    RaiseLower "DoubleClick"
EndFunction

```

## Az fvwm.functions fájl

Az fvwm.bindings fájl az egérhez és a billentyűzethez rendelt műveleteket tartalmazza.

```

#####
#
# fvwm.bindings - keyboard- and mouse configuration
#
#
# Structure of a configuration line:
#
#       <key>           <context> <modifier>  <function>
#
#       <key>           (mouse) key
#       <context>        R - root window
#                           W - application window
#                           T - title bar
#                           S - window sides
#                           F - window frame
#                           I - icon
#                           A - everything but title bar
#                           0,1,2,... - window element
#       <modifier>        N - no modifier key
#                           A - alternate
#                           C - control
#                           M - meta
#                           S - shift
#                           mod1-mod5 - X11 modifiers
#       <function>        fvwm function
#
#
# mouse click on root window
Mouse 1      R      A      PopUp "Programs"
Mouse 2      R      A      PopUp "Window Ops"
Mouse 3      R      A      Module "FvwmWinList" FvwmWinList
Transient

```

```

# window element          Function "window_ops_func"
Mouse 0      1      A      Function "maximize_func"
Mouse 0      2      A      Iconify
Mouse 0      4      A      Function "Resize-or-Raise"
Mouse 1      F      A      Function "Move-or-Raise"
Mouse 1      TS     A

# icon actions           Function "Move-or-Iconify"
Mouse 1      I      A      Iconify
Mouse 2      I      A

# window operations      Function "window_ops_func"
Mouse 2      FST    A      RaiseLower
Mouse 3      TSIF   A

# keyboard shortcut
Key F1       A      M      Popup "Window Ops"
Key F2       A      M      Popup "Programs"
Key F3       A      M      Lower
Key F4       A      M      Destroy
Key F5       A      M      CirculateUp
Key F6       A      M      CirculateDown
Key F7       A      M      Move
Key F8       A      M      Resize
Key F9       A      M      Iconify
Key F10      A      M      Maximize

```

Az fvwm.bindings fájl

Az fvwm.styles fájl az egyes alkalmazások kinézetét definiálja.

```

#####
#
# fvwm.styles - Style configuration
#
Style "*"           BorderWidth 7, HandleWidth 5
Style "FvwmPager"  Sticky, NoTitle
Style "FvwmBanner" StaysOnTop
Style "GoodStuff"  Sticky, WindowListSkip, NoTitle
Style "xterm"       Icon terminal.xpm
Style "xcalc"      Icon rcalc.xpm
Style "xman"        Icon xman.xpm
Style "xvgr"        Icon graphs.xpm
Style "Mail"        Icon sndmail.xpm
Style "emacs"       Icon editor2.xpm

```

Az fvwm.styles fájl

Minden egyes fvwm modul saját konfigurálási lehetőséggel rendelkezik. Ezeket az `fvwm.modules` fájl tartalmazza.

```

#####
# fvwm.modules - Module configuration
#
##### Window identifier #####
*FvwmIdentBack MidnightBlue
*FvwmIdentFore Yellow
*FvwmIdentFont -adobe-helvetica-medium-r-**-12-**-**-**-**-**
#####
# FvwmWinList #####
*FvwmWinListBack #d3d3d3
*FvwmWinListFore Black
*FvwmWinListFont -adobe-helvetica-bold-r-**-10-**-**-**-**-**
*FvwmWinListAction Click1 Iconify -1,Focus
*FvwmWinListAction Click2 Iconify
*FvwmWinListAction Click3 Module "FvwmIdent" FvwmIdent
*FvwmWinListUseSkipList
*FvwmWinListGeometry +0-1

#####
# FvwmIconBox #####
*FvwmIconBoxIconBack #cfcfcf
*FvwmIconBoxIconHiFore black
*FvwmIconBoxIconHiBack #5f9ea0
*FvwmIconBoxBack #cfcfcf
*FvwmIconBoxFore blue
*FvwmIconBoxGeometry 1x5+0+89
*FvwmIconBoxMaxIconSize 64x38
*FvwmIconBoxFont
-adobe-helvetica-medium-r-**-12-**-**-**-**-**
*FvwmIconBoxSortIcons
*FvwmIconBoxPadding 4
*FvwmIconBoxLines 10
*FvwmIconBoxPlacement Top Left
#
# mouse bindings
#
*FvwmIconBoxMouse      1      Click      RaiseLower
*FvwmIconBoxMouse      1      DoubleClick  Iconify
*FvwmIconBoxMouse      2      Click      Iconify -1, Focus
*FvwmIconBoxMouse      3      Click      Module
"FvwmIdent" ndings
#
# Key bindings
#
*FvwmIconBoxKey        x      RaiseLower

```

```

*FvwmIconBoxKey space Iconify
*FvwmIconBoxKey d Close

#
# FvwmIconBox built-in functions

#
*FvwmIconBoxKey n Next
*FvwmIconBoxKey p Prev
*FvwmIconBoxKey h Left
*FvwmIconBoxKey j Down
*FvwmIconBoxKey k Up
*FvwmIconBoxKey l Right

#
# Icon file specifications

#
*FvwmIconBox "+" unknown1.xpm
*FvwmIconBox "Mosaic" www-shape.xpm
*FvwmIconBox "xterm" terminal.xpm
*FvwmIconBox "GoodStuff" toolbox.xpm
*FvwmIconBox "*ircon*" daffy.xpm
*FvwmIconBox "*anual*" xman.xpm

#####
# Pager #####
*FvwmPagerBack #908090
*FvwmPagerFore #484048
*FvwmPagerFont -adobe-helvetica-bold-r-*-*-10-*-*-*-*-*-
*FvwmPagerHilight #cab3ca
*FvwmPagerGeometry 0+0
*FvwmPagerLabel 0 Strobel
*FvwmPagerLabel 1 Uh1
*FvwmPagerSmallFont 5x8

```

#### Az fvwm.modules fájl

A GoodStuff modul konfigurációja saját fájlban (`fvwm.goodstuff`) található. A GoodStuff segítségével a legfontosabb alkalmazások ikonként felvehetők egy ikontárra, így ezek a programok egyszerűen a megfelelő ikonra kattintva elindíthatók. Ez a Swallow beállítással hajtható végre (a példában az `xload` és az `xclock` programnál).

```

#####
# fvwm.goodstuff - Goodstuff . configuration
#
*GoodStuffBack gray60
*GoodStuffGeometry 65x715+1+0
*GoodStuffColumns 1
*GoodStuffFont -adobe-helvetica-medium-r-*-*-12-*-*-*-*-*-

```

#	Name	Icon	Action	WindowTitle	command
*GoodStuff	" "	" "	Swallow	"xclock"	xclock
-bg gray60	&				
*GoodStuff	" "	" "	Swallow	"xload"	xload
-bg gray60	&				
*GoodStuff	" "	" "	Swallow	"xbiff"	xbiff
-bg gray60	&				
*GoodStuff	Xterm	terminal.xpm	Exec	"xterm"	xterm
-sb -sl 400	&				
*GoodStuff	NetScape	www.xpm	Exec	"NetScape"	
netscape	&				
*GoodStuff	Xman	xman.xpm	Exec	"Manual Page"	xman
-bothshow -notopbox	&				
*GoodStuff	Mail	sndmail.xpm	Exec	"Mail"	xterm
-T Mail -e pine	&				
*GoodStuff	Emacs	editor.xpm	Exec	"emacs"	emacs
*GoodStuff	Exit	lbolt.xpm	Quit		

Az fvwm.goodstuff fájl

Az fvwm konfiguráció fenti csoportosítása nem feltétlenül szükséges, de így a system.fvwmrc konfigurációs fájl áttekinthetővé tehető.

# HÁLÓZATBA KAPCSOLÁS

**A**korszerű munkaállomások és operációs rendszerek vizsgálatában jelentős szempont a hálózati használat, azaz a rendszer meglévő hálózatba való integrálásának lehetősége. A Linux teljes kifejlesztése lehetetlen lett volna az Internet nélkül. Így már a rendszermag legelső verzióiban is szerepelt a TCP/IP, és rendelkezésre álltak a PC-s hálózati kártyákhoz szükséges kezelőprogramok.

Ez a fejezet a hálózatba kapcsolás alapjait mutatja be, és az alsóbb hálózati szintek konfigurációját ismerteti. Mivel a TCP/IP részletes tárgyalása túlmutat a könyv lehetőségein, számos téma nál egyszerűsítésre kényszerültünk. A további ismeretek megszerzéséhez az RFC ismertetőkre (lásd alább), valamint a TCP/IP protokolt és a hálózatkezelést megfelelő mélyesben tárgyaló könyvekre utalunk.

## 9.1 A HÁLÓZATI HARDVER

A hálózatba kapcsolás meglehetősen kevés hardvert igényel: két számítógép elemi helyi hálózatba (LAN) kapcsolható két egyszerű Ethernet kártya, Ethernet kábel, két T-csatlakozó és két lezáró ellenállás segítségével. Ez a megoldás a hálózat használatát egyéni felhasználók számára is elérhetővé teszi. A Linux az Arcnet kártya valamint szinte az összes népszerű Ethernet kártya kezelőprogramját tartalmazza. A számítógépek soros vagy párhuzamos kábelezéssel és modemek segítségével még olykor összeköthetők. Az ilyen hálózati használatot egyedi protokollok (például SLIP, CSLIP, PPP és PLIP) segítik.

## 9.2 TCP/IP

A UNIX számítógépek hálózati használatának tényleges szabványa a TCP/IP, az 1970-es évek elején az Internet számára kifejlesztett protokoll, amely azóta szinte valamennyi számítógépes platformon elérhetővé vált. A TCP/IP története szorosan kapcsolódik az Internethöz, így ezt a két téma kört általában együtt kezelik.

### Története

A TCP/IP az elsősorban az amerikai egyetemeket összekapcsoló ARPA NET (később Internet) számára kifejlesztett protokoll. A hálózatot az Advanced Research Project Agency kezdeményezte és finanszírozta. Ez az amerikai kormányhivatal katonai programokkal foglalkozott, így nem meg-lepő, hogy a TCP/IP protokollt az amerikai védelmi minisztérium szabványának nyilvánították. Időközben az Internet világszerte számos alhálózatot befogadó hálózattá fejlődött.

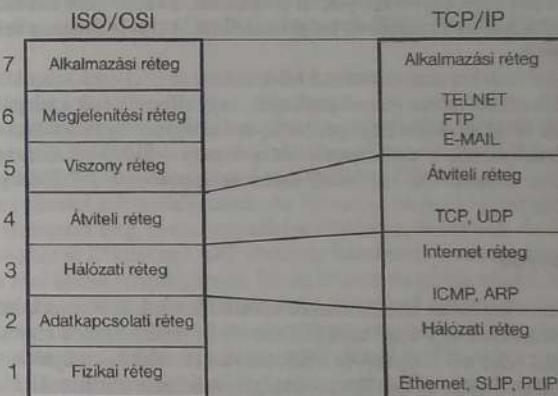
### RFC

A TCP/IP nem olyan szabvány, mint például az ANSI, az ISO vagy az IEEE, de mindenki számára hozzáférhető, gyártótól független leírása Request for Comments (RFC) formájában rendelkezésre

áll. (Az IETF-et, az Internet Engineering Task Force, az Internet műszaki fejlesztéséért felelős bizottságot azóta elfogadták szabványügyi hivatalnak, így a TCP/IP ugyanolyan szabvány, mint a többi – a lektor.) Az RFC általában egy protokoll leírását tartalmazza. (Az RFC-knek úgynevezett életciklusa van, amely meghatározza, hogy az RFC hogyan léphet egyik állapotából, „élelkorából” a másikba, vagyis a benne leírt protokoll állapota hogyan változik. Egy protokoll elindul az egyeni kezdeményezés szintjéről, később kísérleti protokoll lesz belőle, majd szabvány, amely tartalmától függően lehet kötelezően alkalmazandó vagy megkívánt, illetve javasolt. A protokoll élete végén elavult, „történelmi” illetve ellenjavallt állapotba kerül. minden állapotváltást alapos műszaki felülvizsgálatok sorozata előzi meg, amelyet széles közönség javaslatai alapján végeznek – a lektor.) Nem minden RFC igazi szabvány, sok közöttük inkább kevésbé hivatalos ismertető. Az RFC leírások RFC archívumokból és sok egyéb FTP kiszolgálóról FTP-n vagy e-mail-en keresztül szerezhetők be. Németországban például ez az [ftp.uni-stuttgart.de](http://ftp.uni-stuttgart.de), ahol az RFC-k felsorolása a /pub/doc/standards/rfc könyvtárban található. Az RFC-k az InterNIC (ds.internic.net) hivatalos archívumából is lekérhetők e-mail-en keresztül. Ehhez help tartalmú üzenetet kell küldeni a [mailserv@ds.internic.net](mailto:mailserv@ds.internic.net) címre. A kiszolgáló rendszerint fél órán belül leírással és részletes utasításokkal válaszol.

### Felépítés

A TCP/IP lényegében négy réteget tartalmaz (lásd a 9.1. ábrán). (Az ISO/OSI referencia modell és a TCP/IP összevetése nem szerencsés, mert a TCP/IP kb. 10 ével korábbi protokoll, így nyilván nem felelhet meg az OSI szabványnak. Amíg a felületesebb összevetés sikeres lehet, addig az alapos elemzés már lényeges különbségeket tár fel – a lektor.)



9.1 ábra. Az ISO/OSI és a TCP/IP szintek közötti kapcsolat

A második szint az Internet szint az Internet protokollal (IP). Ez az ISO/OSI referencia modell 3-as rétegének felel meg. Ez a réteg a tényleges hálózattól függetlenül adatesomagok (datagramok) átvitelét teszi lehetővé. Ezen a szinten a hálózati szoftver egyedi IP címök alapján azonosítja a számítógépeket. A harmadik szint az ISO/OSI referencia modell 4-es rétegének felel meg, és a TCP és az UDP protokoll használatát kínálja. A TCP a Transmission Control Protocol rövidítése, és virtuális kapcsolat létesítésére szolgál, míg az UDP (User Datagram Protocol) egyszerű csomagszolgáltatást nyújt.

A negyedik TCP/IP szint az ISO/OSI referencia modell 5-ös, 6-os és 7-es rétegének felel meg, elnevezésc alkalmazási szint. Míg az első három szint rendszerint az operációs rendszer része (a Linux esetében ezek a rendszermag részét képezik), a negyedik szint szokásos programokból áll. Ezben a szinten a leggyakoribb és legelterjedtebb szolgáltatások közé tartozik a Telnet virtuális terminálhoz, az FTP fájlok átviteléhez és az SMTP elektronikus levelezéshez. A Telnet és az FTP ugyanzenen a néven programként is létezik, és ezzel a névvel a felhasználó közvetlenül elindíthatja. A TCP/IP esetében az SMTP egyszerű átviteli mechanizmust jelent, amelyet számos egyéb program használ.

A továbbiakban a TCP/IP alsóbb szintjeit tárgyaljuk részletesebben. A következő szakasz az Internet és egyéb alkalmazások bemutatását tartalmazza.

### 9.3 IP

Az Internet protokoll (IP) adatesomagokat továbbít általában különböző számítógépekhez tartozó hálózati interfések között. A magasabb szintű protokolloval (például TCP) küldött adatok az IP szintre kerülnek, ahol az adatokat IP-csomagokká alakítják. Az irányítási táblázat alapján az IP szint meghatározza a csomagok tényeges küldési helyét, és egyidejűleg átadja ezeket a megfelelő hálózati szintnek. Ezt a folyamatot az alapvető fogalmak ismertetése után példán mutatjuk be.

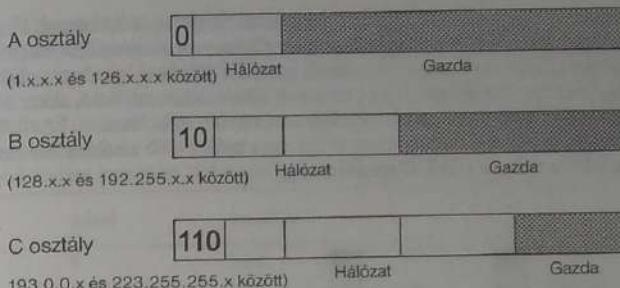
#### Címek

A TCP/IP protokoll használó hálózatokban egy számítógép minden egyes hálózati interfésze egyedi IP címet kap. A cím négy, egymástól ponțal elválasztott, 0 és 255 közé eső számból áll, például 141.7.1.40. Ez a cím hálózati részből és helyi címből áll. A cím beállítása a számítógép üzembe helyezésekor történik.

A kisebb, más hálózatokra nem csatlakozó hálózatoknál az IP címnek nincs különösebb jelentősége, a felhasználók tetszőlegesen megválaszthatják, vagy elfogadhatják a telepítőprogramban felkínált alapértéket. Itt az egyetlen lényeges szempont az egyediség biztosítása a helyi hálózaton belül. (Ezt már szabályozzák, a szabvány szerint igen nagy – 65000-nél több gépet tartalmazó – hálózathan a 10.x.y.z címeket kell használni, kisebb hálózatokban a 192.168.x.y címek közül kell választani – a lektor.)

#### Hálózati osztályok

Az IP címek Internet kapcsolat létrehozásakor válnak fontossá. A Network Information Center (NIC) az új alhálózatokhoz egy vagy több hálózati címet rendel. Ezek a címek az A, B vagy C osztályhoz tartoznak (lásd a 9.2. ábrán), és a hálózati cím részük az osztálytól függően 1, 2 vagy 3 bájtból áll. A címek első bitjei az osztályt azonosítják. A hálózati címet az IP cím eleje alkotja. A többi bitet a helyi hálózat adminisztrátora adhatja meg, ezt nevezik helyi címmek. A C osztályú hálózatokban az IP címek egy hivatalosan meghatározott 3 bájtos hálózati címből és egy helyi, 1 bájtos részből állnak. Mivel a 0-ra és 255-re végződő címek különleges jelentésűek, egy C osztályú hálózat legfeljebb 254 IP címet tartalmazhat. Az említett három osztály mellett D és E osztályú címek is léteznek, ezeket azonban az egyidejű végrehajtás és a bővítmények kezelésére tartják fenn.



9.2 ábra, Hálózati osztályok

Az IP címek helyi része az egyes gépek közvetlen címzésére használható, vagy hálózati maszk segítségével alhálózati címre és helyi címre osztható. Az alhálózatok használatát itt nem tárgyaljuk, erről részletesen lásd az RTC950 dokumentációt.

### Visszacsatoló eszköz

A ténylegesen hálózati interfészről nem tartalmazó számítógépekhez is tartozik egy visszacsatolónak (loopback-knak) nevezett virtuális interfész. Amint nevéből is következik, az ilyen interfésszen kiadott valamennyi információ bemenetként közvetlenül visszakerül a számítógéphez. Ez a gép számára TCP/IP kapcsolatok létrehozását teszi lehetővé önmagával. Ily módon a TCP/IP programok akár hálózati interfész nélkül is használhatók. A visszacsatoló interfész IP címe mindenig 127.0.0.1.

### ARP

A következő rész jobb megértése érdekében röviden tárgyaljuk a hálózati szintű címzést. Egy Ethernet példán bemutatjuk az IP csomagok átadását egyik számítógépről egy másikra.

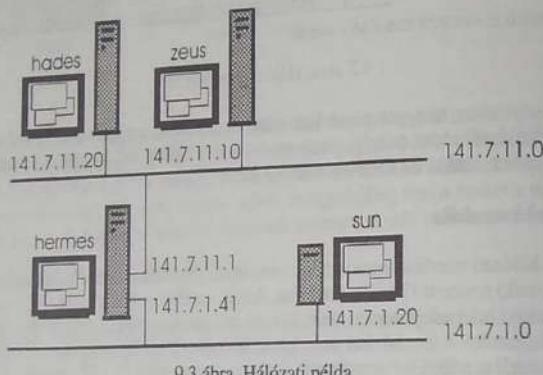
Az IP csomagokat Ethernet hálózaton való továbbításhoz Ethernet csomaggá kell alakítani, és ezeknek Ethernet címmel kell rendelkezniük. Az Ethernet címeket az Ethernet kártyák hardverében közvetlenül határozzák meg, és ezek egyedisége általánosan biztosított. Az IP csomag Ethernet csomaggá alakításakor az IP szint egy ARP (Address Resolution Protocol) csomagot küld az azonos alhálózatban lévő összes számítógépnek. Így az IP szint meghatározhatja, hogy mely Ethernet címre kell elküldeni a csomagot. Az ARP üzenet a következő kérdésként fogható fel: „Kérjük válaszoljon, ha ismeri az a.b.c.d IP című interfész Ethernet címét”. Ha a kívánt cím megtalálható ebben az alhálózatban, a megfelelő számítógép ARP választ küld, közölve az Ethernet címet. Az első számítógép ekkor az IP csomagból elkészítheti a megfelelően megcímezett Ethernet csomagot, és elküldheti azt.

### Útválasztás

Nemcsak az Interneten, de a nagyobb vállalatoknál és egyetemeken is a hálózatot nem egyetlen olyan Ethernet alkotja, amelyre az összes számítógép közvetlenül csatlakozik. A rendszer általában ismétlőkkel, hidakkal és forgalomirányítókkal összekapcsolt számos részből áll.

Mivel az IP-nek függetlennek kell lennie az alatta használt hálózati szintektől, a csomagok Ethernet címmel csak akkor címezhetők meg közvetlenül például az A számítógépről a B számítógépre, ha A és B azonos fizikai alhálózatban található. Ha a számítógépeket egy vagy több forgalomirányító választja el, akkor az adatforgalom csak a forgalomirányítón keresztül történhet. Ez azt

jelenti, hogy a csomagokat a hálózati szinten a forgalomirányítóra kell címzni. IP csomag címzésénél minden számítógépnek ismernie kell, hogy a célcím vele azonos alhálózatban található-e. Ha a célcím azonos alhálózaton van, a számítógép egy ARP kéréssel lekérheti az Ethernet címet, és a csomagot közvetlenül elküldheti. Ha a cím másik alhálózatban található, akkor a számítógépnek a csomagot a megfelelő forgalomirányító Ethernet címére kell címznie. Emellett egy számítógép több hálózati interfészét is tartalmazhat, és így egyidejűleg több alhálózathoz csatlakozhat. A számítógépnek tehát azt is meg kell állapítania, hogy egy adott csomagot melyik interfészre kell küldeni.



9.3 ábra. Hálózati példa

Ezt a döntési műveletet útválasztásnak nevezik, és ezt minden számítógép és a forgalomirányítók végrehajtják. Az IP szint rendelkezik egy megfelelő táblázattal, amely a hálózati vagy gazdacímet és egy interfész, vagy a következő forgalomirányító címét tartalmazza. A táblázat különleges eleme az *alapértelmezett átvonala (default route)*, amely pontos bejegyzés hiányában a csomag küldési helyét határozza meg.

### Példa

A példában egy B osztályú hálózatról lesz szó, amelyet egy hálózati maszk további alhálózatokra oszt. A hivatalosan megadott hálózati cím 141.7.0.0, míg a hálózati maszk 255.255.255.0. A hálózat címzésére két bájt, az alhálózat címzésére 1 bájt, míg az alhálózatban egy adott számítógép címzésére egy bájt áll rendelkezésre. A számítógép telepítésekor az IP cím konfigurálása meghatározza a hálózati maszket. Az elosztóként működő Linux PC két hálózati kártyát tartalmaz, és a 141.7.1.0, illetve a 141.7.1.1.0 alhálózatra csatlakozik. Ez a számítógép biztosítja a két alhálózat közötti kapcsolatot. A hálózati interfések címe 141.7.11.1 és 141.7.1.41. A példa vázala a 9.3. ábrán látható.

A Linux forgalomirányítási táblázata a következő bejegyzésekkel tartalmazza:

célhálózat/ célcím	átjáró	interfész
141.7.11.0	*	eth0
141.7.1.0	*	eth1
default	FH	eth0

Az elosztó közvetlenül kapcsolódik a két alhálózathoz, amelyek címe 141.7.11.0 és 141.7.1.0, de más IP címeket nem érhet el közvetlenül. Az alapértelmezett útvonal bejegyzés azt határozza meg, hogy az előzőektől eltérő című csomagokat az elosztó a központi FH Heilbronn elosztóba küldi. A két interfész (eth0 és eth1) hálózati maszkja 255.255.255.0.

A 141.7.11.0 című alhálózat másik számítógépének irányítási táblázata a következő bejegyzéseket tartalmazza:

célhálózat/ célcím	átjáró	interfész
141.7.11.0	*	eth0
alapérték	141.7.11.1	eth0

Ez a táblázat azt jelenti, hogy a számítógép csak az azonos alhálózatban lévő számítógépnek küldhet csomagot. Az ettől eltérő című csomagok a 141.7.11.1 című Linux elosztóba kerülnek.

Amikor a 141.7.11.10 című számítógép csomagot küld a 141.7.11.20 című számítógépnek, a hálózati maszkok és az irányítási táblázat bejegyzései meghatározzák, hogy a csomagot ténylegesen mely címre és melyik interfészen keresztül kell továbbítani. A hálózati maszk (255.255.255.0) csak az IP cím első három bájtját hasonlíta össze az irányítási táblázat bejegyzéseivel. Így a 141.7.11.20 és a 141.7.11.0 azonos értéket jelent forgalomirányítással szempontjából.

Az útvonalbejegyzés szerint a célcím elérhető közvetlenül az eth0 interfészen keresztül. Ha a 141.7.11.20 Ethernet címe továbbra sem ismert, akkor a számítógép ARP kérést intéz az alhálózata valamennyi számítógépéhez, amelyben a 141.7.11.20 Ethernet címét kéri. A cél-számítógép egy ARP válaszban saját Ethernet címét közli. A 141.7.11.10 című számítógép ezután elkészítheti az IP csomag **Ethernet csomagját**, és azt közvetlenül elküldheti a fogadóhoz.

A folyamat sokkal bonyolultabbá válik, ha a csomagot másik alhálózatra kell küldeni. A küldő számítógép irányítási táblázatában kikeresi a megfelelő című forgalomirányítóhoz tartozó bejegyzést. Ezután a csomag forgalomirányítóra küldéséhez meg kell állapítania annak Ethernet címét. A forgalomirányító, amely a példában egy Linux gép, fogadja az Ethernet csomagot, majd kibontja azt. Amikor az átjáró megállapítja, hogy a kapott IP csomag valójában másik címre szól, irányítási táblázata segítségével megkíséri elküldeni a csomagot.

## Konfigurálás

Az alábbiakban az előző példa 141.7.11.0 című alhálózatában található PC konfigurálását ismertetjük. Először a hálózati interfések konfiguráljuk az ifconfig parancs segítségével. Ezután az egyes interfésekre meghatározzuk az IP címet, a hálózati maszket és a broadcast címet. A broadcast címet akkor használjuk, amikor az alhálózat összes számítógépének szeretnénk csomagot küldeni. A legegyszerűbb esetben és a példában ez az a hálózati cím, amely 0 helyett 255-re végződik, azaz 141.7.11.255. Az ifconfig indítását és paramétereit a megfelelő kézikönyvoldal részletesen közli.

A kezelőprogramok felsorolják az indítási műveletnél rendelkezésre álló hálózati interfések neveit. Általában használt nevek például a következők: eth0 az Ethernet, lo0 a visszacsatoló, vagy s10 a SLIP első soros interfésze esetében.

Egy alhálózat szokásos számítógépeinek címénél az utolsó bájiban 2 és 253 közötti számot kell használni. Az 1 és a 254 általában a forgalomirányítók számára fenntartott, míg a 0 és a 255 a hálózati vagy a broadcast címekben használt.

Az interfések konfigurálása után a belső irányítási táblázat bejegyzéseit kell létrehozni, amely azt rögzíti, hogy adott címek mely interfészen keresztül érhetők el. Ez a művelet a `route` parancssal hajtható végre.

A következő példa az `/etc/rc.d/rc.inet1` parancsfájlt mutatja be, amely a 141.7.11.10 IP című számítógép szükséges beállításait hajtja végre.

```
#!/bin/sh
#
# rc.inet1    This shell script boots up the base INET system.
#
# Version:    @(#) /etc/rc.d/rc.inet1    1.01    05/27/93
#
#
HOSTNAME='hostname'

# Attach the loopback device.
/sbin/ifconfig lo 127.0.0.1
/sbin/route add -net 127.0.0.0

# IF YOU HAVE AN ETHERNET CONNECTION, use these lines below to configure the
# eth0 interface. If you're only using loopback or SLIP, don't include the
# rest of the lines in this file.

# Edit for your setup.
IPADDR="192.168.1.1"           # REPLACE with YOUR IP address!
NETMASK="255.255.255.0"         # REPLACE with YOUR netmask!
NETWORK="192.168.1.0"          # REPLACE with YOUR network address!
BROADCAST="192.168.1.255"       # REPLACE with YOUR broadcast address, if
you                                # have one. If not, leave blank
                                    # and edit below.
GATEWAY="192.168.1.254"        # REPLACE with YOUR gateway address!

# Uncomment the line below to initialize the ethernet device.
/sbin/ifconfig eth0 $IPADDR broadcast $BROADCAST netmask $NETMASK

# Uncomment these to set up your IP routing table.
/sbin/route add -net $NETWORK netmask $NETMASK
/sbin/route add default gw $GATEWAY metric 1
```

### A kapcsolatok ellenőrzése

A hálózati interfész `ifconfig` parancssal végrehajtott konfigurálása és az irányítási adatok bevitelle után sor kerülhet a kapcsolatok ellenőrzésére. Erré a `ping` program használható. Ez rendszerszintűen kis adatesomagokat küld a célgépre, amely ezután megfelelően válaszol.

```
nicetry:~$ ping test
PING zuse (193.6.135.45): 56 data bytes
64 bytes from 192.162.135.1: icmp_seq=0 ttl=255 time=3.0 ms
64 bytes from 192.162.135.1: icmp_seq=1 ttl=255 time=1.9 ms
64 bytes from 192.162.135.1: icmp_seq=2 ttl=255 time=1.8 ms
64 bytes from 192.162.135.1: icmp_seq=3 ttl=255 time=1.9 ms
64 bytes from 192.162.135.1: icmp_seq=4 ttl=255 time=1.9 ms

--- zuse ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 1.8/2.1/3.0 ms
nicetry:~$
```

Ez az eljárás az átviteli sebesség felmérésére is alkalmas. Ha a ping problémákat jelez, az ifconfig és a netstat parancs segítségével a paraméterek beállítását ellenőrizni és módosítani lehet. Ha az ifconfig hívása kapcsolók nélkül történik, akkor az a hálózati interfések állapotáról közöl adatokat. Néhány Linux változatban a netstat parancssal az -i kapcsoló is használható.

```
nicetry:~# ifconfig -a
lo      Link encap:Local Loopback
        inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
              UP BROADCAST LOOPBACK RUNNING MTU:3584 Metric:1
              RX packets:4195 errors:0 dropped:0 overruns:0
              TX packets:4195 errors:0 dropped:0 overruns:0

eth0      Link encap:10Mbps Ethernet  HWaddr 00:00:CO:92:B2:A1
        inet addr:192.168.1.1 Bcast:192.168.1.255 Mask:255.255.255.0
              UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
              RX packets:0 errors:0 dropped:0 overruns:0
              TX packets:9 errors:0 dropped:0 overruns:0
              Interrupt:10 Base address:0x310 Memory:cc000-d0000

nicetry:~#
```

A rendszermag irányítási táblázatát a kapcsoló nélküli route vagy a netstat -r parancssal jeleníthetjük meg.

```
nicetry:~# netstat -r
Kernel routing table
Destination     Gateway         Genmask        Flags Metric Ref Use Iface
ethernet        *           255.255.255.0    U       0     0      1 eth0
loopback        *           255.0.0.0      U       0     0      2 lo
nicetry:~#
```

170 | LINUX  
**9.4 SOROS CSATLAKOZTATÁS**

**9.4 SOROS CSATLAKOZTATÁS**  
A TCP/IP összeköttetés modemén keresztüli megvalósításához a SLIP (Serial Line Internet Protocol) vagy a PPP (Point-to-Point Protocol) protokollt használjuk. Ezen két protokollra alapozott rendszer a szokásos modemes összeköttetéssel szemben azzal az előnyvel rendelkezik, hogy egyidejűleg több felhasználó használhatja.

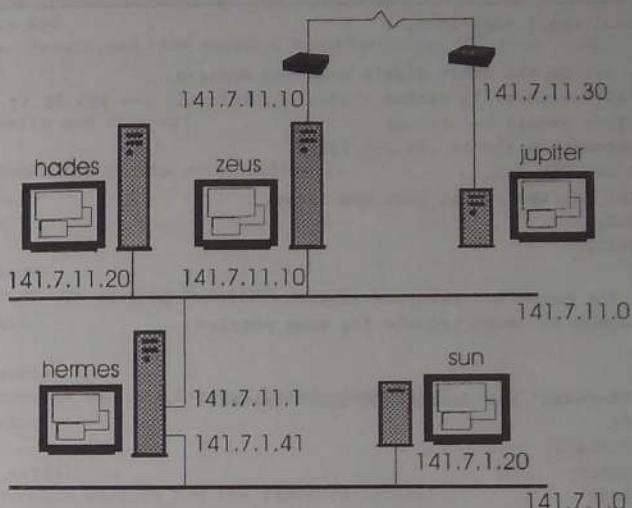
A 9.4. ábra a korábbi példából már ismert hálózatot mutatja, amely kiegészítve, itt a SLIP/PPP kiszolgáló feladatait a zeus gép látja el. Szokásos esetben a soros interfésekkel nem a rendszer indításakor konfigurálják, hanem csak akkor, amikor a működésükhez ténylegesen szükséges. Először ellenőrizni kell, hogy a szóban forgó rendszermag biztosítja-e a PPP vagy a SLIP használatát. A következő parancs a rendelkezésre álló interfésekkel listázza:

Interface	Receive						Transmit					
	Packets	errs	drop	fifo	frame	Packets	errs	drop	fifo	colls	carrier	
lo:	4195	0	0	0	0	4195	0	0	0	0	0	0
eth0:	0	0	0	0	0	9	0	0	0	0	0	0
ppp0:	0	0	0	0	0	9	0	0	0	0	0	0
ppp1:	0	0	0	0	0	9	0	0	0	0	0	0
ppp2:	0	0	0	0	0	9	0	0	0	0	0	0
ppp3:	0	0	0	0	0	9	0	0	0	0	0	0
s10:	0	0	0	0	0	9	0	0	0	0	0	0
s11:	0	0	0	0	0	9	0	0	0	0	0	0
s12:	0	0	0	0	0	9	0	0	0	0	0	0
s13:	0	0	0	0	0	9	0	0	0	0	0	0

A fenti példában az Ethernet interfész mellett négy SLIP (s10-s13) és négy PPP (ppp0-ppp3) eszköz áll rendelkezésre.

SLIP

A SLIP protokoll egyedi IP csomagokat változtatlanul továbbít a vonalakon, ami így viszonylag kis átviteli sebességet eredményez. Éppen ezért újabban a CSLIP (Compressed Serial Internet Protocol) nevű módosított változatát használják. A CSLIP az átvitel előtt tömöríti az IP fej részt, ami az átviteli sebességet észrevehetően növeli.



9.4 ábra. A TCP/IP megvalósítása modemmel

A dip segédprogram szolgál a telefonszám tárcsázására és a kapcsolat létrehozására a SLIP kiszolgálóval. A művelet végrehajtható interaktív módon vagy vezérelhető előre megírt programból (szkript). A következő példa olyan egyszerű programrészletet mutat be, amely egy Linux kiszolgálóval veszi fel a kapcsolatot. Elindítása a dip <szkriptnév> parancssal történik.

```

# sample.dip      Dialup IP connection support program.
# This file (should show) shows how to use the DIP
# scripting commands to establish a link to a host.
# This host runs the 386bsd operating system, and
# thus can only be used for the "static" addresses.
#
# NOTE:          We also need an example of a script used to connect
#                to a "dynamic" SLIP server, like an Annex terminal
#                server...
#
# Version:       @(#)sample.dip1.4007/20/93
#
# Author:        Fred N. van Kempen, <waltje@uWalt.NL.Mugnet.ORG>
# Modified:      Uri Blumenthal <uri@watson.ibm.com>
# Modified:      Jim Van Zandt <jrv@vanzandt.mv.com>
#
main:
  # First of all, set up our name for this connection.
  # I am called "uwalt.hacktic.nl"  (== 193.78.33.238)

```

```

get $local walt.hacktic.nl

# Next, set up the other side's name and address.
# My dialin machine is called 'xs4all.hacktic.nl' (== 193.78.33.42)
get $remote xs4all.hacktic.nl
# Set netmask on s10 to 255.255.255.0
netmask 255.255.255.0
# Set the desired serial port and speed.
port cu02
speed 38400

# Reset the modem and terminal line.
# This seems to cause trouble for some people!
reset

# Note! "Standard" pre-defined "errlvl" values:
# 0 - OK
# 1 - CONNECT
# 2 - ERROR
# 3 - BUSY
# 4 - NO CARRIER
# 5 - NO DIALTONE

# You can find those grep'ping for "addchat()" in *.c...
# You can change this with the "chatkey" command.

# Prepare for dialing.
send ATQ0V1E1X4
wait OK 2
if $errlvl != 0 goto modem_trouble
dial 555-1234567
if $errlvl != 1 goto modem_trouble

# We are connected. Login to the system.
login:
sleep 2
wait ogin: 20
if $errlvl != 0 goto login_error
send MYLOGIN
wait ord: 20
if $errlvl != 0 goto password_error
send MYPASSWD
loggedon:

# We are now logged in.
wait SOMETEXT 15
if $errlvl != 0 goto prompt_error

# Set up the SLIP operating parameters.

```

```
get $mtu 296
# Ensure "route add -net default xs4all.hacktic.nl" will be done
default

# Say hello and fire up!
done:
print CONNECTED $locip ---> $rmtip
mode CSLIP
goto exit

prompt_error:
print TIME-OUT waiting for SLIPlogin to fire up...
goto error

login_trouble:
print Trouble waiting for the Login: prompt...
goto error

password_error:
print Trouble waiting for the Password: prompt...
goto error

modem_trouble:
print Trouble occurred with the modem...
error:
print CONNECT FAILED to $remote
quit

exit:
exit
```

A kapcsolat létrehozása után a `clp` automatikusan konfigurálja a SLIP interfészt, és meghatározza azt a szabványos útvonalbejegyzést, amely alapján az összes csomag a Linux kiszolgálóra kerül átadásra.

## 9.5 PPP

A Point-to-Point Protocol (**PPP**) a másik módja TCP/IP kapcsolat létrehozásának modemen keresztül. A PPP szabványosítása számos RFC-ben hozzáférhető. Emellett a PPP lehetővé teszi más protokollok (például **Appletalk** és **Novell IPX**) használatát is az átvitelben. A PPP saját azonosító protokollal rendelkezik, amely megfelelő biztonságot nyújt a telefonos kapcsolatokban. A PPP további előnye az **IP cím dinamikus hozzárendelésének lehetősége**. Ez azt jelenti, hogy a kiszolgáló a kapcsolat létrehozásakor megfelelő címet rendelhet az ügyfélhez. A CSLIP protokollhoz hasonlóan a PPP is lehetővé teszi az IP fejrések tömörítését, ami az átviteli sebesség jelentős növelését eredményezi.

**Konfigurálás**

A Linux alatt a PPP ügyfelek konfigurálása hasonló a SLIP kapcsolat esetében megismerttel, és véghajtása is ugyanolyan egyszerű. Ehhez a pppd PPP démon és a chat program szükséges. A Linux PPP csomagja mindenből tartalmazza. A démon telepítéskor speciális fájlvédelem szükséges: a fájltulajdonos legyen root, és a SUID bitet 1-be kell állítani:

```
nicetry:~# ls -l /usr/sbin/pppd
-r-sr-xr-x 1 root      root    74652 Jul 21 18:47 /usr/sbin/pppd*
nicetry:~#
```

A pppd teljesen önállóan véghajtja a kapcsolat létrehozását, és vezérli az IP csomagok átvitelét. A számos rendelkezésre álló beállítási lehetőség miatt a parancssor meglehetősen bonyolultnak tűnhet, ezért célszerű az adatokat előre rögzíteni:

```
#!/bin/sh
# pppcall: PPP kapcsolat telepítése

# telefonszám
NR=1234567

# Login
PROMPT1=cgin:
LOGIN=ppp

# Jelszo
PROMPT2=word:
PASSWD=titok

MODEM=/dev/cua0
SPEED=57600

DIAL=ATDT
PPPD=/usr/sbin/pppd

echo "Kapcsolódás a ($NR) számon levo kiszolgalochoz..."
$PPPD connect "chat -v ABORT BUSY ABORT 'NO CARRIER' "
$DIAL$NR CONNECT '' $PROMPT1 $LOGIN $PROMPT2
$PASSWD" $MODEM $SPEED -detach crtscts modem defaultroute
```

A másik lehetőség, hogy a gyakran használt beállításokat az /etc/ppp/options nevű fájlba írjuk. Ennek a fájlnak, használatától függetlenül, léteznie kell (esetleg üres is lehet).

```
# /etc/ppp/options: Global options of pppd
#
lock          # UUCP conform lock file
modem         # modem connectioncrtscs
               # hardware flow control-detach
               # no fork
```

Amint a fenti példából látszik, a `chat` parancs végrehajtja a távoli modem felhívását és a bejelentkezést. A `chat` szkript karakterszorozatokat küld a modemre, vagy adott sorozat vételére vár, így működése hasonló az UUCP kapcsolatot létrehozó `chat` szkript működéséhez.

A `chat` programnak, amint ez a fenti példában is látható, tartalmaznia kell a `BUSY` és a `NO CARRIER` megszakítás-karaktersorozatot is. Ezzel elkerülhető, hogyha a kapcsolat felépítése sikertelen, a program befejezésére történő hosszú várakozás. A fenti példában a `chat` a 0815-ös számot hívja, ezután a `CONNECT` válaszra ér a bejelentkezési üzenetre vár. A PPP kapcsolat létrehozása előtt bejelentkező névként a `ppp`, míg jelszóként a `joshua` a megadott. Ez a rész esetleg módosításra szorul, mivel a kiszolgálóról érkező bejelentkezési üzenet időről időre változhat.

A `chat` és a `pppd` hibakeresési kapcsolója hasznos lehet a konfigurációs hibák felfedezésében. A `chat` -v kapcsolójával, a `syslogd` démon segítségével, az összes művelet listáját kapjuk. A `pppd` esetében az ennek megfelelő kapcsoló a -debug.

Ha másképp nincs megadva, a PPP kapcsolatnál az ügyfél oldalon a helyi IP címet használja a rendszer. **Dinamikusan hozzárendelt cím** a `noipdefault` beállítással kérhető le a kiszolgálóról. A `defaultroute` beállítást is át kell küldeni, hogy így az alapértelmezett útvonal automatikusan a PPP kiszolgáló legyen. Ha alapértelmezett útvonalat nem kíván definiálni, az útválasztás közvetlenül is megadható.

Ennek végrehajtásához a `pppd` a kapcsolat sikeres létrezására után elindítja az `/etc/ppp` könyvtárban található `ip-up` szkriptet. Paraméterként a hálózati interfések neve, a soros interfések elérési útvonala, az átviteli sebesség, a helyi cím és a kihelyezett állomás IP címe kerül átadásra. Az átviteli paraméterektől függően az alábbihoz hasonló szkripttel tetszőleges útvonalak beállíthatók:

```
#!/bin/sh
# IP-UP script for PPP connections
#
case $5 in
  192.7.11.1)
    /sbin/route add -net 192.7.11.0 gw 192.7.11.1
  192.7.12.1)
    /sbin/route add -net 192.7.12.0 gw 192.7.12.1
esac
```

Az `ip-down` program az `ip-up` megfelelője, amely a PPP kapcsolat lezárása után teszi lehetővé tetszőleges számú művelet végrehajtását.

A PPP kapcsolat megszakításához elegendő befejező jelet (`TERM`) küldeni a `pppd`-be a `kill` parancssal. A futó démon processazonosítója a `/var/run/pppN.pid` fájlból található, ahol N a kapcsolt interfész száma. A következő parancs megszakítja a kapcsolatot:

```
nicetry:~# killall pppd
```

Ha párhuzamosan több démon fut, akkor csak a megfelelő pppd-t szabad leállítani. Ez a következő parancssal hajtható végre:

```
nicetry:~# kill `cat /var/run/ppp0.pid`
```

A fenti parancs biztosítja, hogy csak a ppp0 interfészről kezelő pppd futása fejeződjék be.

### Azonosítás

A telefonos kapcsolatokban a jogtalan behatólk elleni védekezés további biztonsági intézkedéseket tesz indokolttá. A PPP két külön protokollt biztosít erre a célra. A Password Authentication Protocol (PAP) lényegében egy egyszerű bejelentkezési művelethez hasonlít. Ez az induláskor a felhasználó nevét és a nem kötelezően használt kódolt jelszót ellenőrzi. A Challenge Handshake Authentication Protocol (CHAP), amelyről később még szó lesz, lényegesen nagyobb biztonságot nyújt. A CHAP adott időközönként különleges adatok cseréjét hajtja végre, így a kapcsolatban lévő állomások kölcsönösen ellenőrizhetik egymás azonosságát.

Az ilyen típusú kölcsönös azonosításhoz az auth kulcsszót fel kell venni az /etc/ppp/options fájlba (vagy a pppd parancssorra). minden kapcsolathoz az egyes kiszolgálókat egyedileg azonosító titkos adatok definiálhatók. A protokolltól függően ezek vagy az /etc/ppp/pap-secrets, vagy az /etc/ppp/chap-secrets fájlból találhatók. Először minden CHAP azonosítás végrehajtására kerül sor, majd ezt követi a PAP azonosítási eljárás. Ha a fenti fájlok egyike sem található, a kapcsolat megszakad.

A CHAP titkos adatokat tartalmazó fájl formátuma a következő lehet:

#	# Chap Secrets for nicetry			
	Client	Server	Secret	Adress
zeus.demo.de	jupiter.demo.de	"catbox"	zeus.demo.de	
jupiter.demo.de	zeus.demo.de	"Eat brown"	jupiter.demo.de	
*	zeus.demo.de	"00Schneider"	192.168.11.50	

Az első oszlop a megfelelő kihelyezett állomás nevét tartalmazza. A Server-oszlop annak a gépnak a nevét tartalmazza, ahol az azonosítás történik. A harmadik oszlopan a titkos adatok találhatók. Az utolsó oszlop azt a gazdanetet vagy IP címet tartalmazhatja, amelyről az ügyfélnek be kell jelentkezni. Ez tetszőleges számú ügyfelet megengedő rendszerekben fontos.

A hibamentes azonosítás érdekében az Ügyfélgépet úgy kell konfigurálni, hogy a hostname meghívásakor teljes nevvel válaszoljon (beleértve a körzetet is). Ha a hostname csak az egy-szerű nevet adja vissza, a domain a pppd domain beállításával határozható meg.

### PPP

A Linux alatt futó PPP démon PPP kiszolgáló létrehozására is használható. Ehhez pl. egy ppp felhasználói névvel rendelkező felhasználó bevitellel szükséges az /etc/password fájlba.

```
ppp:Nikn1AMmYgj3c:997:1000:Public PPP:/tmp:/etc/ppp/ppplogin
```

Bejelentkezési shellként egy ppplogin nevű szkriptet használunk, amely például a következő lehet:

```
#!/bin/sh
#
# ppplogin - Login shell for PPP login
#
# prevents terminal output via write(1)
mesg n
# turns echo off
stty sane
# starts pppd in server mode
exec /usr/sbin/pppd -detach silent modem proxyarp crtscts
```

A silent kapcsoló hatására a démon a kapcsolat létrehozásának kísérlete előtt bejövő csomagokra vár. A proxyarp biztosítja, hogy a kiszolgáló válaszoljon a pont-pont összeköttetésben lévő kihelyezett állomásokat érintő ARP kérésekre (lásd a 9.3. részben). Ily módon egy modemhez kapcsolt számítógép a helyi Ethernet hálózatban lévő géphez hasonlóan címezhető meg.

## 9.6 PÁRHAZAMOS ÖSSZEKÖTTELÉS

Egy párhuzamos port és a PLIP (Parallel Line Internet Protocol) protokoll segítségével gazdaságos TCP/IP kapcsolat építhető fel. Ekkor szabad párhuzamos porton keresztül két számítógépet kötünk össze null nyomtatókábellel (*LapLink* kábelnek is nevezik). Az ilyen típusú hálózat különösen jól használható adatátvitelre hordozható számítógép és munkaállomás között.

Elvben egy PLIP interfész pontosan úgy működik, mint egy soros hálózati interfész (SLIP vagy PPP). Az első párhuzamos interfész neve *plip0*. Ügyeljünk arra, hogy a rendszermag fordításakor megadjuk a PLIP beállítást (lásd a 6.2. részben). Pont-pont kapcsolat konfigurálásához az *ifconfig* hívásakor a *pointtopoint* beállítást és az IP címet kell átadni a kihelyezett állomásnak.

```
#!/bin/sh
#
# PLIPUP - init plip interface
#
# Edit for your setup.
IPADDR="192.168.1.1"          # REPLACE with YOUR IP address!
NETMASK="255.255.255.0"        # REPLACE with YOUR netmask!
POINTTOPPOINT="192.168.1.2"    # REPLACE with YOUR network address!
GATEWAY="192.168.1.2"
echo "Setting up PLIP interface"
```

```
/sbin/ifconfig plip0 $IPADDR netmask $NETMASK pointopoint $POINTOPOINT
/sbin/route add $POINTOPOINT dev plip0
/sbin/route add default gw $GATEWAY metric 1
echo "PLIP is running."
```

## 9.7 A TCP ÉS AZ UDP

Az Internet alkalmazások ritkán használják közvetlenül az IP-szintű rutinokat. Helyette inkább a TCP vagy az UDP protokollt használják, amelyek az IP-nél nagyobb címzési lehetőségeket biztosítanak.

A TCP virtuális kapcsolatot hoz létre. Ez azt jelenti, hogy egy TCP szintű kapcsolat egy biztonságos adatcsatornának fogható fel. minden, amit a csatorna egyik végéről elküldünk, a csatorna másik végén olvasható. A TCP lebontja a küldendő fájlokat, és küldésükhez az IP protokollt használja. Ha az átvitelben hiba történik, vagy ha az IP csomagok elvesznek, a TCP szintű feladat az átvitel megismétlése.

Az UDP csak egyedi csomagok küldésére alkalmas. A csomag tartalmát egy ellenőrző összeggel hasonlíta össze, amelyet a felhasználói program generál. Ha a vevő átviteli hibát jelez, a csomag nem kerül ismételten elküldésre, hanem törli a protokoll. Az UDP használata nem olyan kényelmes, mint a TCP, de meglehetősen hatékony.

### Portok

Az ezen a szinten történő kapcsolat létrehozásához az IP címen kívül egy portszám is szükséges, amely általában egy adott hálózati szolgáltatásnak felel meg. Ez egyetlen hálózati interfészen keresztül több független kapcsolatot tesz lehetséges. A portszámok közül több a szabványos TCP/IP alkalmazások részére föntartott, így például a 23-as port a telnet, a 20-as és a 21-es az ftp számára. Ezeket a föntartott portokat az /etc/services fájl tartalmazza.

### Foglalatok (socket)

A hálózati alkalmazások programozása számára egységes interfész biztosítására a **BSD UNIX** fejlesztői az 1980-as évek elején bevezették a Berkeley foglalatokat. Ezek két számítógép összekapcsolásához és közöttük adatok átviteléhez szükséges központi rutinok sorozatából állnak. Mind a TCP, mind az UDP összekötések foglalatokon keresztül dolgoznak.

Fontos! Függetlenül attól, hogy az adatátvitel helyi hálózaton (Local-Area Network, LAN) vagy nagy területre kiterjedő hálózaton (Wide-Area Network, WAN) keresztül történik, a programozási interfész mindig ugyanaz.

A Linux rendszermag tartalmaz megfelelő foglalatinterfészeti, így a legismertebb UNIX hálózati alkalmazások hozzáérhetők.

## 9.8 GÉPNEVEK

A legtöbb felhasználó az IP címek közvetlen bevitelét nem tartja kényelmes megoldásnak. Többek között ez a tény is összönözte a szimbolikus címek bevezetését. A szimbolikus címek a gazzanévből és a körzettelnevből állnak.

A gépnév és a körzettelnev a számítógépet az egész világon egyedileg azonosítja. Például a 141.7.1.40 IP cím helyett a linux1.rz.fh-heilbronn.de szimbolikus cím használható, ebben az esetben a linux1 a gépnév, míg az rz.fh-heilbronn.de olyan körzet neve, amely a fh-heilbronn.de alkörzete.

Az IP címek és a szimbolikus címek közötti megfeleltetés az /etc/hosts fájlból helyileg adható meg. Itt további nevek, úgynevezett hivatkozási nevek (alias) definiálhatók a gépre. A hivatkozási nevek általában rövidebbek, így a címek bevitelle egyszerűbb. A következő példa az /etc/hosts fájl egy lehetséges részletét mutatja be:

```

#
# hosts      This file describes a number of hostname-to-address
#             mappings for the TCP/IP subsystem. It is mostly
#             used at boot time, when no name servers are running.
#             On small systems, this file can be used instead of a
#             "named" name server. Just add the names, addresses
#             and any aliases to this file...
#
# By the way, Arnt Gulbrandsen <agulbra@nvg.unit.no> says that 127.0.0.1
# should NEVER be named with the name of the machine. It causes problems
# for some (stupid) programs, irc and reputedly talk. :^)
#
127.0.0.1      localhost
192.168.1.1    moon
192.168.1.2    nicetry.try.net nicetry
192.168.1.5    mars
192.168.1.6    sxlin
192.168.1.254   sc
192.168.1.2    ipx
# End of hosts.

```

Részlet az /etc/hosts fájlból

## DNS, névkiszolgáló

A fenti fájlból több száz, vagy akár több ezer bejegyzés karbantartása szinte lehetetlen, ezért a szimbolikus nevek kezelésére és a címek automatikus megfeleltetésére névkiszolgálók (name server) hierarchikus rendszerét hozták létre. minden egyes körzethez, azaz egy adott körzettelnevezhez tartozó összes számítógéphez egy felelős névkiszolgáló tartozik. A körzettelnev, az alkörzet és névkiszolgáló fogalmáról és rendszeréről részletes ismertetés található az RFC1034 és az RFC1035 dokumentációban.

A UNIX TCP/IP programok egy C könyvtárrutin meghívásával a gépneveket a megfelelő IP címre konvertálják. Ez a resolver nevű rutin az /etc/hosts fájlt olvassa, és ha szükséges

(azaz, ha a cím saját körzetén kívülre csik), kapcsolatot létesít a következő névkiszolgálóval. Ennek sorrendje az /etc/host.conf fájlban adható meg.

```
order hosts, bind
multi on
```

#### Az /etc/host.conf fájl

Az `order hosts, bind` bejegyzés azt jelenti, hogy az /etc/hosts fájlt kell először olvasni. Ha `cz` nem tartalmaz bejegyzést a szóban forgó gépnévről, a névkiszolgálóval kell felvenni a kapcsolatot. A következő névkiszolgáló címe az /etc/resolv.conf fájlból található. A `multi on` sor jelentése: a resolver rutinnak egy gép összes érvényes címét vissza kell adnia, ha az /etc/hosts fájlból több címbejegyzés található. A kézikönyv `resolv+` oldala részletesen ismerteti ezeket a paramétereiket.

A resolver rutin tényleges konfigurációját az /etc/resolv.conf fájl tárolja. Ez a fájl emellett tartalmazza a legközelebbi névkiszolgáló címét, a helyi körzet nevét, valamint a névkiszolgálók lekérdezéséhez a gépnév-kiterjesztésekét is.

```
domain try.net
nameserver 192.162.1.1
nameserver 192.162.4.1
search try.net
search dialin.try.net
```

A `search` sorok azt jelentik, hogy a névkiszolgáló lekérdezésében minden a `demo.de`, minden a `beispiel.de` nevet meg kell próbálni a gazdanév kiterjesztéseként. A gépnév így a körzetnév nélkül is elegendő egy másik körzethez tartozó számítógépnél, feltéve, hogy a gépnév egyedi.

### Névkiszolgálók

Számos Linux telepítőcsomag tartalmazza a `named` névkiszolgáló démont. Ennek a konfigurálása nem egyszerű, de a kisebb hálózatokban a névkiszolgáló általában nem szükséges. A névkiszolgálók működését és a TCP/IP konfigurációs fájlok részletesen ismerteti a *Linux Network Administration Guide* (NAG), valamint az O'Reilly kiadásában megjelent *DNS and Bind*. Ez a rész csak a használt programok és fájlok rövid áttekintésére vállalkozik. (Mivel egy DNS konfigurációs fájl összeállítása nem egyszerű feladat, ezért a könyv példái a DNS szerver-szoftver, a BIND rendszer példa-fájljait tartalmazzák.)

A `named` névkiszolgáló démon első konfigurációs fájlja a /etc/named.boot. Ez a fájl meghatározza azt a körzetet, amelyeknek ez a kiszolgáló az elsődleges felcímese, és azt a körzetet, amelynek neveit és címeket rendszeresen kezelnie kell, valamint azokat a fájlokat, amelyekben a megfelelő táblázatok találhatók.

```
; boot file for authoritative master name server for Berkeley.EDU
; Note that there should be one primary entry for each SOA record.
```

```

sortlist 10.0.0.0

directory      /usr/local/adm/named

; type      domain           source host/file        backup file

cache          .               root.cache
primary       Berkeley.EDU   berkeley.zone
primary       32.128.IN-ADDR.ARPA berkeley.rev
primary       0.0.127.IN-ADDR.ARPA localhost.rev

```

Az /etc/named.boot fájl

A fenti példában az egyes táblázatok az /etc/bind könyvtárban találhatók. Ezt a könyvtárat az /etc/named.boot fájl is megadja. A táblázatok két típusba sorolhatók: az egyik típusú a címeket és a gépnév egyéb adatait szolgáltatja, míg a másik típusú a címek gépnevét adja meg. A következő példákból minden két típusú tábla részleteit megismerhetjük.

```

;                                         @(#)named.local 1.1      (Berkeley)      86/01/21
;

@     IN      SOA      ucbvax.Berkeley.EDU. kjd.ucbvax.Berkeley.EDU. (
                      1986012101 ; Serial
                      3600      ; Refresh
                      300       ; Retry
                      3600000  ; Expire
                      14400 )  ; Minimum
                      IN      NS      ucbvax.Berkeley.EDU.
0      IN      PTR      loopback.ucbvax.Berkeley.EDU.
1      IN      PTR      localhost.

```

A db.127.0.0 fájl

```

;                                         @(#)named.rev    1.1      (Berkeley)      86/02/05
;

@     IN      SOA      ucbvax.berkeley.edu kjd.ucbvax.berkeley.edu (
                      1986020501 ; Serial
                      10800     ; Refresh 3 hours
                      3600      ; Retry   1 hour
                      3600000  ; Expire  1000 hours
                      86400 )  ; Minimum 24 hours
                      IN      NS      ucbvax.Berkeley.EDU.
0.0   IN      PTR      Berkeley-net.Berkeley.EDU.
                      IN      A       255.255.255.0

```

			PTR	Csdiv-net.Berkeley.EDU.
0.130	IN		PTR	monet.Berkeley.EDU.
2.129	IN		PTR	ucbarpa.Berkeley.EDU.
2.140	IN		PTR	cad.Berkeley.EDU.
3.132	IN		PTR	ucbarpa.Berkeley.EDU.
4.0	IN		PTR	cad.Berkeley.EDU.
5.0	IN		PTR	ernie.Berkeley.EDU.
6.0	IN		PTR	monet-cs.Berkeley.EDU.
6.130	IN		PTR	monet.Berkeley.EDU.
7.0	IN		PTR	kim.Berkeley.EDU.
7.130	IN		PTR	esvax.Berkeley.EDU.
9.0	IN		PTR	cbxbav.Berkeley.EDU.
10.0	IN		PTR	kim.Berkeley.EDU.
11.0	IN		PTR	esvax-156.Berkeley.EDU.
11.156	IN		PTR	monet.Berkeley.EDU.
38.131	IN		PTR	

Adh.141.7.11 fájl

```
; Authoritative data for Berkeley.EDU (ORIGIN assumed Berkeley.EDU)
;
;                               IN      SOA      ucbvax.berkeley.edu kjd.ucbvax.berkeley.edu (
;                                         1986020501      ; Serial
;                                         10800      ; Refresh 3 hours
;                                         3600      ; Retry   1 hour
;                                         3600000  ; Expire  1000 hours
;                                         86400 ) ; Minimum 24 hours
;
;                               IN      MX      ucbvax 10
;                               IN      NS      monet
;
localhost      IN      A       127.1
ucb-arpa      IN      A       10.0.0.78
                IN      A       128.32.0.4
;
arpa          IN      HINFO    VAX-11/780 UNIX
ucb-vax       9999  IN      CNAME    ucbarpa
                IN      A       10.2.0.78
                IN      A       128.32.0.10
;
ucbvax        IN      HINFO    VAX-11/750 UNIX
monet         IN      CNAME    ucbvax
                IN      A       128.32.0.7
;
ucbmonet     IN      HINFO    VAX-11/750 UNIX
                IN      CNAME    monet
```

## A db.demo fájl

A `named.root` fájl különleges fontosságú: ez a központi névkiszolgálók címét tárolja.

```

; This file holds the information on root name servers needed to
; initialize cache of Internet domain name servers
; (e.g. reference this file in the "cache . <file>" configuration
; file of BIND domain name servers).

; This file is made available by InterNIC registration services
; under anonymous FTP as
;   file          /domain/named.root
;   on server     FTP.RS.INTERNIC.NET
; -OR- under Gopher at
;   under menu    InterNIC Registration Services (NSI)
;   submenu       InterNIC Registration Archives
;   file          named.root

; last update: May 11, 1994
; related version of root zone: 940516

;
;          99999999 IN NS  NS.INTERNIC.NET.
NS.INTERNIC.NET. 99999999 A  198.41.0.4
.
;          99999999 IN NS  NS1.ISI.EDU.
NS1.ISI.EDU. 99999999 A  128.9.0.107
.
;          99999999 IN NS  C.NYSER.NET.
C.NYSER.NET. 99999999 A  192.33.4.12
.
;          99999999 IN NS  TERP.UMD.EDU.
TERP.UMD.EDU. 99999999 A  128.8.10.90
.
;          99999999 IN NS  NS.NASA.GOV.
NS.NASA.GOV. 99999999 A  128.102.16.10
.
;          99999999 IN NS  192.52.195.10
.
;          99999999 IN NS  NS.NIC.DDN.MIL.
NS.NIC.DDN.MIL. 99999999 A  192.112.36.4
.
;          99999999 IN NS  AOS.ARL.ARMY.MIL.
AOS.ARL.ARMY.MIL. 99999999 A  128.63.4.82
.
;          99999999 IN NS  NIC.NORDU.NET.
NIC.NORDU.NET. 99999999 A  192.36.148.17
; End of File

```

A named.root fájl

## Hibakeresés

Az nslookup programot gyakran használják a névkiszolgálókkal kapcsolatos hibák felidézésére. A programmal a névkiszolgálók interaktív módon lekérdezhetők. A program indítható a parancssorban megadott névvel vagy argumentumok nélkül. Az argumentum nélküli futtatásnál az nslookup bevitelkérőjelet jelenít meg, amelyre a felhasználó különleges parancsokkal vagy a keresett nevekkel válaszolhat. A help parancssal a használható legfontosabb parancsok megjeleníthetők.

## 9.9 UUCP

Az UUCP a **UNIX to UNIX Copy program** rövidítése, és ez a UNIX gépek hálózati összeköttetésének valószínűleg legrégebbi megoldását jelenti. A kapcsolat létrehozása általában két egyedi gép között történik soros kábel és **modem** segítségével. Az UUCP hálózat tetszőleges mennyiségi adatot és parancsot továbbíthat. Általában ez biztosítja a leggazdaságosabb levelező és hírszolgáltatást.

Az UUCP az *adatovábbítás* (*data forwarding*) elvén működik. Ez azt jelenti, hogy először egy menetben egy adatescomagot továbbít, majd ezután a kihelyezett gépen az átküldött adatok feldolgozására szolgáló parancsot. Levél küldésénél például az `rmail` parancs kerül továbbításra a levél tényleges tartalmával együtt. Az adatok általában nem keletkezésük pillanatában, azonnal kerülnek elküldésre, hanem bizonyos időközönként (kötegelt rendszer). Ez a módszer a telefonköltségeket a lehetséges legalacsonyabban tartja, mivel kihasználhatók az olcsóbb tarifákú időszakok. Az adatok keletkezésükkor egy különleges könyvtárba tölti a rendszer, majd innen a sikeres átvitel után töri ezeket.

Az UUCP nem csak két közvetlen összeköttetésben álló számítógép között teszi lehetővé az adatátvitelt, hanem a tárolás és továbbítás (*store and forward*) elve alapján az átvitel megvalósítható több állomáson keresztül is. Ekkor azonban ismerni kell a tényleges adatátviteli útvonalat. Ezt a lehetőséget az UUCP korai szakaszához képest manapság már egyre kevésbé használják. Mivel nagyobb hálózatokban az útvonalak nyilvántartása az UUCP táblázatokkal meglehetősen nehézkes, a felhasználók általában az IP-alapú átvitelt részesítik előnyben. Napjainkban az UUCP elsődleges felhasználási területe adatok átvitelle egy Internetre TCP/IP protokollon keresztül csatlakozó kiszolgálóról közvetlen összeköttetéssel nem rendelkező kisebb rendszerre.

Az UUCP minden munkához fontosságának megfelelően, prioritási *fokozatot* rendel. A fokozatokat a 0–9, A–Z és a–z tartományokba eső karakterrel definíáljuk, ahol a legmagasabb prioritást a 0 jelenti. A levelek általában B vagy C prioritási fokozatúak, míg a hírek fokozata N. Az `ux` és az `uucp` parancs a–g kapcsolóval használható a kívánt prioritási fokozat megadására.

Az adatátvitel különböző protokollok szerin hajtható végre. Nem minden UUCP csomag érti az összes változatot. A g-protokoll a legrégebbi és a legelterjedtebb. A System V a g-protokollt részesít előnyben. A Taylor UUCP felismer egy igen gyors, kétirányú i-protokollt is. Ez azonban csak Taylor rendszerek között használható. A Taylor UUCP csomag több parancsból áll. Az alábbiakban röviden ezeket a parancsokat ismertetjük.

- **uucico** – egy másik UUCP rendszerrel létesít kapcsolatot, adatokat küld és fogad, majd ezután a kísérő parancsoknak megfelelő feldolgozást indítja el, ilyen parancs például az `rmail`, vagy az `uuxqt` parancson keresztül az `rnews`.
- **uuxqt** – az `ux` parancs által kért parancsokat hajta végre távoli rendszeren.
- **uucp** – adatok másolását teszi lehetővé tetszőleges UUCP rendszerek között.
- **ux** – adott parancsok végrehajtását teszi lehetővé távoli rendszeren. A parancs neve mellett a célcímen használni kívánt adatok is átvihetők.
- **uustat** – az aktuális UUCP munkákat sorolja fel. A még végre nem hajtott munkák eltávolításától a várlistáról.
- **uname** – az összes ismert helyi UUCP rendszert listázza.
- **uulog** – az UUCP naplójához tartalmát adja vissza. A listázás adott rendszerekre vagy vagy felhasználókra szűkíthető.
- **cu** – interaktív kapcsolatot létesít egy távoli UUCP rendszerrel.

## Konfigurálás

Az elmúlt évek során számos különböző UUCP rendszert fejlesztettek ki. Ezek elsősorban konfigurálásukban különböznek egymástól. A legkényelmesebbben használható UUCP változat az ingyenesen terjesztett **Taylor UUCP**. Ezzel nemcsak a kissé már elavult V2 és HDB konfigurálási mód használható, hanem saját, egyedi Taylor módja is. A régi konfigurációs fájlok egy konverterrel átalakíthatók ebbe a formátumba. A következőkben ezért csak a Taylor módban való konfigurálást tárgyaljuk. Feltételezzük, hogy a konfigurációs fájlok az `/usr/lib/uucp` könyvtárban találhatók. Ez az útvonal a fordításkor definíálható. Ezen a helyen a következő fájlok találhatók:

`config, sys, port, dialer`

A helyi rendszer neve a `config` fájlban található:

<code>nodename</code>	<code>nicetry.uucp.try.net</code>
<code>spool</code>	<code>/var/spool/uucp</code>
<code>pubdir</code>	<code>/var/spool/uucppublic</code>
<code>logfile</code>	<code>/var/lib/uucp/logfile</code>
<code>debugfile</code>	<code>/var/lib/uucp/debugfile</code>
<code>statfile</code>	<code>/var/lib/uucp/statistics</code>
<code>sysfile</code>	<code>/var/lib/uucp/taylor_config/sys</code>
<code>dialfile</code>	<code>/var/lib/uucp/taylor_config/dial</code>
<code>callfile</code>	<code>/var/lib/uucp/taylor_config/call</code>
<code>portfile</code>	<code>/var/lib/uucp/taylor_config/port</code>
<code>passwdfile</code>	<code>/var/lib/uucp/taylor_config/passwd</code>
<code>max-uuxqts</code>	<code>5</code>
<code>debug</code>	<code>abnormal</code>

A kapcsolat létrehozásához a helyi számítógépnek ismernie kell a szomszédos UUCP rendszert. Ez a `sys` fájjal érhető el, amely a rendszer neve mellett számos további adatot tartalmazhat, így a telefonszámot, a bejelentkezési nevet és a jelszót. A fájl például a következőképpen nézhet ki:

<code>success-wait</code>	<code>5</code>
<code>port</code>	<code>phone</code>
<code>call-login</code>	<code>*</code>
<code>call-password</code>	<code>*</code>
<code>system</code>	<code>nicetry.uucp.try.net</code>
<code>alias</code>	<code>nicetry</code>
<code>time</code>	<code>any</code>
<code>phone</code>	<code>94495112</code>

A paraméterek, például a port neve és átviteli sebessége, a különálló `port` fájlban kerülnek megadásra.

port	phone
type	modem
seven-bit	false
reliable	false
device	/dev/ttyS2
speed	38400
dialer	hayes
protocol	f,g

Ha a kapcsolat modernen keresztül jön létre, akkor ehhez a diaeler fájl is szükséges, amely a modemre vonatkozó adatokat tartalmazza.

dialer	hayes
chat	" ATZ OK ATDT CONNECT
chat-fail	BUSY
chat-fail	NO
complete	+++ ath
abort	+++ ATH

## Naplótájlok

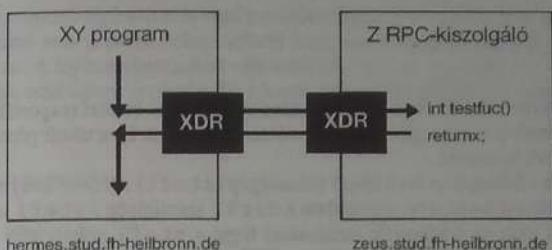
Az UUCP rendszer valamennyi műveletét több naplótájlról rögzíti. Ezek nagyon hasznosak lehetnek a konfiguráláshoz is. A /var/spool/uucp/Log fájl az általános protokolladatokat, míg a /var/spool/uucp/Stats fájl az elérő átviteli sebességek statisztikáját tartalmazza (láss még ulog). Ha az uucico indítása az -x parancssozi kapcsolóval hibakereső üzemmódban történik, akkor a hibakeresési adatok a /var/spool/uucp/Debug fájlba kerülnek.

## Automatikus kapcsolatok

A kapcsolat létrehozásnak automatizálására az UUCP felhasználónak a saját crontab táblázatát kell kibővítenie (lássd a 2.9 részben). A vonal foglaltágánál a cron démon a hívásmismétlést is egyszerűvé teszi. Ebben az esetben a sikeresen létrehozott kapcsolatot követő adott időtartamra az uucico parancsban érdemes kizárnai a végrehajtást. A következő példában a gallien rendszerrel naponta délután 7:00 és 7:15 között kapcsolatot létesítő crontab táblázatot mutatjuk be.

```
# If you don't want the output of a cron job mailed to you, you have to direct
# any output to /dev/null. We'll do this here since these jobs should run
# properly on a newly installed system, but if they don't the average newbie
# might get quite perplexed about getting strange mail every 5 minutes. :^)
# mail any output to uucp no matter whose crontab this is
MAILTO=uucp
# daily call to system 'nicetry' after 7:00pm
0,3,6,9,12,15 * * /usr/lib/uucp/uucico -s nicetry
```

Az újraindítási időtartamot ebben az esetben 15 percre kell állítani (`success=wait 900`). A crontab táblázat két további bejegyzése az UUCP naplófájlok lerövidítésére szolgál.



9.5 ábra. RPC művelet

## 9.10 RPC

Bizonyos hálózati szolgáltatások, például a Network File System (NFS), a távoli eljáráshívásokon Remote Procedure Call (RPC), alapulnak. Ez olyan mechanizmust jelent, amely egyedi rutinok végrehajtását teszi lehetővé a hálózati távoli számítógépen. Ebben a rendszerben az RPC kiszolgáló az alprogramokat biztosítja, míg az ügyfelek paramétereik használatával ezeket az alprogramokat hívják meg (lásd 9.5. ábra). Az RPC tehát egy különleges típusú kommunikációt jelent a processzek között a hálózaton keresztül.

A különböző processzor-architektúrájú számítógépek közötti adatcseréhez az RPC-vel együtt használják általában az eXternal Data Representation (XDR) gépfüggetlen adatábrázolást. A különböző processzorok között gyakran előforduló különbség például az egész számok ábrázolása. Az XDR leírás lehetővé teszi az RPC rutinok paramétereinek megadását, így ezek a megfelelő szubrutinnal átkonvertálhatók a gépfüggetlen formátum és az egyes gépek formátuma között.

Az `rpcgen` program segítségével válik lehetővé a rutinok és a belső RPC rutinok automatikus hívását. Ez az RPC rutinok formális leírásából C forráskódot generál az RPC kiszolgáló és az ügyfél számára.

Egy RPC kiszolgáló és egy másik számítógép közötti kapcsolat létrehozásához a gépen a `portmap` démont kell futtatni. Ez felismeri a rendelkezésre álló szolgáltatásokat, és az RPC hívást a hálózatról a megfelelő RPC kiszolgálóba továbbítja. A `portmap` démon által tárolt regisztrált programok adatai diagnosztikai célból az `rpcinfo` programmal kérhetők le.

## 9.11 NIS

Mivel egy hálózaton a bejelentkezési adatok, jelszók és más információk megfelelő nyilvántartása és kezelése meglehetősen összetett feladat, a Sun Microsystems kifejlesztette a Network Information System (NIS) rendszert. (A rendszert korábban *Yellow Pages*-nek, *YP*-nek hívták, de mivel ez bejegyzett véjjegy Angliában, lecseréltek a nevét – a lektor.) Ebben a felhasználók adatait, a rendelkezésre álló hálózati szolgáltatásokat és a rendszerkonfigurációkat nem az egyes gépeken, hanem egy központi NIS kiszolgálón tárolják és kezelik. Ha az adminisztrátor új felhasználót lépett be, vagy ha egy felhasználó megváltoztatja a jelszóját, a NIS gondoskodik az új információ eljuttatásáról az érintett gépekbe.

A NIS ügyfélcsiszolvert már számos kiadás tartalmazza, de FTP kiszolgálókról is letölthető, például az [ftp.uni-paderborn.de](http://ftp.uni-paderborn.de) kiszolgáló (Paderborni Egyetem, Németország) /pub/linux/local/yp könyvtárából.

### 9.12 NFS

A Network File System (NFS) protokoll más számítógépek által kiadott (exportált) fájlrendszerek csatlását teszi lehetővé a helyi gép saját fájlrendszeré részeként. Ez a távoli gépek könyvtárainak transzparens elérését biztosítja.

Az alábbi példa a hálózatban lévő távoli számítógép (`stef1`) /home könyvtárában lévő fájlokhoz való hozzáférést mutatja be. Kezdetben a `dir1` számítógép /`stef1` könyvtára üres. A betöltési folyamat után ez a könyvtár tartalmazni fogja a `stef1` számítógép /home könyvtárában lévő (lényegében) összes fájlt. A példa az NFS mount eljárásának használatát mutatja be.

```

nicetry:/dev# ls /
bin/          etc/          lib/          proc/
boot/         fa/           linux2001   root/
cdrom/        install/      linux20141  tmp/
dev/          kernel.cfg.old mnt/         var/
dosc/         kernel.cnf    netware/    vmlinuz
nicetry:/dev# ls mnt
nicetry:/dev# mount stef1:/home /mnt
nicetry:/dev# ls mnt
dirk/          fritz/        peter/       root/        stefan/
nicetry:/dev#

```

Példa az NFS mount használatára

Az NFS fejlesztője a Sun Microsystems. Mivel a Sun kiadta az NFS definícióit, számos más gyártó is integrálhatja operációs rendszerébe ezt a protokolt. Ily módon az NFS szabvánnyá vált, és szinte az összes platformon megtalálható, jóllehet magasabb szintű hatóság nem készítette ezt elő. Az NFS megtalálható a UNIX szinte összes változatában, az MS-DOS és sok más operációs rendszerben.

Az NFS lényeges jellemzője az *állapotmentes kiszolgáló (stateless server)*. Ez azt jelenti, hogy a könyvtárat exportáló NFS kiszolgáló az ügyfelekről semmiféle információt sem tárol, minden egyszerű irási és olvasási műveleteket hajt végre. Ha például egy NFS kiszolgálót bármilyen okból újra kell indítani miközben egy NFS ügyfél fájl másol a kiszolgáló által exportált könyvtárból, akkor az ügyfél másolási eljárása nem lesz megszakítva, hanem az ügyfél megvárja, míg a kiszolgáló újból válaszol, majd folytatja a másolási eljárást. Ez az eljárás akkor válik problematikussá, amikor szinkronizálni kell ugyanabba a fájlba írni kívánó több ügyfél tevékenységét.

Az NFS másik hátránya a biztonsággal, azaz a jogtalan hozzáféréssel kapcsolatos. Az újabban megjelenő fájkezelő rendszerek, például az Andrews File System (AFS) vagy a Distributed File System (DFS) ebben a kérdésben felülmúlja az NFS-t. A DFS az Open Software Foundation (OFS) Distributed Computing Environment (DCE) rendszerének a része. Az AFS és a DFS azonban még kevésbé elterjedt, mint az NFS.

A Linux is tartalmaz hálózati fájkezelő rendszert, de a megfelelő kezelőprogramokat a rendszermagba kell fordítani. Ezután a futtatás idején az NFS használatához a portmap, valamint az `rpc.nfsd` és az `rpc.mountd` démont kell futtatni. Az `rpc.nfsd` NFS démon az NFS

ügyfélgépekről érkező olvasási és írási kéréseket kezeli. Az `rpc.mountd` démon magáért a betöltési információkért felelős: kezeli a könyvtárakat, és letöltési kérések nél az ügyfél azonosságát ellenőrzi.

A démonokhoz tartozó legfontosabb konfigurációs fájl az `/etc(exports`, amely az NFS rendszeren keresztül más gépekre felkapcsolható könyvtárak listáját, valamint ezek jogosultsági adatait tartalmazza. A fájlból végrehajtott változtatások csak az `rpc.nfsd` és az `rpc.mountd` újraindítása után lépnek érvénybe. Ez a konfigurációs fájl például a következő lehet:

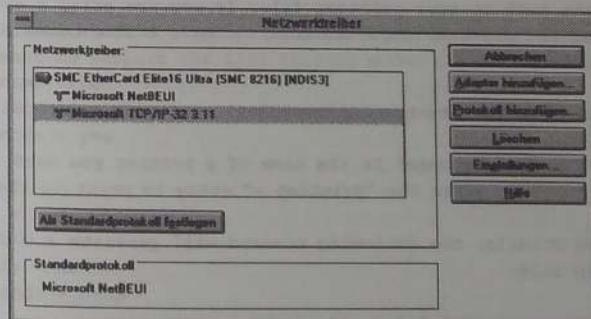
```
#  
# Exported directories  
#  
/      linux1(rw)  
/home   192.168.1.49(rw)  
/progs  risc1(rw)
```

## 9.13 LAN MANAGER

A PC-k területén gyakori hálózati protokoll a LAN Manager SMB protokollja. Az IBM és a Microsoft is kínál az OS/2, illetve a Windows NT alatti kiszolgálókat. E megoldások legnagyobb hátrányát a szoftverek ára jelenti. E probléma megoldására az Interneten egy projektet indítottak el egy UNIX-alapú LAN Manager kiszolgáló kifejlesztésére, amelyet Samba névre kereszteltek. Ez a szabadon letölthető kiszolgáló mind fájlkiszolgálóként, mind nyomtatás-kiszolgálóként használható.

Mivel a Samba a felhasználói szinten normál processzként fut, a rendszermag módosítása nem szükséges. A UNIX rendszermag azonban általában csak a TCP/IP protokolit ismeri fel, így a Samba sem kezeli a tényleges SMB protokolot. Ehelyett az egyedi SMB csomagokat egy IP csomagállalítja. Elvileg ezzel a módszerrel tetszőleges számú hálózati protokoll küldhető az IP hálózaton keresztül. Ezzel a trükkkel az útválasztásra egyébként alkalmatlan LAN-kezelő protokoll WAN hálózatokban is használható.

A TCP/IP vermet azonban az ügyfél oldalon telepíteni kell. Szerencsére a Microsoft egy FTP kiszolgálón (`ftp.microsoft.com`) ingyenesen biztosítja a DOS és Windows alatti megfelelő szoftvert. A TCP/IP verem a Windows 3.11-es verziójánál hálózati telepítésen keresztül áll rendelkezésre (lásd 9.6. ábra).



9.6 ábra. A TCP/IP verem a WfW 3.11-es verziójában

### A kiszolgáló (szerver) telepítése

Ha a Samba csomag forráskódként rendelkezésre áll, két démon, az smbd (LAN Manager kiszolgáló) és az nmbd (névkiszolgáló) fordítása szükséges. Először a helyi igényekhez konfigurálni kell a csomagot a Linux számára, majd a létrehozott programokat az /usr/sbin könyvtárba kell másolni, ahol a legtöbb egyéb hálózati démon is található. Mivel minden démon az Internet démonon keresztül is indítható, az /etc/inetd.conf fájlban elérhető a megfelelő bejegyzés.

netbios-ssn	stream	tcp	nowait	root	/usr/sbin/smbd	smbd
netbios-ns	dgram	udp	wait	root	/usr/sbin/nmbd	nmbd

Bejegyzés az /etc/inetd.conf fájlban

A kiszolgáló konfigurálása, ahogy ez általában a UNIX alatt szokásos, egy megfelelő ASCII fájl segítségével történik. A fájl elérési útvonalának megadása a fordításkor történik. A Samba csomaghoz mellékelt konfigurálási példa kiindulási alapként használható az egyéni beállítások végrehajtásához. Az itt rendelkezésre álló általános belépési lehetőség biztosítja az egyes felhasználóknak a UNIX oldali home könyvtár elérést, valamint a UNIX nyomtatási szolgáltatások használatát. A konfigurációs fájl itt következő részlete a kiszolgáló elérést biztosítja:

```
; Configuration file for smbd.
;

; For the format of this file and comprehensive descriptions of all the
; configuration option, please refer to the man page for smb.conf(5).
;

; The following configuration should suit most systems for basic usage and
; initial testing. It gives all clients access to their home directories and
; allows access to all printers specified in /etc/printcap.
;

; Things you need to check:
;

;

; 1: Check the path to your printcap file. If you are using a system that
; does not use printcap (eg., Solaris), create a file containing lines
; of the form
;

;     printername|printername|printername|
;

; where each "printername" is the name of a printer you want to provide
; access to. Then alter the "printcap =" entry to point to the new file.
;

; If using Solaris, the following command will generate a suitable
; printcap file:
;

;     lpc status | grep ":" | sed s/:/\// > myprintcap
```

```
; 2: Make sure the "print command" entry is correct for your system. This
; command should submit a file (represented by %s) to a printer
; (represented by %p) for printing and should REMOVE the file after
; printing.

; One most systems the default will be OK, as long as you get
; "printing =" right.

; It is also a good idea to use an absolute path in the print command
; as there is no guarantee the search path will be set correctly.

; 3: Make sure the "printing =" option is set correctly for your system.
; Possible values are "sysv", "bsd" or "aix".

; 4: Make sure the "lpq command" entry is correct for your system. The
; default may not work for you.

; 5: Make sure that the user specified in "guest account" exists. Typically
; this will be a user that cannot log in and has minimal privileges.
; Often the "nobody" account doesn't work (very system dependant).

; 6: You should consider the "security =" option. See a full description
; in the main documentation and the smb.conf(5) manual page

; 7: Look at the "hosts allow" option, unless you want everyone on the
; internet to be able to access your files.

[global]
printing = bsd
printcap name = /etc/printcap
load printers = yes
guest account = pcguest
; This next option sets a separate log file for each client. Remove
; it if you want a combined log file.
log file = /usr/local/samba/log.%m

; You will need a world readable lock directory and "share modes=yes"
; if you want to support the file sharing modes for multiple users
; of the same files
; lock directory = /usr/local/samba/var/locks
; share modes = yes

[homes]
comment = Home Directories
browseable = no
read only = no
create mode = 0750

[printers]
```

```

comment = All Printers
browseable = no
printable = yes
public = no
writable = no
create mode = 0700

; you might also want this one
; [tmp]
;   comment = Temporary file space
;   path = /tmp
;   read only = no
;   public = yes

;

; Other examples.

; A private printer, usable only by fred. Spool data will be placed in fred's
; home directory. Note that fred must have write access to the spool
; directory, wherever it is.
:[fredsprn]
;   comment = Fred's Printer
;   valid users = fred
;   path = /homes/fred
;   printer = freds_printer
;   public = no
;   writable = no
;   printable = yes
;

; A private directory, usable only by fred. Note that fred requires write
; access to the directory.
:[fredsdir]
;   comment = Fred's Service
;   path = /usr/somewhere/private
;   valid users = fred
;   public = no
;   writable = yes
;   printable = no
;

; A publicly accessible directory, but read only, except for people in
; the staff group
:[public]
;   comment = Public Stuff
;   path = /usr/somewhere/public
;   public = yes
;   writable = no
;   printable = no
;   write list = @staff
;

```

```

; a service which has a different directory for each machine that connects
; this allows you to tailor configurations to incoming machines. You could
; also use the %u option to tailor it by user name.
; The %m gets replaced with the machine name that is connecting.

:[pchome]
; comment = PC Directories
; path = /usr/pc/%m
; public = no
; writeable = yes
;

;
; A publicly accessible directory, read/write to all users. Note that all files
; created in the directory by users will be owned by the default user, so
; any user with access can delete any other user's files. Obviously this
; directory must be writable by the default user. Another user could of course
; be specified, in which case all files would be owned by that user instead.

:[public]
; path = /usr/somewhere/else/public
; public = yes
; only guest = yes
; writable = yes
; printable = no
;

;
; The following two entries demonstrate how to share a directory so that two
; users can place files there that will be owned by the specific users.
; In this setup, the directory should be writable by both users and should
; have the sticky bit set on it to prevent abuse. Obviously this could be
; extended to as many users as required.

:[myshare]
; comment = Mary's and Fred's stuff
; path = /usr/somewhere/shared
; valid users = mary fred
; public = no
; writable = yes
; printable = no
; create mask = 0765

```

### Egyszerű SMB kiszolgáló konfigurálása

A Samba konfigurációs fájl több **szakaszra** oszlik: mindegyik a kiszolgáló egy szolgáltatását definiálja. A **global**, a **homes** és a **printers** szakasz különleges funkciójú. Az egyes szakaszok **attribútumokat** és ezekhez rendelt értékeket tartalmaznak, amelyek a szolgáltatás jellemzőit adják meg. A globális paraméterek vagy az alapértelmezett értékek megadása a **global** szakaszban történik.

Ha a **homes** szakasz létezik, akkor olyan szolgáltatás jön létre, amely lehetővé teszi a kiszolgáló számára ismert felhasználóknak, hogy bejelentkezzenek egy ügyfélgépről, és hozzáférjenek a **home könyvtárakhoz**. Ez azt jelenti, hogy nem szükséges minden egyes felhasználó számára egyedi bejelentkezési szolgáltatást deklarálni. A kiszolgáló a bejelentkezéshez szükséges jelszót az

/etc/passw fájlból veszi. A path változó beállítása a felhasználó megfelelő home könyvtárára mutat.

A printers szakasz hasonló a homes szakaszhöz, de nem bejelentkezési, hanem nyomtatási szolgáltatásokat definiál. Ily módon az ügyfelek a rendszer számára ismert összes nyomtatónak (/etc/printcap) elérhetik. Különösen érdekes jellemző az, hogy a kiszolgálón PostScript nyomtatáv várolistája is rendelkezésre áll. Ezt a megfelelő lpr szűrő és a Ghostscript teszi lehetővé.

Ezzel a módszerrel a Linux gazdaságos Raster Image Processor (RIP) rendszerként működhet még akkor is, ha nem áll rendelkezésre PostScript nyomtatás. A nyomtatás előtt a fájlt a kiszolgálóra kell másolni, és ott a /var/spool/public könyvtárban található.

A következő táblázat a szolgáltatásoknál módosítható attribútumokat ismerteti. A listában megkülönböztetjük a globális (G) és szolgáltatás-attribútumokat (S). A megfelelő alapértelmezett beállítást zárójelben adtuk meg.

allow hosts GS azon ügyfelek listája, aik szolgáltatást használhatnak; a helyettesítő karakterek megengedettek: \*.fh-heilbronn.de, 192.0.2.\*

available	S	az ügyfél szolgáltatást használhat (yes)
copy	S	az elfőző szolgáltatás definíciójának másolata
create mask	GS	alapértelmezett állományvédelem maszkja új fájl létrehozásakor
create mask	G	szabványos állományvédelem maszkja új fájl létrehozásakor
dead time	G	inaktív kapcsolat megszakításának ideje percben
debug level	GS	hibakeresési szint
default services	G	szabványos szolgáltatás ismeretlen szolgáltatás kérésre
deny hosts	GS	szolgáltatás eléréséből kizárt gépek listája
dfree command	G	programrész a szabad memória megállapítására
dont descend	S	azon könyvtárak listája, amelyeknek üresként kell megjelenniük az ügyfél oldalon (none)
getwd cache	G	az aktuális könyvtár gyorsítótára (yes)
guest account	G	szabványos felhasználónév vendégszolgáltatások számára
guest ok	S	vendéghozzáférés engedélyezett bármely felhasználónak (no)
guest only	S	kizárolt vendéghozzáférés számára (no)
keep alive	G	kapcsolatfenntartó (keep alive) csomag küldése n percenten
lock directory	G	lock (zároló) fájlok cléresi útvonala
locking	G	zárolás vezérlése (yes)
lpq command	GS	elérési útvonal az lpq-n
mangled names	G	UNIX fájlnévek helyettesítése (yes)
map hidden	S	rejtett fájlokra az Execute bit 1-be állítása (no)
map system	S	rendszerfájlokra az Execute bit 1-be állítása (no)
max connections	G	az egyidejűleg aktív kapcsolatok maximális száma
max xmit	G	az átvitt csomagok maximális mérete
only user	GS	azt szabályozza, hogy csak a regisztrált felhasználók férhetnek-e rendszerhez
password level	G	a jelszóban a nagybetűk maximális száma
path	GS	a megfelelő szolgáltatások elérési útvonala
print command	GS	az átvitt nyomtatáfájlok kimeneti parancsa

print ok	S	nyomtató-hozzáférés engedélyezett (no)
printcap name	G	a printcap fájl elérési útvonala
printer name	S	az alapértelmezett nyomató neve
protocol	G	az aktuális protokoll verziószáma
read only	S	a szolgáltatás csak olvasásra elérhető (yes)
read prediction	G	prediktív olvasás engedélyezése
read raw	G	az adatokat nagy csomagokban olvassa
root directory	G	a kiszolgáló az átvitt könyvtáron chroot parancsot hajt végre
set directory	S	a felhasználó a könyvtárak módosítására használhatja a setdir parancsot
username	S	a szerver-használó neve
wide links	G	tetszőleges csatolás követése
write raw	G	írás nagy csomagokban

### SMB szolgáltatások attribútumai

A Samba kiszolgálóval egy PC-s ügyfél a kiszolgáló egyéb erőforrásait is egyszerűen használhatja, ilyen lehet például egy CD-ROM-egység vagy egy központi merevlemez.

### Az ügyfél (kliens) konfigurálása

Egy PC-s ügyfél egy Samba LAN-kezelő kiszolgálót hozzáférési útvonalakon a következő formában érhet el:

```
\<server name>\<service>
```

A kiszolgálón a /home/pcuser könyvtár eléréséhez az útvonalnak a következőképpen kell kinéznie:

```
\\\master\pcuser
```

Nyomtató elérése: például a phoenix kiszolgáló ps nevű nyomatóját így lehet elérni:

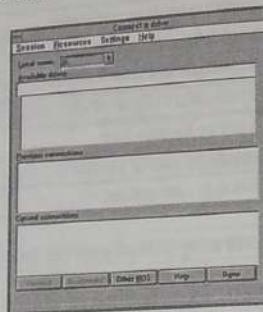
```
\\\phoenix\ps
```

A Windows for Workgroups rendszerben a kiszolgáló-kapcsolatok ilyen típusú konfigurálása a Filkezelőben vagy a Nyomtatókezelőben történik (lásd a 9.7. ábrát).

A kiszolgáló első ízben történő elérésekor be kell írni a megfelelő jelszót.

### 9.14 PC/NFS

Az NFS nem csak UNIX számítógépek összekapcsolására használható, hanem DOS operációs rendszert használó PC-k UNIX hálózatba integrálására is. Ehhez a kereskedelemben kapható számos termék mellett shareware csomagok is használhatók (`xfs/xfs32`). A kiszolgáló oldalon az `rpc.pcnfsd` is szükséges a szokásos `rpc.nfsd` és az `rpc.mountd` démonokon túl.



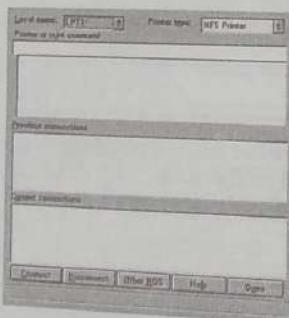
9.7 ábra. Könyvtár letöltése

Az előbbi elsőszorban a felhasználók azonosításához és a nyomtató kezeléséhez szükséges. A legtöbb Linux változat már tartalmazza ezeket. Futtatásukra az `rc.M` indítószkriptben kerül sor (lásd a 7.2. szakaszban). A nyomtatási feladatok könyvtára paraméterként kerül átadásra.

<code>/usr/sbin/rpc.pcnfsd</code>	<code>/var/spool/pcnfs</code>
-----------------------------------	-------------------------------

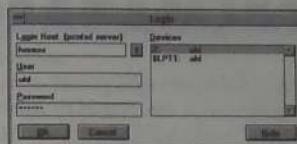
A PC/NFS protokoll segítségével a kiszolgáló számára ismert fájlok és nyomtatási várólisták érhetők el. A PC/NFS kiszolgáló hozzáférési jogosultságai, hasonlóan az NFS-hez, az `/etc/exports` fájlból adhatók meg (lásd a 9.12. szakaszban). A PC ügyfélén való telepítés után exportált könyvtár letöltése a Filekezelőben az `xfs32` új menüelem segítségével lehetséges.

A Linux nyomtatási várólistához való hozzáférés hasonló elven működik (lásd a 9.9 ábrát).



9.8 ábra. A Linux nyomtatási várólista elérése

Az azonosítás a letöltés során vagy később a Filekezelőben egy párbeszéddpanelen történik (lásd a 9.9 ábrát).



9.9 ábra. Azonosítás

Az azonosítási eljárás nélkül a felhasználó csak a közös könyvtárakhoz férhet hozzá, mivel a kiszolgáló oldal azonostójához a nobody érték kerül hozzárendelésre. A felhasználói hozzárendelés később bármikor módosítható.

## 9.15 COLUMBIA APPLETALK (CAP)

A Linux kiszolgáló szolgáltatásai nem korlátozódnak a szokásos PC-k integrálására. A Columbia Appletalk Packet (CAP) segítségével az Apple Macintosh számítógépek is kiszolgálhatók. A CAP számos változatot kínál az integrálásra. Ezek közül a legérdekesebb a közvetlen Ethershare-hoználat biztosítása.

A fájl- és nyomtatókezelés mellett a CAP hozzáférést biztosít az Apple Ethershare nyomtatóhoz is. A telepítéshez a Linux rendszermag 1.1.70-es vagy újabb verziója szükséges.

## 9.16 ISODE

Amint a 3.2. szakaszban leírtuk, megfelelő ISO szabvány készült a hálózatok felépítéséről és megvalósításáról. A TCP/IP már létezett, amikor ezt a szabványt megfogalmazták. Bár a TCP/IP megfeleltethető az ISO/OSI modellnek, de nem teljesen kompatibilis a szabvánnyal. Azok a fejlesztők, akik a Linux alatt OSI-kompatibilis alkalmazásokat szeretnének létrehozni, az ISODE nevű (készítője Marshall T. Rose) szabadon terjeszthető ISO/OSI vermet használhatják, amely illeszkedik a Linux-hoz. Ez többek között megtalálható a [sunsite.unc.edu](http://sunsite.unc.edu) kiszolgálón a /pub/Linux/system/Network/isode könyvtában.

## 9.17 NOVELL

A Novell Netware hálózatokra (IPX protokoll) való csatlakozás és Netware meghajtók transzparenst elérésének biztosítására számos megoldás született. A Linux alatt egyelőre nincs valódi Netware fájlkezelő rendszer Novell kiszolgáló letöltésére közvetlenül a Linux alá. A Novell kiszolgálók azonban elérhetők a DOS emulátorban (lásd a 4.1. szakaszban). Egy másik lehetőség a Novell kiszolgálóhoz NFS bővítműcsomag használata, amellyel a kiszolgáló az NFS-en keresztül exportálhatja könyvtárait.

(Ezt az angol eredeti könyvben leírt helyzetet szerencsére az idő és a programozók már jócskán túlhaladták. Az ncdfs csomaggal hozzáférhetünk IPX protokolloval Netware kiszolgálók köteteihez és nyomtatóihoz. A mars\_nwe csomaggal pedig netware kompatibilis állománykiszolgálót csinálhatunk linuxos gépünkből. A két csomag ugyan még a fejlesztés korai fázisában van – az ncdfs előbbre tart –, de már most is nagyon jól használható minőségűek – a lektor.)

# HÁLÓZATI ALKALMAZÁSOK

**A**z előző fejezetben a TCP/IP elméleti alapjaival és alsóbb protokolszintekkel ismerkedhetünk meg. Most az alsóbb szinteket használó programok és kiszolgálók tárgyalására kerül sor.

## 10.1 HÁLÓZATI DÉMONOK

Ahogy korábban már említettük, az operációs rendszermag a TCP/IP kiszolgáló szolgáltatásainak többségét nem tartalmazza. Ezek megvalósítására külön démonok szolgálnak. Ezek között a Telnet, az FTP és az e-mail kiszolgálók is megtalálhatók. Az alábbi lista a Linux alatt elérhető legfontosabb hálózati démonokat tekinti át. A változattól függően a névben az `rpc`, vagy `in.` előtag is szerepelhet, ezeket a felsorolásban nem tüntettük fel.

- **bootpd** – lemezegység nélküli munkaállomások és X terminálok elindításához szükséges.
- **fingerd** – a (másik) rendszeren aktív más felhasználók vizsgálatát teszi lehetővé.
- **ftpd** – adatok átvitelére szolgál az egyik rendszerből egy másikba az FTP protokollal.
- **gated** – irányítási protokoll megvalósítása dinamikus útválasztáshoz.
- **httpd** – WWW kiszolgáló démon.
- **identd** – User Identification Server (felhasználó-azonosító kiszolgáló), az RFC1413-ban definiált protokoll szerint felhasználó azonosítására szolgál.
- **imapd** – imap kiszolgáló; imap ügyfelekkel (például `pine`) rendelkező postafiókok előrére használt.
- **pop2d** és **pop3d** – kiszolgáló a levelek eléréésében használt POP2 és POP3 protokolloz.
- **lpd** – nyomtatódémon, távoli gépről nyomtató elérésének lehetőségét is tartalmazza.
- **mountd** – másik számítógép fájlrendszerének használatát teszi lehetővé a helyi rendszeren.
- **nfsd** – az adatokat NFS kiszolgálóként bocsátja rendelkezésre.
- **nmbd** – Netbios névkiszolgáló; a DNS névkiszolgálóhoz nincs köze.
- **nntpd** – hálózati hírek elosztását/olvasását segítő protokoll.
- **ntalkd** – kiszolgáló egy `talk` változat számára.
- **pcnfsd** – kiszolgáló a PC/NFS számára; rendszerfájlok és -nyomtatások elérést teszi lehetővé PC-k számára.
- **pppd** – démon a PPP protokoll számára (lásd a 9.5. szakaszban).
- **rlogind** – az `rlogin` parancs használatával távoli rendszerről teszi lehetővé a bejelentkezést.
- **routed** – a dinamikus útválasztást végzi; használható a gated helyett is.
- **rplayd** – kiszolgáló a hanglejátszára használt `rplay` parancs számára.
- **rshd** – parancs végrehajtását teszi lehetővé távoli rendszerről.
- **rstatd** – az `rstat` kiszolgálója; a rendszermag statisztikai adatait szolgáltatja.
- **rusersd** – kiszolgáló az `rusers` parancs számára; a bejelentkezett felhasználókról szolgáltat információt.
- **rwalld** – kiszolgáló az `rwall` parancs számára; felhasználóknak szóló hirdetményeket tesz közöz.

- **rwhod** – kiszolgáló az rwho parancs számára; a bejelentkezett felhasználóról gyűjt és szolgáltat információt.
- **sendmail** – levelezést küld és fogad a hálózaton; helyette az smail is használható.
- **smail** – levelezést küld és fogad a hálózaton. Fontos továbbá a közvetlen IP-kapcsolat nélküli (például az UUCP) rendszerekben.
- **smbd** – az SMB/LAN manager kiszolgálója (lásd a 9.13. szakaszban).
- **talkd** – interaktív kommunikációt biztosít más felhasználókkal a talk parancs segítségével.
- **tcpd** – TCP démon. Az aktuális kiszolgálókat tcpd-n keresztül hívják az ügyfelek címének és jogosultságainak ellenőrzése céljából.
- **telnetd** – hasonló, mint az rlogind; felhasználók távoli rendszerről való bejelentkezését teszi lehetővé.
- **tftpd** – a hálózat más számítógépeinek elindítását teszi lehetővé (mint a bootp.d).
- **timed** – időszinkronizáló démon; a helyi számítógép időbeállítását hozza összhangba a hálózat többi gépével.
- **xntpd** – egy másik időbeállító démon; az RFC135 szerinti NTP protokollt valósítja meg.

## 10.2 AZ INTERNET SZUPERSZERVER (INETD)

A legtöbb TCP/IP démon aktivizálására csak egy másik számítógépről érkező megfelelő szolgáltatás kérése esetén kerül sor. Ha ezek a háttérben állandóan aktívak lennének, feleslegesen foglalnák a memóriát és a gépidőt. Ezért a rendszer elindításakor ezen démonok automatikus indítása nem történik meg.

A UNIX rendszerekben általában rendelkezésre áll egy **Internet démon** (Internet szuperkiszolgáló), amely szemben a többi hálózati démonnal, állandóan aktív, és várja a hálózatról érkező üzeneteket. minden szolgáltatáshoz rögzített portszám tartozik. A szolgáltatások jegyzékét és a megfelelő portszámokat az /etc/services fájl tartalmazza. Egy másik fájlból (/etc/inetd.conf) a kért szolgáltatáshoz tartozó megfelelő démonok listája található. Az inetd csak lényeges kapcsolatkérésnél indítja el a megfelelő démont, amely átvesszi a kapcsolat kezelését. A kapcsolat lezárása után a kérdéses démon is befejezi működését, míg az inetd továbbra is aktív marad.

```

#
# Network services, Internet style
#
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1340, 'Assigned Numbers' (July 1992). Not all ports
# are included, only the more common ones.
#
#      from: @(#)services      5.8 (Berkeley) 5/9/91
# $Id: services,v 1.9 1993/11/08 19:49:15 cgd Exp $
#
tcpmux          1/tcp      # TCP port service multiplexer
echo            7/tcp
echo            7/udp
discard         9/tcp      sink null
discard         9/udp      sink null

```

		users
sysstat	11/tcp	
daytime	13/tcp	
daytime	13/udp	
netstat	15/tcp	
quotd	17/tcp	quote
msp	18/tcp	# message send protocol
msp	18/udp	# message send protocol
chargen	19/tcp	ttytst source
chargen	19/udp	ttytst source
ftp	21/tcp	
# 22 - unassigned		
telnet	23/tcp	
# 24 - private		
smtp	25/tcp	mail
# 26 - unassigned		
time	37/tcp	timserver
time	37/udp	timserver
rip	39/udp	resource # resource location
nameserver	42/tcp	name # IEN 116
whois	43/tcp	nickname
domain	53/tcp	nameserver # name-domain server
domain	53/udp	nameserver

## Részlet egy /etc/services fájlból

A fenti fájlok módosítására csak ritkán kerül sor: ez új szolgáltatások felvételekor vagy egy démon frissítések szükséges.

```

#
# See "man 8 inetd" for more information.
#
# If you make changes to this file, either reboot your machine or send the
# inetd a HUP signal:
# Do a "ps x" as root and look up the pid of inetd. Then do a
# "kill -HUP <pid of inetd>".
# The inetd will re-read this file whenever it gets that signal.
#
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
#
# Echo, discard, daytime, and chargen are used primarily for testing.
echo      stream    tcp    nowait    root    internal
echo      dgram     udp    wait      root    internal
discard   stream    tcp    nowait    root    internal
discard   dgram     udp    wait      root    internal
daytime   stream    tcp    nowait    root    internal
daytime   dgram     udp    wait      root    internal
chargen   stream    tcp    nowait    root    internal
chargen   dgram     udp    wait      root    internal

```

```

time      stream   tcp    nowait   root    internal
time      dgram    udp    wait     root    internal

#
# These are standard services.
ftp       stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/wu.ftpd
www       stream   tcp    nowait   httpd   /usr/local/sbin/httpd          httpd
#www      stream   tcp    nowait   root    /usr/local/etc/httpd/httpd      httpd
telnet    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.telnetd
#
# Use this one instead if you want to snoop on telnet users (try to use this
# for ethical purposes, ok folks?):
#telnet   stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.telnetsnooopd
#
# The line below is set up for running Smail:
#smtp    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/bin/rsmtp -bs
#
# If you want to read NNTP news via TERM, comment out the nntp
# line below, and use a command like this once the TERM
# connection is up: tredir 119 my.nntp.host:119
# You'll also want to do this: set NNTPSERVER my.nntp.host ; export
NNTPSERVER
ntp      stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.nntpd
#
# Shell, login, exec and talk are BSD protocols.
shell    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.rshd
login    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.rlogind
#exec    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.rexecd
talk     dgram    udp    wait     root    /usr/sbin/tcpd /usr/sbin/in.ntalkd
ntalk    dgram    udp    wait     root    /usr/sbin/tcpd /usr/sbin/in.ntalkd
#
# Pop mail servers
#pop2    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.pop2d
pop3    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/sbin/in.pop3d
pop1    stream   tcp    nowait   root    /usr/sbin/tcpd /usr/local/bin/
                                         /popper.linux
#
# The Internet UUCP service.
uucp    stream   tcp    nowait   uucp   /usr/sbin/tcpd /usr/lib/uucp/uucico -l
#
# Tftp service is provided primarily for booting. Most sites
# run this only on machines acting as "boot servers."
# Since these can be security holes, they are commented out by default.
tftp    dgram    udp    wait     root    /usr/sbin/tftpd tftpd /usr/local/ixp
bootps  dgram    udp    wait     root    /usr/local/bin/bootpd bootpd -i

```

### 10.3 TELNET

A Telnet az egyik legrégebbi protokoll az Interneten. Segítségével a felhasználók a hálózatban lévő más számítógépekre jelenkezhetnek be. Ez azt jelenti, hogy az ügyfélgép billentyűzetén végrehajtott összes bevitel a kiszolgálóba kerül, és a kiszolgáló valamennyi képernyőadata az ügyfélgépen is megjelenik. Ily módon az ügyfél virtuális terminált szimulál.

A Linux és a legtöbb más UNIX változat alatt a Telnet megvalósítása a `telnetd` kiszolgáló démon és a `telnet` ügyfélprogram segítségével történik. A Telnet ügyfélprogramok a legtöbb egyéb operációs rendszer (még a DOS) alatti is rendelkezésre állnak.

Egy másik számítógépre való bejelentkezéskor a felhasználó a programot egy **számítógépnév**-vel vagy egy **IP címmel** indítja. Ezután a Telnet démon segítségével létrejön a TCP kapcsolat a megnevezett számítógéppel.

```
bucipc:~$ telnet sol
Trying 192.162.135.50...
Connected to sol.
Escape character is '^]'.

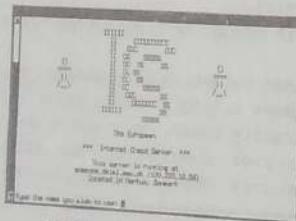
UNIX(r) System V Release 4.0 (sol)

login:
```

A Telnet program a kapcsolat létesítéséhez csak a Telnet démon portszámát használja. A portszám a Telnet hívásakor a gépnév után adható meg. Például a `telnet <gépnév> 25` azt jelenti, hogy a felhasználó a számítógép 25-ös portjával létesíthet kapcsolatot. A port az e-mail protokoll (SMTP) számára fenntartott. A megfelelő portszámmal a felhasználó a hírcsoport NNTP démonjával is kapcsolatot létesíthet (119-es port). Ennél a módszermel a `help` parancs segítségével a kiszolgáló démon protokollparancsainak leírása is lekérhető. Ez a hibakeresésnél lehet hasznos.

Néhány kiszolgáló megfelelő portjain játékok is rendelkezésre állnak (többrészvevős kalandjátékok, sakktér., lásd a 12.9 szakaszban). Ezek elérésének egyetlen feltétele a gépnév és a megfelelő portszám ismerete. Például az Internet sakkkiszolgáló (Internet Chess Server, ICS) az `anemone.daimi.aau.dk` gép 5000-es portszámán található. A kiszolgáló használatához ezt kell megadni: `telnet anemone.daimi.aau.dk 5000`.

Az `xboard` programmal, amely egyébként a GNU sakkkprogram grafikus előlétprocesszora, létrehozható ilyen Telnet kapcsolat, és ez emellett a grafikus megjelenést is elvégzi.



10.1 ábra. Az ICS kiszolgáló

A fentiekhez hasonló Internet szolgáltatások forrásainak listája a különböző FTP kiszolgálókon `internet.services` vagy `internet.resources` név alatt található. Ilyen listák a FAQ (frequently asked questions, gyakran felmerülő kérdések) kiadványok mellett a `news.answers` hírcsoportban is rendszeresen megjelennek. Különösen az Internet új felhasználói számára lehet hasznos egy hírolvasó rendszer megismerése, hiszen ez megkönnyíti az egyéb kiszolgálók és információforrások elérést is (lásd a 10.8. szakaszat).

## 10.4 FTP

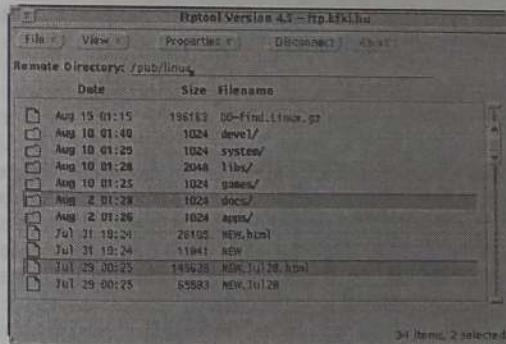
Az FTP (File Transfer Protocol) az Interneten keresztül adatok átvitelére szolgáló protokoll. Az FTP megvalósítása a Linux és az egyéb UNIX változatok alatt általában az FTP kiszolgáló démonnal (`ftpd`) és az `ftp` ügyfélprogrammal történik. Az egyszerű `ftpd` démon helyett gyakran használt Washington Egyetem `wu-ftpd` démonja is. Utóbbi főleg az adatbiztonság és a konfigurálhatóság területén tartalmaz újdonságokat.

A köznapi szóhasználatban az FTP kiszolgáló fogalma nem az `ftpd` kiszolgáló démont jelenti, hanem egy olyan számítógépet, amely az FTP-n keresztül anonim hozzáférést biztosít. A nagy FTP kiszolgálók általában több gigabájtos merevlemezrel rendelkeznek, és a szabadon letölthető, ingyenesen terjeszthető és shareware programok széles választékát kínálják.

Ha a Finnországban található legfontosabb **FTP kiszolgálóval** szeretnénk kapcsolatba lépni, ezt kell beírnunk: `ftp ftp.funet.fi`. Ez elindítja az `FTP` programot, amely a másik számítógéppel TCP kapcsolatot kísérel meg létrehozni az `FTP` démon 21-es portszámán keresztül. Ha a kapcsolat létrejött, a vendégnél felhasználói azonosítót kell bevinnie. A szabadon elérhető `FTP` kiszolgálóknál a vendégek az `ftp` vagy az `anonymous` felhasználónévet adhatják meg, míg jelszóként saját e-mail címüket kell beírni. Ezután a felhasználók az `ftp` program parancsaival (például `cd`, `dir`, `get` vagy `put`) fájlokat kereshetnek és töölhetnek le.

A fenti parancsok nem felelnek meg az **FTP protokoll** belső parancsainak, de az `FTP` program felismeri és átalakítja őket a megfelelő protokoll-parancsokkal (például `port` vagy `retr`). Csak az ilyen protokoll-parancsok átvitelére kerül sor. Adatátvitelnél egy újabb TCP kapcsolat jön létre az adatok számára.

Egyedi parancsok találhatók az `ftp` kézikönyv oldalon vagy az `ftp` programhoz tartozó súgófájlokban, amelyek a `help` beírásával hívhatók. A legfontosabb Linux `FTP` kiszolgálók listáját az 5.2. szakasz tartalmazza. A következő példa egy `FTP` futtatást mutat be:



10.2 ábra. ftptool

```

nicetry:~$ ftp localhost
Connected to localhost.
220 nicetry FTP server (Version wu-2.4(1) Tue Aug 8 15:50:43 CDT 1995) ready.
Name (localhost:demo): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password: my_email@address
230-Welcome, archive user! This is an experimental FTP server. If have any
230-unusual problems, please report them via e-mail to root@nicetry
230-If you do have problems, please try using a dash (-) as the first character
230-of your password -- this will turn off the continuation messages that may
230-be confusing your ftp client.
230-
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 9
drwxrwxr-x  8 root    wheel        1024 Aug  1 1994 .
drwxrwxr-x  8 root    wheel        1024 Aug  1 1994 ..
drwxrwxr-x  2 root    wheel        1024 Dec  3 1993 bin
drwxrwxr-x  2 root    wheel        1024 Aug 30 1993 etc
drwxrwxr-x  2 root    wheel        1024 Dec  3 1993 incoming
drwxrwxr-x  2 root    wheel        1024 Nov 17 1993 lib
drwxrwxr-x  3 root    wheel        1024 Jul 22 18:56 pub
drwxrwxr-x  3 root    wheel        1024 Aug 30 1993 usr
-rw-r--r--  1 root    root         312 Aug  1 1994 welcome.msg
226 Transfer complete.
ftp>

```

A fájlok átvitеле kényelmesebben is végrehajtható, ha a parancssoros `ftp` program helyett valamelyik grafikus előfeldolgozót használjuk. A grafikus előfeldolgozók számos változata rendelkezésre áll a szokásos Linux kiszolgálókon vagy az X11-es programok hivatalos kiszolgálóján (`ftp.x.org`). Az XView-alapú `ftptool` is népszerűvé vált.

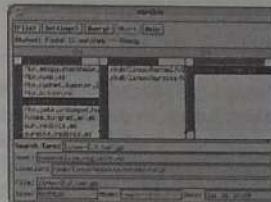
## 10.5 ARCHIE

A nagyszámú FTP kiszolgálón rendelkezésre álló halászat mennyiségi ingyenes szoftver esetén meglehetősen nehéz megtalálni az adott szoftvert tároló FTP kiszolgáló nevét és a könyvtár nevét. Ezen a területen az Archie kiszolgálók hasznos segítséget jelenthetnek. Az Archie kiszolgálók több gigabájtost adatházist tartalmaznak, amely az Interneten található legfontosabb FTP kiszolgálók tartalomjegyzékét foglalja magába. Az adatházist rendszeres időközönként frissítik. Archie kiszolgálók a következő címeken találhatók:

- `archie.th-darmstadt.de` (Németország)
- `archie.funet.fi` (Finnország)
- `archie.ans.net` (New York)

- archie.au (Ausztrália)
- archie.doc.ic.ac.uk (Nagy-Britannia)

Az Archie kiszolgáló lekérdezéséhez a telnet segítségével be kell jelentkezni a megfelelő gazdagépre archie felhasználónévvel, majd egy egyszerű lekérdezőnyelvvel kereshetők a programok.



10.3 ábra. Fájlereresés az xarchie segítségével

Az xarchie grafikus előfeldolgozó, amely a Linux alatt is rendelkezésre áll, kényelmesebb keresést biztosít. A felhasználónak csak a kulcsszót kell megadnia, majd a keresési üzemmódot beállítania és az xarchie létrehozza a kapcsolatot a már beállított kiszolgálóval. Hamarosan egy tallázóban megjelennek a fájlok és feldalálási helyük. Az xarchie újabb verziójában a megfelelő FTP kiszolgálóról a kívánt fájlok közvetlen lekéréséhez FTP átvitel is elindítható.

A programok név szerinti keresése mellett az Archie kiszolgálók egy what is (mi ez) adatbázist is tartalmaznak, amelyben a what is parancssal a kívánt programok rövid leírása jeleníthető meg. Ha a keresett program neve részben vagy teljesen ismeretlen, a what is segítségével az összes regisztrált program megfelelő részhalmaza jeleníthető meg név szerint rövid ismertetővel együtt.

Az Interneten keresztüli elérés helyett az Archie kiszolgálók e-mail segítségével is lekérdezhetők. Itt a megfelelő parancs kerül elküldésre az ilyen kiszolgálón lévő archie nevű felhasználónak. Az FTP levelezési kiszolgálókkal megegyezően, ha a levél első sorában a help parancsot küldik az Archie kiszolgálóba, részletesebb leírás érkezik.

## 10.6 A BERKELEY R-SEGÉDPROGRAMOK

A Kaliforniai Egyetemen Berkeley-ben kifejlesztett **BSD UNIX** nagy jelentőségű volt a UNIX hálózati jellemzőinek fejlődésében. A TCP/IP integrálása a UNIX operációs rendszerbe hatalmas lökést adott a TCP/IP további elterjedésének.

A Berkeley r-segédpogramokat olyan programok alkotják, amelyek neve r betűvel kezdődik. Közülük a legfontosabbak az rlogin, az rsh és az rcp. Az r betű a remote (távoli) szóra utal. A Berkeley r-segédpogramok a hálózatba kapcsolt munkaállomások szabványos szoftveréhez tartoznak. Az idők során a csoport számos programmal bővült (például rwho és ruptime). Számos UNIX változathoz hasonlóan a Berkeley r-segédpogramokat a Linux is tartalmazza.

A segédpogramok alapgondolata az, hogy a hálózatban több számítógépre bejelentkezési joggal rendelkező felhasználóknak, a jelsz minden egyes alkalmmal való beírása nélküli, egyszerű módszert biztosítson a hálózat többi számítógépére való bejelentkezésre vagy fájlok másolására. Ennek véghajtása, az adatbiztonság sértése nélkül, úgy történik, hogy más számítógépek adott felhasználóiit megbízható felhasználóként definiáljuk. Ezután csak ezek a felhasználók jeleníthetnek be jelsz megadása nélkül a számítógépre, és használhatják szolgáltatásait. A teljes rendszerre kiterjedően ez a definíció az /etc/hosts.equiv vagy a megfelelő felhasználók home könyvtárában az .rhosts fájlból a saját bejelentkezési adatoknál található.

```
# Accepted hosts and users
#
zeus.demo.de
hermes.demo.de peter
sun.demo.de arnlod
```

Az /etc/hosts.equiv fájlt csak a rendszer adminisztrátora változtathatja meg. Ha egy felhasználó egy másikhoz hozzáférési jogosultságot szeretne szerezni, ezt a megfelelő home könyvtárban lévő .rhosts fájlba való bejegyzéssel lehet megtenni. Az ilyek típusú fájlok különösen akkor hasznosak, amikor egy felhasználó eltérő azonosítót használ a hálózat különböző gépein. A megfelelően felépített .rhosts fájlokkal a felhasználók saját könyvtárihoz való hozzáférés a teljes hálózatban lényegesen egyszerűbb történik.

A fentartott portok használata az adatbiztonságot növeli. Ennél a kiszolgáló ellenőrzi az ügyfél TCP portszámát: ha nem privilegizált portról van szó, akkor a kiszolgáló megtagadja a hozzáférést. A UNIX gépeken a fentartott portok csak a superuser hozzáférési jogosultságokkal rendelkező felhasználók számára állnak rendelkezésre. Ezzel megakadályozható, hogy egy normál felhasználó hamis számítógép- vagy felhasználónál másik számítógéphez hozzáférjen. Ez azonban azt is jelenti, hogy ehhez a szolgáltatáshoz az r-szegédprogramokat a root felhasználó-azonosítóval kell futtatni.

A Berkeley r-szegédprogramok újabb változataiban a Kerberos-rendszer is használható, amely szintén az adatbiztonság növelésében játszik szerepet. A jelszót és felhasználói azonosítót ellenőrző Kerberos-rendszer a massachusettsi műszaki egyetemen (Massachusetts Institute of Technology, MIT) fejlesztették ki.

### rlogin

A felhasználó számára az rlogin távoli bejelenkező program működése a telnet programéhoz hasonlít. A különbség annyi, hogy megfelelő konfigurálásnál az előbbihez nem szükséges a felhasználói azonosító és a jelszó. Az rlogin nem teszi lehetővé eltérő portszám megadását.

```
nicetry:~$ rlogin zeus
zeus:~$
```

### rcp

Az rcp fájlok másolását teszi lehetővé a számítógépek között. A művelethez a pontos elérési útvonalakat ismernie kell. Éppen ezért a felhasználók számos esetben inkább az interaktív ftp programot vagy az NFS-en (Network File System) kereszttüli hozzáférést választják. Az FTP-vel szemben az rcp hierarchikus fájlrendszerek alágainak rekurzív másolását is lehetővé teszi (az újabb ftp ügyfélprogramok már tudnak rekurzívan másolni).

```
nicetry:~$ ls
Disktools/ Help/ News/ demo.tex filecount*
Documents/ Motif/ UsrAdmin/ demo.txt rc*
nicetry:~$ rcp -r peter@zeus:/home/prog/xprog .
```

```
nicetry:~$ ls
Disktools/ Help/
Documents/ Motif/
xprogs/
nicetry:~$
```

**rsh**

Az rsh távoli héjprogram programok végrehajtását teszi lehetővé más számítógépeken. A végrehajtandó parancs mellett a távoli gép nevét és esetleg a felhasználói azonosítót kell megadni. A végrehajtás után az eredmények a hálózaton keresztül a helyi gépre kerülnek.

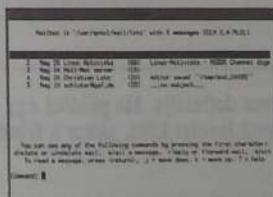
A programok más gépeken való végrehajtása mellett ez a program a számítógépek közötti gyors adatátvitelre is használható. Itt az rsh program azon jellemzője használható ki, hogy a helyi gép saját szabványos be- és kimenete kombinálható azzal a programmal, amely ezt a célszámítógépen végrehajtja. Ezzel például elkészíthető egy helyi merevlemez fájljainak biztonsági másolata a hálózat másik számítógépén lévő szalagos egységre.

```
nicetry:~$ rsh zeus "ls .em*"
.emacs
.emacs-bkmkrs
.emacs-places
.emacs-skp
nicetry:~$ rsh zeus " tar cfz - .em*" | tar xvzf -
.emacs
.emacs-bkmkrs
.emacs-places
.emacs-skp
nicetry:~$
```

## 10.7 LEVELEZÉS

A TCP/IP e-mail szolgáltatása általában egy démonból, amely az SMTP protokoll szerint üzeneteket továbbít a többi számítógépre, és a levelek frására és olvasásra szolgáló programokból áll. Ezeket levélolvasóknak is nevezik.

A Linux alatt a levelek továbbítására mind a népszerű sendmail, mind ennek alternatívája, az smail program is rendelkezésre áll. A TCP/IP hálózaton kereszttüli átvitel mellett ezek az UUCP protokollnál is használhatók.



10.4 ábra. Az elm program

A legtöbb csomag a pine és az elm levélolvasót is tartalmazza. Ezek mellett grafikus programok is rendelkezésre állnak (például a mutmail), amelyek X11 alatt a levelezés egyszerű kezelését teszik lehetővé.

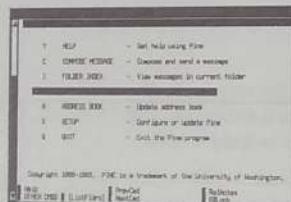
### elm

Az elm meglehetősen régi, de széles körben elterjedt, levelek írására és olvasására szolgáló program (lásd a 10.4. ábrát). A legtöbb UNIX platformon megtalálható. Az elm első elindításakor két alkönyvtárat hoz létre a felhasználó könyvtárában: egy konfigurációs könyvtárat és egyet a beérkező levelek tárolására. Egy listában az új levelek tekintethetők meg feladójukkal és az üzenet a címével. A kurzormozgató billentyűkkel ezek kiválaszthatók, majd elolvashatók és fájlként tárolhatók.

Levelek frászához a felhasználó különböző programot adhat meg az elm beállításai között. Az elm a Multipurpose Internet Mail Extension (MIME) formátumban lévő leveleket is kezeli tudja. Ennél a levél grafikákat, hangot vagy programokat is tartalmazhat. Az ilyen MIME levelek a metamail és a megfelelő kezelőprogrammal jeleníthetők meg.

### Pine

A pine, amely a „Pine Is No longer Elm” („A Pine egyáltalán nem Elm”) vagy a Program for Internet News & E-mail rövidítése is lehet, több kényelmet biztosít, mint az elm. Az egyik leglénnyegesebb különbség, hogy a pine lehetővé teszi az USENET News elérését (lásd a 10.8. szakaszban). A pine tartalmaz egy pico nevű szerkesztőt, amely jelentősen megkönnyíti a levél megrázását. Mivel a pine az IMAP2 protokoll használja, a távoli gépen lévő postafiókokat is kezeli tudja. Ez különösen az olyan felhasználók számára érdekes, akik több különböző számítógépen dolgoznak. Az IMAP2 egyszerű biztosítja a home gép levelezéséhez való hozzáférést.



10.5 ábra. A Pine levelezési előfeldolgozó

A pine konfigurálásához a felhasználó home könyvtárában található .pinerc fájl használható. Az összes felhasználó beállításai a pine.conf fájlból tárolhatók, amely általában az /usr/local/lib könyvtárban található. Az alábbiakban néhány fontosabb beállítást mutatunk be:

```
user-domain=try.net
```

Ez a kimenő levelekben a körzetnevet definiálja. Ha például egy mueller nevű felhasználó a hermes.demo.de számítógépről levelet küld, akkor a feladó címe a fenti beállítás alapján mueller@demo.de lesz.

```
inbox-path=mail/inbox
```

Az inbox-path beállítás a bejövő levelek mappájának elérési útvonalát adja meg. Ez a mappa általában egy olyan fájl, amely a /var/spool/mail könyvtárban a felhasználó nevét tartalmazza. A Pine programon belül a mappa neve INBOX. Ha más könyvtárba érkezik levél, vagy ha egy program, például a deliver, az üzeneteket automatikusan különböző fájlokba rendezi, a fájl elérési útvonala módosítható.

```
incoming-folders=Linux mail/linux,  
Project mail/project,  
Zeus zeus.demo.de
```

Az incoming folders sorban további postafiókok írhatók be. Ezek a mappák lehetnek más helyi fájlok, vagy más számítógépeken lévő postafiókok. minden mappához cím és útvonal tartozik. Ha egy postafiók másik számítógépen található, akkor az útnév elő szögletes zárójelben a számítógép neve kerül. Ebben az esetben a postafiókok az IMAP2 protokollon keresztül érhetők el.

A feature-list változóval a pine működése befolyásolható. A lehetséges beállítások a .pinerc fájl kommentárijaiban olvashatók. Ezek közül gyakran használt az old-growth és az auto-move-read-msgs. Utóbbi a pine bezárásakor az elolvast leveleket automatikusan áthelyezi a read-messages mappába.

```
feature-list=oldgrowth, aut-move-read-msgs
```

Az is megadható, hogy a pine melyik programot indítsa el a MIME levelek grafikáinak megtekintéséhez.

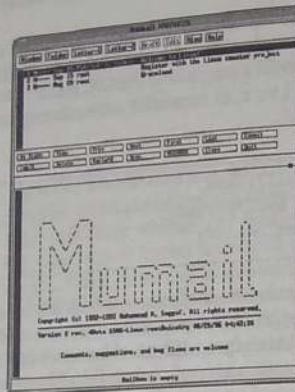
## MuMail

A mu mail grafikus interféssel rendelkező program (lásd a 10.6. ábrán). A program az X11 felhasználói grafikus interfészét használja, és kezelése az egérrrel kényelmesen végrehajtható. Jellemzői közelítőleg azonosak az elm programével. A MuMail multimédiás levelek létrehozására és megtekintésére is használható.

## deliver

A levelezés automatikus rendezésére számos program áll rendelkezésre, és ezek közül a legelterjedtebb a deliver és a pocmail. A deliver példáján bemutatjuk az ilyen típusú programok működését és konfigurálásának módját.

Mindenek előtt a deliver programnak kezelnie kell a bejövő leveleket. A leveleit rendezni kívánó felhasználó a .forward fájlból tett bejegyzéssel átadhatja ezeket a deliver programnak. A fájl tartalma például a következő lehet:



10.6 ábra. A MuMail levelező program

```
| /usr/bin/deliver <username>
```

Az is megtehető, hogy a programot közvetlenül beépítjük a levélátviteli démon konfigurációjába. Ennek leírása az smail konfigurálásánál, a 10.7. szakaszban található.

Ezt követően a deliver minden elindul, valahányszor levél érkezik, és az üzenet a deliver szabványos bemenetén lesz. A deliver a levél fejrészét és tartalmát egy átmeneti fájlba írja, majd elindítja a felhasználó home könyvtárában lévő .deliver szkriptet. Ez meghatározza a levél célját. Az egyéni felhasználóknak olyan szkriptet kell létrehozniuk, amely képes a levél tartalmának elemzésére. Az elemzés általában a feladó címe vagy a tárgy sorban lévő kulcsszók alapján történik.

A szkript a szabványos kimeneten közli, hogy mi történjék az üzenettel. A legfontosabb kimeneti lehetőségek szintaxisa a következő:

- :Mailbox – az aktuális levelet a felsorolt fájlhoz fűzi.
- !parancs – a megadott parancsot indítja el, és a levelet a szabványos bemenetre küldi.

A következő példa ilyen szkriptet mutat be:

```
user="$1"
HEADERS="/usr/bin/header -f To -f CC -f SENDER -f Reply_to $HEADER"
SUBJECT="/usr/bin/header -f Subject $HEADER"

# Filtering Mailing lists
case "$HEADER" in
  *tkined@ibr.cs.tu-bs.de*)
  *samba*)
  *new-tty*)
  *linux* | *Linux*)
  *Firewalls* | *firewalls*)
    echo :Mail/IN/tkined; exit;;
    echo :Mail/IN/samba; exit;;
    echo :Mail/IN/linux; exit;;
    echo :Mail/IN/linux; exit;;
    echo :Mail/IN/firewall; exit;;
  esac
```

```

esac

case "$$SUBJECT" in
*Daily"" Usenet"" report*) echo :Mail/IN/news; exit;;
*Quota"" usage"" on*) echo :Mail/IN/news; exit;;

esac
echo ":Mail/IN/default"

```

A rendszer adminisztrátora úgy is telepítheti a deliver programot, hogy a .deliver szkript előtt egy központi szkriptet hívjon meg, hogy így a levelek előbb a különböző felhasználókhoz kerüljenek. További részletek a deliver kézikönyv oldalán olvashatók.

### A smail konfigurálása

A levelek egyszerű továbbítására számos program áll rendelkezésre. Sok rendszerváltozat a sendmail programot használja, másokban az smail fordul elő. Ebben a szakaszban az smail konfigurálását ismertetjük közvetlen Internet kapcsolattal rendelkező számítógépen. Ezután az eltéréseket mutatjuk be UUCP kapcsolat esetében.

Az smail démon konfigurációs adatait a /var/lib/smail könyvtárban lévő fájlok tartalmazzák. Néhány változatban ezek a konfigurációs fájlok az /usr/lib/smail.smail könyvtárban találhatók a súgófájlokkal együtt. Az smail egyéni beállítása a fordításkor vagy később a konfigurációs fájlokban hajtható végre. Utóbbi esetben a fájlból végrehajtott beállítások felülírják a fordításkor megadott értékeket.

A legfontosabb konfigurációs fájlok a következők:

- config
- routers
- transports
- directors

A config fájl globális beállításokat tartalmaz, ilyen például a számítógép neve. A bejegyzések általában attribútum=érték alakúak. A megjegyzéssort jelent.

```

# @(#) $Id: config,v 1.7 1992/09/06 04:14:28 tron Exp $

# This file specifies all of the config file variables, and gives
# default values all of them. The default values correspond to
# using defaults for all EDITME file variables, and for compiling
# under basic 4.3BSD.

# VERY IMPORTANT NOTE
#
# DO NOT COPY THE CONFIG FILE INTO THE /usr/lib/smail DIRECTORY TO
# MAKE MODIFICATIONS. Instead, if you want to change any default
# values, create a /usr/lib/smail/config file and add lines to it only

```

```

# for those variables that you wish to change. Many values in this
# file have os-dependent defaults, which may differ from the values
# given in this samples file. By setting only those attributes that
# you wish to modify, you will avoid setting inappropriate values for
# os-dependent attributes.

-auth_domains
+auto_mkdir
auto_mkdir_mode=0755
console=/dev/console
copying_file=COPYING
date_field="Date: $spool_date"
delivery_mode=background
director_file=directors
domains=uucp
-error_copy_postmaster
# +flock_mailbox # under 4.3BSD
-flock_mailbox# otherwise
fnlock_interval=3
fnlock_mode=0666
fnlock_retries=5
from_field="From: $sender$if def:sender_name: ($sender_name)"
grades="special-delivery:9:air-mail:A:first-class:C:bulk:a:junk:n"
hit_table_len=241
-hostnames
-lock_by_name# on BSD systems or System V.2/V.3
# +lock_by_name# otherwise
lock_mode=0444
log_mode=0644
logfile=/usr/spool/smail/log logfile
max_hop_count=20
max_message_size=100k
# message_buf_size=4096# for machines with small memories
message_buf_size=100k# otherwise
message_id_field="Message-Id: <$message_id@$primary_name>"
message_log_mode=0664
method_dir=methods
-more_hostnames
nobody=nobody# under 4.3BSD or SunOS
# nobody=guest# another possibility
open_interval=2
open_retries=0# if an atomic rename() is available
# open_retries=2# otherwise
paniclog=/usr/spool/smail/log/paniclog
postmaster_address=root
-queue_only
received_field="Received:
$if def:sender_host

```

```

from $sender_host by $primary_name

$if def:sender_proto: with $sender_proto

(Smail$version #$compile_num)

else by $primary_name $if def:sender_proto:with $sender_proto

(Smail$version #$compile_num)

id $message_id; $spool_date"
-require_configs
retry_duration=5d
retry_file=retry
retry_interval=10m
return_path_field="Return-Path: <$sender>"
router_file=routers
-second_config_file
smail=/usr/lib/sendmail
smail_lib_dir=/usr/lib/smail
smail_util_dir=/usr/lib/smail
-smart_path
-smart_transport
-smart_user
smtp_banner="$primary_name Smail$version #$compile_num ready at $date"
smtp_debug
smtp_accept_max=20
smtp_accept_queue=5
smtp_receive_command_timeout=5m
smtp_receive_message_timeout=2h
spool_dirs=/usr/spool/smail
spool_grade=C
# spool_mode=0640# under System V
spool_mode=0440# otherwise
transport_file=transports
-trusted
-trusted_groups
-uucp_name
-visible_name

```

A hostname a helyi számítógép nevét adja meg. Ez a kimenő üzenetek fejrészében, valamint a helyi számítógépnek küldött bejövő üzenetek felismerésénél használatos. A more hostnames alternatív neveket sorol fel. A fenti példában hermes a demo.de körzet levelezési kiszolgálója. A more hostnames=demo.de bejegyzés azt jelenti, hogy a tényleges számítógép-név a levelezési címekben elhagyható. Ily módon például a strobel@hermes.demo.de hosszú cím helyett elegendő a strobel@demo.de használata.

A routers, a transports és a directors fájlok szerepe összetettebb. Ezek a cím alapján az útvonalat és a kézbesítés módját határozzák meg. Először a címet célcímre és maradékra különítí el. A

*strobel@demo.de* címben a célcím *demo.de*, a maradék pedig *strobel*. A célcím arról közöli információt, hogy a levelet helyi vagy másik számítógéphez kell továbbítani. Helyi levél esetében a *directors* fájl segít annak eldönthésében, hogy miként kell a levelet kézbesíteni a címzethez. A *routers* fájl a távoli cím útvonalát és a használandó protokollt határozza meg. A különböző típusú átvitlek (SMTP a TCP/IP-n keresztül, UUCP stb.) beállításait a *transports* fájl tartalmazza.

A *routers* fájl elosztó bejegyzések listáját tartalmazza. Ezek az elosztók nem azonosak az IP elosztókkal: az *smail* programon belül hozzárendelt kezelőprogrammal rendelkező példányt definiálnak. Az *smail* a levélcímetek egymás után továbbítja a definiált elosztókba, amelyek előírtuk, hogy fel kell-e dolgozniuk vagy sem.

Minden egyes elosztó általános és a kezelőprogramról függő attribútumait a *routers* fájlban lehet megadni. Az általános attribútumokat az egyes számítógépekre lehet bevenni, és ezek a használandó kezelőprogramot és átviteli módot definíálják. Az egyedi attribútumok az egyes kezelőprogramok speciális jellemzőit tartalmazzák.

A következő példában a *routers* fájl egy Internet és UUCP kapcsolattal rendelkező számítógép definícióját tartalmazza. Mivel az MX rekordok kezelése a legtöbb rendszerváltozatban hiányzik, az *smail* az Interneten csak átkonfigurálás után használható.

```
# 0(#) $Id: routers,v 1.3 1992/08/08 16:40:26 tron Exp $

# This file defines the configuration of the router subsystem as
# compiled into the smail binary. By modifying the source files
# conf/EDITME, src/config.h or src/default.c the actual internal
# configuration can be changed. Thus, this should be matched up
# against these files before assuming this is completely correct.

# force_paths - override file for routing
#
# This entry is not in the compiled-in routers, for now. However,
# this entry can be very useful for adding temporary override routes,
# if some paths are known to be useless for transient reasons, or for
# organizing internal networks without the use of map files. It can
# also be used to define UUCP sites that you MX for, if you are using
# DNS with the bind router. In the MX case, add an entry to the file
# /usr/lib/smail/forcepaths such as:
#
#.foo.orgfoo-uucp-gateway!$s
#
# to cause all E-mail to the foo.org domain to be sent to its uucp
# gateway, rather than being matched by the MX router, which will
# not correctly handle MX to UUCP gatewaying, if you use the bind
# version of the inet_hosts router, below.
#
# The file is defined to be an unsorted ASCII file, so that direct
# editing is possible and reasonable. The "always" attribute forces
# use of the router even if a later router (such as the bind version
# of inet_hosts) gets a better match (note that DNS wildcard matches
# count as complete hostname matches). Thus, with the .foo.org
# entry above, foo.foo.org would be matched here, even if foo.foo.org
```

```

# were matched by inet_hosts or the regular paths router.

#
#force_paths:
#driver=pathalias,
#transport=uux,# deliver using uux to rmail
#always;# force use even if some other
## router gets a more complete match

#
#file=forcepaths,# plain ASCII file (unsorted)
#proto=lsearch,# use linear search
#optional,# ignore if the file does not exist

# inet_addrs and inet_hosts are only defined when BSD networking exists

# inet_addrs - match domain literals containing literal IP addresses
#
# For example, [128.103.1.1] will match harvard.harvard.edu on the internet.
# The library routine gethostbyaddr(3N) will be called to see if a reverse
# mapping to the canonical hostname is available.

inet_addrs:
driver=gethostbyaddr,# router to match IP domain literals
transport=smtp;# deliver using SMTP over TCP/IP

fail_if_error,# fail malformed domain literal addrs
check_for_local,# see if this is really the local host

# inet_hosts - match hostnames with gethostbyname(3N)
#
# Comment this out if you wish to use the bind version below, instead.
inet_hosts:
driver=gethostbyname,# match hosts with the library function
transport=smtp;# use default SMTP (may be UUCP-style)

-required,# no required domains
-domain,# no defined domain suffixes
-only_local_domain,# don't restrict to defined domains

# inet_hosts - alternate version using BIND to access the DNS
#
# This router is not compiled-in by default. It can be used as a
# replacement for the above inet_hosts router, if your system supports
# a bind compatible DNS resolver library. This should be used for
# systems on the internet.

inet_hosts:
#driver=bind,
#transport=smtp;# use TCP/IP SMTP for delivery
#
#defnames,# use standard domain searching
#defer_no_connect,# try again if the nameserver is down

```

```

#-local_mx_okay,# fail (don't pass through) an MX
## to the local host

# paths - route using a paths file, like that produced by the pathalias
program
paths:driver=pathalias,# general-use paths router
transport=uux;# for matches, deliver over UUCP

file=paths,# sorted file containing path info
proto=bsearch,# use a binary search
#proto=dbm,# use a dbm files
optional,# ignore if the file does not exist
-required,# no required domains
domain=uucp,# strip ending ".uucp" before searching

# uucp_neighbors - match neighbors accessible over UUCP
uucp_neighbors:
driver=uuname,# use a program which returns neighbors
transport=uux;

cmd=/usr/bin/uuname,# specifically, use the uuname program
domain=uucp,# strip ending ".uucp" before searching

# smart_host - a partially specified smarthost director
#
# If the config file attribute smart_path is defined as a path from the
# local host to a remote host, then hostnames not matched otherwise will
# be sent off to the stated remote host. The config file attribute
# smart_transport can be used to specify a different transport.
#
# If the smart_path attribute is not defined, this router is ignored.
smart_host:
driver=smarthost,# special-case driver
transport=uux;# by default deliver over UUCP

-path,# use smart_path config file variable
#path=uunet,# alternate, set path in this file

```

Az itt megadott első cílosztó csak akkor szükséges, ha az Internet kapcsolat mellett UUCP kapcsolatok is használhatók. Az cílosztó lehetővé teszi egy forcepath snevű fájl létrehozását, amely az uucp számára tartalmaz körzetneket és célcímeket. Az ebben a fájlból megadott körzetekbe szánt levelet a fájlba bevitt célcímre küldi a program.

A match-inet-addrs cílosztó IP címek használatát teszi lehetővé a levélcímekben. Ha egy szimbolikus név valamelyen okból nem oldható fel, akkor használható az IP cím: például a strobel@hermes.demo.de cím helyett a strobel@[141.7.41.1] cím.

Az Interneten levelezésre általában a match\_mx\_hosts cílosztót használják. Szimbolikus címzés MX bejegyzéseit kezeli.

A match-inet-hosts bejegyzés a címeket a resolver segítségével kezeli (lásd a 9.8. szakaszban). Ez a név/számítógépcím alakú szimbolikus címek használatát teszi lehetővé.

Ebben az esetben azonban az MX rekordok nem használhatók, azaz levél továbbítatható a `strobel@hermes.demo.de` címre, de a `strobel@demo.de` címre nem, mivel nincs ilyen nevű számítógép. Csak egy MX rekord hivatkozik a `hermes` névre.

A smarthost elosztó az összes levelet egy másik számítógéphez továbbítja. Ezt az elosztót akkor használják, ha a levelezés kézbesítéséről felelős központi levelezési átjárót alkalmaznak, és a helyi számítógépet nem használják, vagy az nem is képes a levelezés közvetlen kézbesítésére. A levelezési átjáró címét, amelyre a smarthost kezelőprogram a levelezést továbbítja, általában a config fájl `smart_path` változóiban definiálják. A config fájlból az átvitel (lásd később) is megadható a `smart_transport` változóval. Ezek a beállítások felülírják a routers fájlból megadott attribútumokat.

A smarthost elosztó használatának további ismertetését lásd az uucp beállításoknál.

Az egyedi átviteli lehetőségek jellemzőinek megadása a transports fájlból történik. A fájl felépítése megegyezik a routers fájl szerkezetével. minden átvitel számára általános és egyedi attribútumok adhatók meg.

```
# $Id: transports,v 1.11 1992/09/06 04:41:55 tron Exp $

# This file defines the configuration of the transport subsystem as
# compiled into the smail binary. By modifying the source files
# conf/EDITME, src/config.h or src/default.c the actual internal
# configuration can be changed. Thus, this should be matched up
# against these files before assuming this is completely correct.
#
# If a run-time transports file is created, then its entries will
# complement or replace the compiled-in transport entries. Thus,
# contrary to use of the routers and directors files, you do not need
# to copy and localize this file in order to add new transports.
#
#
# IMPORTANT FOR INTERNET USERS
#
# The smtp, uucp_zone_smtp, inet_zone_smtp, and local_smtp transports
# should be configured to use the DNS for finding MX and A records,
# if the host was not resolved by the bind version of the inet_hosts
# router. To configure this, uncomment the use_bind attributes on
# the various tcpsmtp-based transport definitions.
#
#
# IMPORTANT FOR SCO UNIX SYSTEMS
#
# Recent SCO UNIX systems use the MMDF file format. If you wish to
# have smail generate this format, then comment out the unix_from_hack
# and the first suffix lines, in the "local" and "file" transports,
# and uncomment all lines that are commented with "MMDF mailbox format".
#
#
# IMPORTANT FOR SYSTEM V RELEASE 4 USERS
#
# The SVR4 mailx expects to find Content-Length header fields on
```

```

# messages. If such a header is not found (or if a remote site
# supplies an incorrect Content-Length header), then mailx may split
# your mailbox file into messages at inappropriate boundaries. To
# add a Content-Length field to messages appended to your mailbox
# files, and sent to shell-command or file addresses, uncomment all
# attributes that are indicated with "SVR4 mailbox format". This
# will also ensure that you have a "Content-Type" field, defaulting
# the content type to "text".
#
# You will likely also wish to uncomment unix_from_hack from the
# local, pipe, and file transports, since prepending > to lines
# starting with From is not necessary with this the SVR4 mailbox
# format. You can also comment out the suffix="\n" lines in the
# local, and file transports, since a blank line is not required
# between messages for the SVR4 mailbox format.
#
#
# IMPORTANT FOR USE WITH HoneyDanBer UUCP
#
# Systems with HoneyDanBer UUCP (for example, System V Release 4)
# can invoke uux with -a$sender -g$grade to cause UUCP errors to
# be mailed to the message sender, and to alter UUCP queue priorities
# based on Precedence header fields. For SVR4, you should add the
# following to your /etc/uucp/Grades file:
#
#      9      9      Any      User      Any
#      A      A      Any      User      Any
#      C      C      Any      User      Any
#      a      a      Any      User      Any
#      n      n      Any      User      Any
#
# This is because the SVR4 HDB UUCP uses long message grade names
# on the uux command line, which are not supported by Smail.
#
#
# HANDLING TRANSIENT UUCP FAILURES
#
# Many systems seem to get failures from uux from time to time. I
# guess this is load related. Smail normally returns bounce messages
# in such situations. However, generating a bounce message can be
# annoying if the failures are truly transient.
#
# The only way to handle this, if you encounter this often, is to
# make sure that your UUCP configuration doesn't have any conflicts
# with your smail configuration (which implies that uux should never
# fail for configuration-related errors). Then, uncomment the
# defer_child_errors attribute in the various uux-based transports
# defined in this file. This will cause smail to retry (at a later
# time) any failures encountered by executing uux.

```

```

# local - deliver mail to local users
#
# By default, smail will append directly to user mailbox files.
#
# IMPORTANT FOR SYSTEM V AND SCO USERS

#
# comment out the mode=0600 line below, and uncomment the mode=0660
# line, to get the correct mailbox file permissions for your system.
local:    driver=appendfile,      # append message to a file
          return_path,        # include a Return-Path: field
          from,               # supply a From_ envelope line
          unix_from_hack,    # insert > before From in body
          # comment out the above line for
          # MMDF mailbox format and for
          # use with the Content-Length
          # header fields.

# SVR4 mailbox format: uncomment the below 3 lines
#
# remove_header="Content-Length",
# append_header="$if !header:Content-Type :Content-Type: text",
# append_header="Content-Length: $body_size",
local:      # use local forms for delivery

file=/usr/spool/mail/$lc:user, # location of mailbox files
file=/usr/mail/$lc:user, # use this location for System V
group=mail,             # group to own file for System V
mode=0600,              # For BSD: only the user can
# read and write file
#
mode=0660,              # under System V, group mail can access
# use this for SCO UNIX, as well
suffix="\n",            # append an extra newline
# comment out the above line for
# MMDF mailbox format and for
# use with the Content-Length
# header fields.

notify_comsat,          # notify comsat daemon of delivery
suffix="\l\l\l\l\n",     # MMDF mailbox format
prefix="\l\l\l\l\n",      # MMDF mailbox format

# local - an alternate local transport that calls on /bin/lmail
#
# Some systems have special local conventions for mail delivery that
# smail does not understand. Such conventions may include adding
# special headers, or may include particular locking conventions. For
# such systems, smail can use a system-provided program for delivery
# to user mailbox files. On most systems, the /bin/mail program will
# perform mail delivery according to local conventions. However,
# smail will often require that /bin/mail be replaced with a program

```

```

# that calls out to smail to perform delivery. By convention, the
# original /bin/mail program should be saved to /bin/lmail.
#
# It may be necessary to modify this entry to operate within your
# local conventions.
#
# NOTE: If you wish to use this alternate local entry, you must
# comment out the regular local transport entry, and uncomment the
# entry below.
local: driver=pipe,          # call out to a program
       return_path,        # include a Return-Path: field
       local,              # use local forms for delivery
       from,               # supply a From_ envelope line
       -max_addrs;         # give multiple addresses to command
#
# cmd="/bin/lmail ${user$}"
#
# pipe - deliver mail to shell commands
#
# This is used implicitly when smail encounters addresses which begin with
# a vertical bar character, such as "|/usr/lib/news/recnews talk.bizarre".
# The vertical bar is removed from the address before being given to the
# transport.
pipe:   driver=pipe,          # pipe message to another program
        return_path,        # include a Return-Path: field
        from,               # supply a From_ envelope line
        unix_from_hack,     # insert > before From in body
        # comment out the above line for
        # use with the Content-Length
        # header fields.
#
# SVR4 mailbox format: uncomment the below 3 lines
# remove_header="Content-Length",
# append_header="${if!header:Content-Type :Content-Type: text}",
# append_header="Content-Length: $body_size",
local;           # use local forms for delivery
#
cmd="/bin/sh -c $user", # send address to the Bourne Shell
parent_env,           # environment info from parent addr
pipe_as_user,         # use user-id associated with address
ignore_status,        # ignore a non-zero exit status
ignore_write_errors, # ignore write errors, i.e., broken pipe
umask=0022,            # umask for child process
-log_output,          # do not log stdout/stderr
#
# file - deliver mail to files
#
# This is used implicitly when smail encounters addresses which begin with
# a slash or squiggle character, such as "/usr/info/list_messages" or
# perhaps "~/Mail/inbox".

```

```

file:    driver=appendfile,
         return_path,          # include a Return-Path: field
         from,                 # supply a From_ envelope line
         unix_from_hack,       # insert > before From in body
         # comment out the above line for
         # MMDF mailbox format and for
         # use with the Content-Length
         # header fields.

# SVR4 mailbox format: uncomment the below 3 lines
# remove_header="Content-Length",
# append_header="$!header:Content-Type :Content-Type: text",
# append_header="Content-Length: $body_size",
local;           # use local forms for delivery

file=$user,        # file is taken from address
append_as_user,   # use user-id associated with address
expand_user,      # expand ~ and $ within address
suffix="\n",       # append an extra newline
# comment out the above line for
# MMDF mailbox format and for
# use with the Content-Length
# header fields.
mode=0644,        # you may wish to change this
# mode, depending upon local
# conventions and preferences
suffix="\1\1\1\1\n", # MMDF mailbox format
prefix="\1\1\1\1\n", # MMDF mailbox format

# uux - deliver to the rmail program on a remote UUCP site
#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
uux:   driver=pipe,
       uucp,                # use UUCP-style addressing forms
       from,                # supply a From_ envelope line
       max_addrs=5,          # at most 5 addresses per invocation
       # max_addrs=1,          # use this if some of your neighbors
       # can't handle multiple addresses
       # given to their rmail. You might,
       # alternately, want to configure
       # a uux_one_addr transport that
       # does this.
       max_chars=200;        # at most 200 chars of addresses

# the -r flag prevents immediate delivery, parentheses around the
# $user variable prevent special interpretation by uux.
cmd="/usr/bin/uux - -r $host!rmail $((\$user))",
#cmd="/usr/bin/uux - -r -a$sender -g$grade $host!rmail $((\$user))",
pipe_as_sender,   # have uucp logs contain caller

```

```

        log_output,          # save error output for bounce messages
        defer_child_errors,  # retry if uux returns an error

# demand - deliver to a remote rmail program, polling immediately
#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
demand: driver=pipe,
        uucp,                # use UUCP-style addressing forms
        from,                # supply a From_ envelope line
        max_addrs=5,          # at most 5 addresses per invocation
        max_addrs=1,          # use this if some of your neighbors
        # can't handle multiple addresses
        # given to their rmail. You might,
        # alternately, want to configure
        # a demand_one_addr transport that
        # does this.
        max_chars=200;        # at most 200 chars of addresses

# with no -r flag, try to contact remote site immediately
cmd="/usr/bin/uux - $host!rmail $((\$user))",
#cmd="/usr/bin/uux - -a$sender -g$grade $host!rmail $((\$user))",
pipe_as_sender,      # have uucp logs contain caller
log_output,           # save error output for bounce messages
defer_child_errors,   # retry if uux returns an error

# uusmtp - deliver to the rsmtip program on a remote UUCP site
#
# The rsmtip program is assumed to take batched SMTP requests.
#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
uusmtp: driver=pipe,
        uucp,                # use !-style addresses for routing
        bsmtp,                # send batched SMTP commands
        -max_addrs, -max_chars; # no limit on number of addresses

# supply -r to prevent immediate delivery, the recipient addresses
# are stored in the data sent to the standard input of rsmtip.
cmd="/usr/bin/uux - -r $host!rsmtip",
#cmd="/usr/bin/uux - -r -a$sender -g$grade $host!rsmtip",
pipe_as_sender,      # have uucp logs contain caller
log_output,           # save error output for bounce messages
defer_child_errors,   # retry if uux returns an error

# demand_uusmtp - deliver to a remote rsmtip program, polling immediately
#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.

```

```

demand_uusmtp:
    driver=pipe,
    uucp,                      # use !-style addresses for routing
    bsmtp,                      # send batched SMTP commands
    -max_addrs, -max_chars;     # no limit on number of addresses

    # with no -r flag, try to contact remote site immediately
    cmd="/usr/bin/uux - $host!rssmtp",
    #cmd="/usr/bin/uux - -a$sender -g$grade $host!rssmtp",
    pipe_as_sender,            # have uucp logs contain caller
    log_output,                # save error output for bounce messages
    defer_child_errors,        # retry if uux returns an error
#
# inet_uusmtp, inet_demand_uusmtp - batched SMTP conforming to specification
#
# These transports specify that transmitted addresses will conform to
# the SMTP specification. If a route is needed to deliver to a
# specified host, then route-addr addresses (@host1,@host2:user@destination)
# will be used. This violates recommendations of RFC1123, but routes
# are not generally required in networks where RFC1123 recommendations
# fully apply.
#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
inet_uusmtp:
    driver=pipe,
    inet,                      # use route-addr addresses for routing
    bsmtp,                      # send batched SMTP commands
    -max_addrs, -max_chars;     # no limit on number of addresses

    # supply -r to prevent immediate delivery, the recipient addresses
    # are stored in the data sent to the standard input of rssmtp.
    cmd="/usr/bin/uux - -r $host!rssmtp",
    #cmd="/usr/bin/uux - -r -a$sender -g$grade $host!rssmtp",
    pipe_as_sender,            # have uucp logs contain caller
    log_output,                # save error output for bounce messages
    defer_child_errors,        # retry if uux returns an error
#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
inet_demand_uusmtp:
    driver=pipe,
    inet,                      # use route-addr addresses for routing
    bsmtp,                      # send batched SMTP commands
    -max_addrs, -max_chars;     # no limit on number of addresses

    # with no -r flag, try to contact remote site immediately
    cmd="/usr/bin/uux - $host!rssmtp",
    #cmd="/usr/bin/uux - -a$sender -g$grade $host!rssmtp",

```

```

    pipe_as_sender,          # have uucp logs contain caller
    log_output,              # save error output for bounce messages
    defer_child_errors,      # retry if uux returns an error

# smtp - deliver using SMTP over TCP/IP
# The below four drivers are configured in only if your system supports
# BSD socket networking.
# Connect to a remote host using TCP/IP and initiate an SMTP conversation
# to deliver the message. The smtp transport is included only if BSD
# networking exists.
# The uucp attribute can be specified for transfers within the UUCP
# zone. The inet attribute must be specified for transfers within the
# Internet.
# NOTE: This is hardly optimal, a backend should exist which can handle
# multiple messages per connection.
# ALSO: It may be necessary to restrict max_addrs to 100, as this is the
# lower limit SMTP requires an implementation to handle for one
# message.

smtp:   driver=tcp.smtp,
        inet,                  # if UUCP_ZONE is not defined
        #uucp,                 # if UUCP_ZONE is defined
        -max_addrs, -max_chars; # no limit on number of addresses

        short_timeout=5m,      # timeout for short operations
        long_timeout=2h,        # timeout for longer SMTP operations
        service=smtp,          # connect to this service port

# For internet use: uncomment the below 4 lines
# use_bind,                # resolve MX and multiple A records
# defnames,                # use standard domain searching
# defer_no_connect,         # try again if the nameserver is down
# local_mx_okay,           # fail an MX to the local host

uucp_zone_smtp:
        driver=tcp.smtp,
        uucp,                  # use !-style addresses for routing
        -max_addrs, -max_chars; # no limit on number of addresses

        short_timeout=5m,      # timeout for short operations
        long_timeout=2h,        # timeout for longer SMTP operations
        service=smtp,          # connect to this service port

# For internet use: uncomment the below 4 lines
# use_bind,                # resolve MX and multiple A records
# defnames,                # use standard domain searching
# defer_no_connect,         # try again if the nameserver is down

```

```

#           -local_mx_okay,      # fail an MX to the local host

inet_zone_smtp:
    driver=tcp.smtp,
    inet,                      # use route-addr addresses for routing
    -max_addrs, -max_chars;     # no limit on number of addresses

    short_timeout=5m,          # timeout for short operations
    long_timeout=2h,            # timeout for longer SMTP operations
    service=smtp,               # connect to this service port

# For internet use: uncomment the below 4 lines
#       use_bind,              # resolve MX and multiple A records
#       defnames,                # use standard domain searching
#       defer_no_connect,        # try again if the nameserver is down
#       -local_mx_okay,          # fail an MX to the local host

local_smtp:
    driver=tcp.smtp,
    local_xform,                # transfer using local formats
    -max_addrs, -max_chars;      # no limit on number of addresses

    short_timeout=5m,          # timeout for short operations
    long_timeout=2h,            # timeout for longer SMTP operations
    service=smtp,               # connect to this service port

# For internet use: uncomment the below 4 lines
#       use_bind,              # resolve MX and multiple A records
#       defnames,                # use standard domain searching
#       defer_no_connect,        # try again if the nameserver is down
#       -local_mx_okay,          # fail an MX to the local host

# local_* - local forms for all of the remote transport entries
#
# Local format transports are useful when transferring mail messages
# within coordinated networks that all run Smail3.1. When the local
# attribute is enabled for a transport that delivers messages to a
# remote machine, any local addresses in the header or envelope of the
# message are left as local-format addresses, and the sender is left
# as a local login name.
#
# This convention for message transfers allows local networks to be
# hidden by the common mail users and by users external to the
# network. Messages transferred to remote nodes in the network will
# appear as though they originated on the receiving node. Messages
# that eventually leave the network (and are thus delivered by a
# transport that does not have the local attribute set) will be
# transformed into remote-format messages, with qualified domain names
# for all local-format addresses.
#
# This can be very convenient for networks that have a central mail

```

```

# processor that handles all mailing lists and forwarding for the
# network, and where user names are kept coordinated throughout the
# network.
#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
local_uux:
    driver=pipe,
    local_xform,          # transfer using local message format
    uucp,                 # use uucp-conformant addresses
    from,                 # supply a From_ envelope line
    max_addrs=5,          # at most 5 addresses per invocation
    max_chars=200;         # at most 200 chars of addresses

    # the -r flag prevents immediate delivery, parentheses around the
    # $user variable prevent special interpretation by uux.
    cmd="/usr/bin/uux - -r $host!rmail $((\$user))",
    #cmd="/usr/bin/uux - -r -a$sender -g$grade $host!rmail $((\$user))",
    pipe_as_sender,        # have uucp logs contain caller
    log_output,            # save error output for bounce messages
    defer_child_errors,   # retry if uux returns an error

# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
local_demand:
    driver=pipe,
    local_xform,          # transfer using local formats
    uucp,                 # use uucp-conformant addresses
    from,                 # supply a From_ envelope line
    max_addrs=5,          # at most 5 addresses per invocation
    max_chars=200;         # at most 200 chars of addresses

    # with no -r flag, try to contact remote site immediately
    cmd="/usr/bin/uux - -Shost!rmail $((\$user))",
    #cmd="/usr/bin/uux - -a$sender -g$grade $host!rmail $((\$user))",
    pipe_as_sender,        # have uucp logs contain caller
    log_output,            # save error output for bounce messages
    defer_child_errors,   # retry if uux returns an error

# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
local_uusmtp:
    driver=pipe,
    local_xform,          # transfer using local formats
    bsmtp,                # send batched SMTP commands
    -max_addrs, -max_chars; # no limit on number of addresses

    # supply -r to prevent immediate delivery, the recipient addresses
    # are stored in the data sent to the standard input of rsmtp.

```

```

cmd="/usr/bin/uux - -r $host!rsmt",
#cmd="/usr/bin/uux - -r -a$sender -g$grade $host!rsmt",
pipe_as_sender,      # have uucp logs contain caller
log_output,          # save error output for bounce messages
defer_child_errors,  # retry if uux returns an error

#
# HDB UUCP users should comment out the first cmd= line below, and
# uncomment the second.
local_demand_uusmtp:
  driver=pipe,
  local_xform,        # transfer using local formats
  bsmtp,              # send batched SMTP commands
  -max_addrs, -max_chars;  # no limit on number of addresses

  # with no -r flag, try to contact remote site immediately
  cmd="/usr/bin/uux - $host!rsmt",
#cmd="/usr/bin/uux - -a$sender -g$grade $host!rsmt",
  pipe_as_sender,      # have uucp logs contain caller
  log_output,          # save error output for bounce messages
  defer_child_errors,  # retry if uux returns an error
#

```

A local átvitel a levelet a helyi felhasználóknak továbbítja. Ez a szöveget a /var/spool/mail könyvtárban található, a megfelelő felhasználóhoz tartozó postafiókfájhoz fűzi. A művelethez használt kezelőprogram neve `appendfile`. Egyedi attribútumai a felhasználót, a csoportot és a hozzáférési jogosultságokat definiálják: a postafiók-fájl írása ezeknek megfelelően történik.

A pipe átvitelt akkor használják, ha a helyi számítógépre való kézbesítéskor a levelet egy programhoz kell továbbítani. Ez az eset akkor fordul elő, ha a felhasználók a home könyvtárukban létrehoztak egy .forward fájlt, amelyben címként programnevét tartalmazó átirányítás szerepel, például `! /usr/bin/deliver` felhasználó (lásd korábban).

Az smail a file átvitelt implicit módon használja, ha a cím elérési útvonalat tartalmaz. A local átvitelhez hasonlóan ez is az `appendfile` kezelőprogramot használja.

Az uux átvitel a leveleket az uucp segítségével kézbesíti. A pipe átvitelhez hasonlóan ez is a pipe kezelőprogramot használja egy levél továbbítására egy program szabványos kimenetére. Az uux hívásakor a -r kapcsolóval elérhető a levél azonnali kézbesítése. Ebben az esetben a levelek tárolása a spool könyvtárban történik, és továbbításukra a következő lekéréskor kerül sor.

A helyi levelek kézbesítésének eljárása a `directors` fájlból adható meg. A fájl bejegyzései elsősorban a belső jellemzőket szabályozzák, így például a címek kifejtését az /usr/lib/aliases könyvtárban lévő fájl alapján, vagy a levelek továbbítását a .forward fájlon keresztül. A fájl felépítése a `routers` és a `transports` fájl szerkezetét követi.

```

# @(#) $Id: directors,v 1.6 1992/09/06 04:41:29 tron Exp $

# This file defines the configuration of the director subsystem as
# compiled into the smail binary. By modifying the source file config.h
# the actual internal configuration can be changed. Thus, this should
# be matched up against config.h and default.c before assuming this is
# completely correct.

```

```

# aliasinclude - expand ":include:filename" addresses produced by alias files
aliasinclude:
  driver=aliasinclude,      # use this special-case driver
  nobody;                  # associate nobody user with addresses
                            # when mild permission violations
                            # are encountered

  copysecure,               # get permissions from alias director
  copyowners,               # get owners from alias director

# forwardinclude - expand ":include:filename" addrs produced by forward files
forwardinclude:
  driver=forwardinclude,   # use this special-case driver
  nobody;

  checkpath,                # check path accessibility
  copysecure,               # get perms from forwarding director
  copyowners,               # get owners from forwarding director

# aliases - search for alias expansions stored in a database
aliases:
  driver=aliasfile,         # general-purpose aliasing director
  -nobody,                  # all addresses are associated
                            # with nobody by default, so setting
                            # this is not useful.

  sender_okay,              # don't remove sender from expansions
  -sender_okay,              # do remove sender, sendmail compatible
  owner=owner-$user;        # problems go to an owner address

  file=/usr/lib/aliases,     # default: sendmail compatible
  file=mail.aliases,         # use this for YP
  modemask=002,              # should not be globally writable
  optional,                 # ignore if file does not exist
  proto=lsearch,             # unsorted ASCII file
  proto=bsearch,             # sorted file
  proto=aliasyp,             # use this for YP
  proto=dbm,                 # use this to be sendmail compatible

# dotforward - expand .forward files in user home directories
dotforward:
  driver=forwardfile,       # general-purpose forwarding director
  owner=real-$user,          # problems go to the user's mailbox
  nobody,                   # use nobody user, if unsecure
  sender_okay;              # sender never removed from expansion

  file=~/forward,            # .forward file in home directories
  checkowner,                # the user can own this file
  owners=root,                # or root can own the file

```



```

    -sender_okay,          # do remove sender from list expansions
    # sender_okay,
    # owner=$owner-$user;   # do NOT remove the sender
                           # system V sites may wish to use
                           # o-$user, as owner-$user may be
                           # too long for a 14-char filename.

    # map the name of the mailing list to lower case
    file=lists/$lc:user,
                           # smart_user - a partially specified smartuser director
                           # If the config file attribute smart_user is defined as a string such as
                           # "Guser@domain-gateway" then users not matched otherwise will be sent
                           # off to the host "domain-gateway".
                           # If the smart_user attribute is not defined, this director is ignored.
smart_user:
    driver=smartuser;      # special-case driver
                           # do not match addresses which cannot be made into valid
                           # RFC822 local addresses without the use of double quotes.
                           # well_formed_only,

```

A lists, az owners és a requests alatti bejegyzések különös figyelmet érdemelnek. Ezek címjegyzékek egyszerű létrehozását teszik lehetségesre. Ehhez elegendő egy fájlt létrehozni a /var/lib/smайл könyvtárban, amely a lista címét névként használja. A lista összes címzett-jének címét ebbe a fájlba kell bevenni.

### A mail és a deliver

Ha számos olyan felhasználó van, aki levelezését különböző postafiókokba szeretné rendezni egy program segítségével (például procmail vagy deliver, lásd korábban), célszerű a programot közvetlenül a local átvitelbe integrálni. A deliver példáján ebben a szakaszban azt mutatjuk be, hogy ehhez milyen módosításokat kell végrehajtani.

A local átvitel az appendfile helyett a pipe kezelőprogramot használja. Programként a deliver teljes elérési útvonalát kell megadni. Ez általában /usr/bin/deliver.

```

pipe:  driver=pipe,                      # pipe message to another program
       return_path,                     # include a Return-Path: field
       from,                           # supply a From_ envelope line
       unix_fromHack,                  # insert > before From in body
                                         # comment out the above line for
                                         # use with the Content-Length
                                         # header fields.

                                         # SVR4 mailbox format: uncomment the below 3 lines
# remove_header="Content-Length",
# append_header="$if !header:Content-Type :Content-Type: text",
# append_header="Content-Length: $body_size",

```

```

local;                                # use local forms for delivery

cmd="/usr/bin/deliver $lc:user", # send address to deliver
parent_env,                         # environment info from
                                     # parent addr
pipe_as_user,                        # use user-id associated
                                     # with address
ignore_status,                       # ignore a non-zero exit status
ignore_write_errors,                 # ignore write errors, i.e.,
                                     # broken pipe
umask=0022,                           # umask for child process
-log_output,                          # do not log stdout/stderr

```

Ezután azoknak a felhasználóknak, akik levelezésüket automatikusan különböző mappákba szeretnék rendezni, minden össze egy `.deliver` szkriptet kell elhelyezniük home könyvtárukban.

### A small és az UUCP

Ha valaki nem rendelkezik közvetlen Internet kapcsolattal, vagy levelezést bármilyen más okból az UUCP protokollon keresztül szeretné átvinni, a legegyszerűbb megoldás számára a smart-host closztó használata, és a kimenő levelezés továbbítása egy jobb kapcsolattal rendelkező számítógépre. Ennek megvalósításához a smart host kivételével az összes bejegyzést törölni kell a routers fájlóból, és a smart path változót a levelezési átjáró címére, míg a smart transport változót uux értékre kell beállítani a config fájban.

Részlethes leírás és a konfigurálás ismertetése az `smail` kézikönyv oldalán található.

## 10.8 NEWS

A USENET News az új fejlesztések és egyéb figyelemre méltó információk egyik legfontosabb hírforrása.

### Felépítés és telepítés

A működési elv meglehetősen egyszerű. Az Interneten számos helyen telepítettek News kiszolgálókat, így például szinte minden egyetemen. A News kiszolgálók különböző hírcsoportokat (newsgroup) kezelnek, amelyek kétirányú nyilvános elektronikus fórumokat jelentenek adott téma-körben, és tartalmazzák a küldési, az olvasási és a válaszolási lehetőségeket. Ezek a téma-körök az operációs rendszerektől és programuktól a sportig, a számítógépes játékoktól és egyéb szabadidős tevékenységektől a társasági eseményekig szinte mindenekkel tartalmaznak. Világszerte számos ilyen hírcsoport működik (*jelenleg több mint 14 000 hírcsoport van*).

Az összes News kiszolgáló állandó hírcserét folytat a szomszédos kiszolgálókkal, így egy adott csoport News kiszolgálóján feladott üzenet gyorsan átkerül a többi News kiszolgálón is. Ezek az üzenetek adott ideig, például két hétag, a News kiszolgálókon maradnak, majd törlik őket.

A hírekkel ilyen üzenetek formájában cserélő számítógépek teljes hálózatát Usenet rendszerek nevezik. Az ezen számítógépek közötti kommunikáció eredetileg modemeken keresztül és az UUCP (Unix to Unix Copy Program) protokoll segítségével történt. Ez lényegesen lassabb volt, mint a jelenleg Interneten keresztül használt megoldás, amely az UUCP helyett egy NNTP nevű speciális protokollt használ.

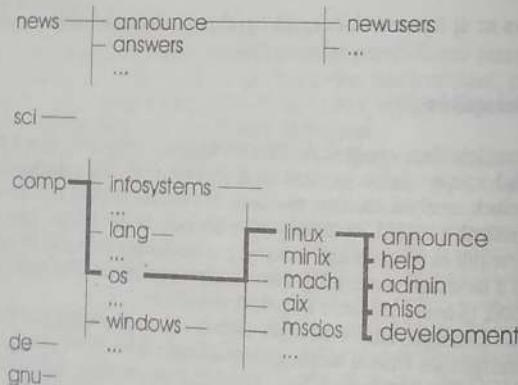
### Hírcsoportok

A hírcsoportok hierarchikus felépítésük (lásd a 10.7. ábrán). Teljes, pontokkal tagolt nevük a hierarchiában elfoglalt helyetket azonosítja. Például a comp.os.linux.announce azt jelenti, hogy ez a hírcsoport a comp alatt található, amely számítógépekkel, szoftverekkel és általában a számítógépes tudományval foglalkozik. Az os az operációs rendszerekre utal, és így a comp.os.linux minden olyan nevű csoportot magába foglal a comp hierarchiában, amely a Linux rendszerrel foglalkozik. (A Linuxszel foglalkozó levelezési listák news másolatai a linux.listanév csoportokban találhatók, a magyar hírcsoportok pedig a hun. alatt - a lektor.)

A comp hierarchia mellett számos egyéb fontos főkategória (úgynevezett terjesztés) is létezik, így az sci a tudományos kérdéseket tárgyaló csoportok számára, vagy az alt az egyéb területek széles tartományára.

### Moderált csoportok

A csoportok téma nyitott lehet, vagy lehetnek szűkitettek témaúj csoportok. Utóbbi esetben egy moderátor előnti, hogy a javasolt üzenetek megfelelnek-e a csoport létrehozásakor rögzített elveknek: ha igen, akkor azokat továbbítja, míg a kiesők süllyesztőbe kerülnek. Például a Linux esetében a comp.os.linux.announce ilyen moderált csoport (*a moderáltság jó értelemben vett, önként vállalt cenzúráját jelenti itt, a csoport hatékony működése érdekében vezeti be - a lektor*), amely kizárolag az új programokra vagy rendszervőlítésekre vonatkozó bejelentésekkel foglalkozik. (Tehát törölnek minden egyéb témaúj hozzászólást, így azok, akik kizárolag az újdonságokról szeretnének olvasni, mentesülnek – pontosabban a moderátor mentesíti őket – a levelek tömegének elolvasásától. Amit itt találnak, az biztosan érdekes számukra, tehát a saját hatékonyságuk nő – a lektor.) Ennek bevezetése különösen azért szükséges, mert a többi Linux csoport közleményeinek száma olyan hatalmas mértéket ért el, hogy az érdeklődőknek tekintelyes időbe kerül ezek végigolvásása és rendszerezése. (Valóban félelmetes mértéket öltött a news forgalma, a több, mint 14000 csoportba naponta 800 MB-1 GB közötti összméreben, egy forgalmasabb news kiszolgáló akár 800 ezer-1 millió cikket is továbbíthat felhasználónak – a lektor.)



10.7 ábra. A hírcsoport-hierarchia egy részlete

A nem moderált csoportok nyitottak, azaz minden résztvevő szabadon közölni közleményeket, kérdéseket és válaszokat.

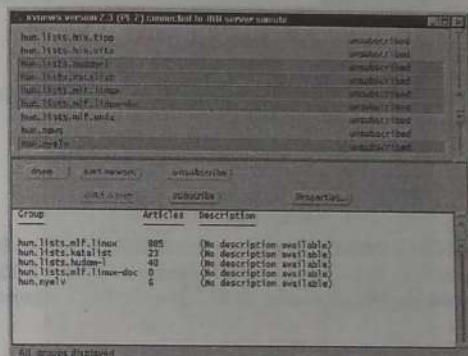
#### **Szabályok és a hálózati etikett**

**Új hírcsoport létrehozására** akkor kerül sor, ha egy Usenet résztvevő azt javasolja, és a hálózati szavazáson ezt elfogadják. A javasolt új csoportokról az eszmecsérére általában a news . group csoportban kerül sor.

Amikor egy résztvevő üzenetet kíván küldeni egy hírcsoportba, bizonyos szabályokat be kell tartani: ezeket összefoglalóan hálózati etikettnek (netiquette) nevezzük. Sérző megjegyzések vagy személyes támadások nem megengedettek. A news.announce.newusers hírcsoport által kírtést ad a hálózati viselkedés ezen szabályairól.

Műveltség

A NetNews közleményeinek olvasásához egy hírolvasó és hálózaton vagy modernen keresztül elérhető News kiszolgáló szükséges. Az egyetemeken ez általában nem jelent problémát, mivel ezek rendszerint saját News kiszolgálókat működtetnek.



10.8. ábra Az xynews hírolvásó

A hírolvasók szinte az összes, a hálózati működést biztosító számítógépes platformon és operációs rendszerben rendelkezésre állnak (lásd a 10.8. ábrán). Az elterjedt hírolvasók közé tartozik az *rn*, a *trn*, a *tin* és az *xrn*, az X Window System rendszer számára az *xnews*, *knews*, és a PC-k számára a *trumpet*. Ezek a hírolvasók szinte kivétel nélkül megtalálhatók a Linux alatt is. A legtöbb hírolvasó mindenkor egyetlen konfigurációs beállítást igényel: az **NNTPSERVER** környezeti változóban meg kell adni a News kiszolgáló nevét (lásd a 2.7. szakaszban).

Elterjedt még a két változatban (Athena és Motif) rendelkezésre álló *x̄n* is (lásd a 8.5. szakaszban).

## **News kiszolgálók**

News kiszolgálók telepítése némileg bonyolultabb egy ügyfél telepítésénél. Egy hírolvasó számára elegendő a programot a számítógépre másolni, és beállítani egy környezeti változót. A kiszolgálóhoz ezzel szemben számos programot és konfigurációs fájlt kell telepíteni. Ahhoz, hogy közlekedémenyeket kaphassunk, egy másik News kiszolgálóra is szükség van, amely a híreket továbbítja számunkra.

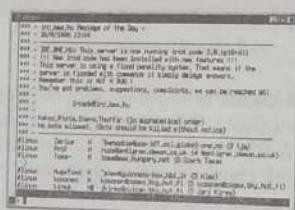


10.9 ábra. xrn

Az Internet a régebbi C news kiszolgálóprogram mellett általában az INN kiszolgáló használja. Az INN konfigurációját több szövegfájl tartalmazza.

10-9IRC

Az Internet Relay Chat (IRC) több résztvevő közvetlen beszélgetését teszi lehetővé az Interneten. A News rendszerhez hasonlóan az IRC is egymással adatcsérét folytató, hálózatba kapcsolt kiszolgálókból áll. A témaörök szerinti fórumokat az IRC rendszerben *csatornáknak* nevezik (lásd a 10. ábrán).



10.10 ábra. JRC

A News és az IRC közötti lényeges különbség az, hogy utóbbiban az üzenetek cseréje késleltetés nélkül, azonnal megtörténik, és így a felhasználó közvetlenül beszélgethet a többi résztvevővel. Ennek akkor lehet különös jelentősége, ha súrűs problémával küzd.

A felhasználók a hozzáférést egy különleges ügyfélprogrammal (*irc client*) biztosítják, amely létrehozza a kapcsolatot a legközelebbi IRC kiszolgálóval. Az IRC ügyfél **bizonyos parancsokat** használhat, amelyek mindegyike a (/) karakterrel kezdődik. Ezek a parancsok például lehetővé teszik a bejelentkezést egy csatornára vagy a kijelentkezést egy csatornáról. A parancs nélküli bevitt szöveg azonnal továbbításra kerül, és megjelenik az összes többi olyan IRC résztvevőnél, aki bejelentkezett ugyanarra a csatornára.

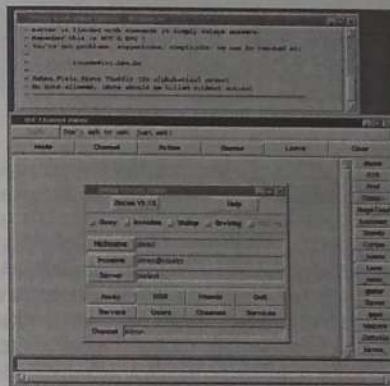
Az IRC a Linux számára fenntartott csatornával is rendelkezik. A beszélgetésben való részvételhez el kell indítani az irc programot, majd ezt kell beírni: /join #linux. A /join egy csatornára való bejelentkezés parancsa. Az ezután a csatornán elküldött üzenetek a feladó nevével együtt megjelennek.

A csatornák száma folyamatosan változik, mivel minden felhasználó jogosult új csatornák megnyitására. Ez szintén a /join paranccsal hajtható végre. Ha a megadott csatorna még nem létezik, létrehozza azt a rendszer. Általában csatornák ezrei aktívak egyidejűleg; az aktuálisan megnyitott csatornák a /list paranccsal jeleníthetők meg. (Jobb, ha nem próbáljuk ki, sokszor 10-20 percig is listázza a csatornákat – a lektor.)

Az on-line beszélgetésen kívül az IRC protokoll kiegészítései kisebb fájlok cseréjére is lehetőséget ad. Emellett személyes üzenetek is továbbíthatók.

Az IRC rendszerben való részvétel lehetséges, és talán célszerűbben megvalósítható, a zircon grafikus előfeldolgozón keresztül is (lásd a 10.11. ábrán). A zircon az egérrel kezelhető, és a megnyitott csatornákat külön ablakokban jeleníti meg.

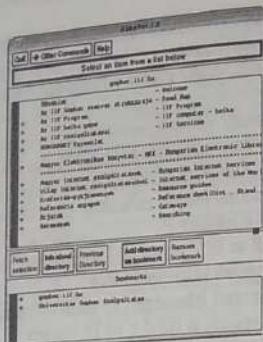
A zircon a Tcl/Tk nyelvben készült, és a TCP/IP eléréséhez a tcl-dp Tcl értelmező szükséges (lásd a 14.5. szakaszat).



10.11 ábra. A zircon, a grafikus IRC ügyfél

## 10.10 GOPHER

A Gopher olyan Internet szolgáltatás, amely számos egyéb, világszerte az egyetemek és más intézmények által rendelkezésre bocsátott szolgáltatást tesz elérhetővé, ezeket egyetlen felhasználói interfésszen egyesíti. A Gopher a felhasználóknak egyetlen nagy, hierarchikus menüként jelenik meg. Ezen a struktúrán belül a felhasználók átléphetnek egyik kiszolgálóról egy másikra, és különböző hálózati szolgáltatásokat használhatnak. A kínálat magában foglalja az irodalmi adatbázisokat, a meteorológiai szolgáltatásokat, a dokumentumok számos típusát (szöveget, képeket és hangot), valamint a különböző egyetemek menüit, Telnet futtatásokat és átjárókat más szolgáltatásokra (például WAIS). Egyre több egyetem kínálja olyan információit saját Gopher szolgáltatásán keresztül, amelyek korábban csak nyomtatott formában voltak elérhetők.



10.12 ábra. Az xgopher

A Gopher használatához a felhasználónak egy Gopher ügyfélre van szüksége. A Linux alatt ez lehet az X11 alatt futó *xgopher*. Elindításakor az *xgopher* kapcsolatot létesít a Gopher kiszolgálóval, és ezután a felhasználó szabadon mozoghat az Interneten (lásd a 10.12. ábrán).

Az utóbbi években a World Wide Web népszerűségének növekedése nyomán a Gopher jelentősége csökken. Szinte valamennyi intézmény, amely információt eredetileg a Gopher szolgáltatásain keresztül kínálta, mára áttért a WWW használatára.

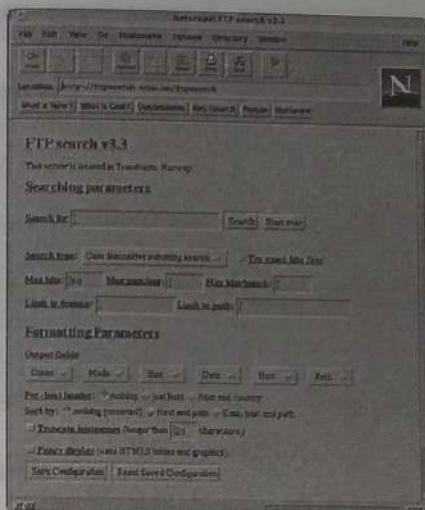
## 10.11 WORLD WIDE WEB

A World Wide Web (amelyre gyakran a WWW vagy a W3 rövidítéssel is hivatkoznak) multimédiás hypertext formátumú dokumentumok terjesztésével foglalkozó Internet szolgáltatás. Ezek a dokumentumok nem csak formázott szöveget, hanem más dokumentumokra való hivatkozásokat, képeket, videót és hangfájlokat is tartalmazhatnak. Ezek a dokumentumok a **HTML** (HyperText Markup Language) nyelven ASCII formátumban írhatók le. Az adatátvitelre az egyedi **HTTP** (HyperText Transfer Protocol) protokollt használják.

A HTML dokumentumokban a hivatkozások megadása a Universal Resource Locator (URL) segítségével történik, amely a **protokoll://gépnév:portszám/útvonal/ /fájlnév** leírást tartalmazza. A portszám beírása nem kötelező: ha a portszám nem szerepel, akkor a megadott protokoll szabványos portját használja a rendszer. Az ilyen URL-ek a kifejezetten lehetővé teszik a hivatkozást más HTML fájlokra, valamint az Interneten elérhető szinte valamennyi szolgáltatás címére. Például a *tsx-11.mit.edu* FTP kiszolgálón lévő */pub* könyvtára hivatkozó URL a következő: *ftp://tsx-11.mit.edu/pub/*.

A WWW eléréshez egy WWW ügyfél szükséges. Ezeket **WWW tallózónak** is nevezik. Az első és legismertebb ezek közül a *Mosaic*. Ez az összes általánosan használt operációs rendszeren rendelkezésre áll, és egyéni használata ingyenes. Az újabb PC-s operációs rendszerek, például a Windows 95 és az OS/2 Warp már tartalmaz WWW ügyfelszoftvert. A *Mosaic* fordításához OSF/Motif szükséges, amely nem szabadon terjeszthető (lásd a 10.13. ábrát). Létezik azonban egy előre lefordított verziója, amelyhez statikusan hozzászerkesztik a Motif részeket, így ehhez nem szükséges az OSF/Motif futtató rendszer.

A Linux alatt a *Mosaic* mellett az újabb WWW tallózók, így a *Netscape* és az *Arena* is rendelkezésre áll. A *Netscape* jelenleg kereskedelmi termék, amelynek előnye az adatbiztonság figyelembe vétele. Az átvitel különleges kiszolgálókkal RSA kódolással történhet. Ez az olyan kereskedelmi alkalmazásokban jelenthet előnyt, ahol személyes és titkos adatok, például hitelkártya-adatok, átvitele történik.



10.13 ábra. A Netscape WWW tallózó

Az Arena a tallózók első olyan megvalósítása, amely már a **HTML3**-as szabványt használja. Ez a WWW lapok leírására jelenleg használt HTML szabványt fogja felváltani. Ez a fejlesztés még tart, így az Arena most még csak elősorban tudományos érdeklödésre tarthat számot.

A **HTTP kiszolgáló** segítségével egy elosztott információs rendszer a Linux alatt egyszerűen felépíthető. A háttérben démonként futó ilyen kiszolgáló a megadott porton keresztül foglalat-kapszolatra várokozik; miután a kapcsolat egy WWW ügyfél (például a Mosaic) segítségével megvalósult, a kiszolgáló hozzáférést biztosít adott könyvtárakban elhelyezkedő fájlokhoz. Ha közvetlen Internet hozzáférés valósítatható meg, akkor a helyi dokumentumok megfelelő csatlakozást biztosítanak egyéb **HTTP kiszolgálókhoz** is. A Linux felhasználók számára a fontosabb kiszolgálók a következő URL címeken találhatók:

<http://www.linux.org/> WWW kiszolgáló, amely FAQ és HOWTO ismertetők, kézikönyvek és más WWW kiszolgálók kiterjedt listáját tartalmazza.

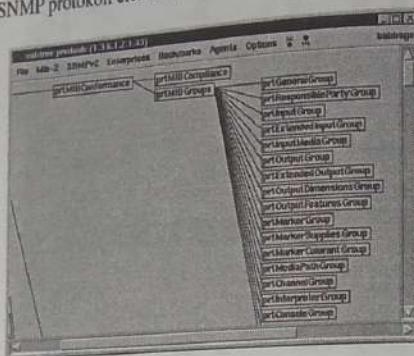
<http://www-i2.informatik.rwth-aachen.de/Linux.html> az RWTH Aachen (Németország) Linux archívumának WWW kiszolgálója. Információkat tartalmaz a Linux rendszerről, valamint hozzáférést biztosít más FTP és WWW kiszolgálóhoz.

<http://www.linux.org/> Ezt az angol kiszolgálót Alan Cox vezeti. A Linux rendszerre készült kereskedelmi szoftverek listáját tartalmazza.

## 10.12 HÁLÓZATKEZELÉS

A tkined különös figyelmet érdemlő eszköz a helyi hálózatok kezelésének területén (lásd a 10.15. ábrán). Helyi TCP/IP és nagy területre kiterjedő hálózatok (WAN) felügyeletét és adminisztrálását teszi lehetővé. Szolgáltatásai közé tartozik a hálózatmenedzsment-struktúra automata felismerése, a grafikus megjelenítés megvalósítása, kapcsolatok és egyedi elemek nyomon követése, SNMP elemek adminisztrálása és még számos egyéb szolgáltatás. Mivel a tkined fejlesztése a Tcl/Tk alatt történt, továbbfejlesztése egyszerű, és alkalmas az egyedi ízénekek kiel-

gítésére. A **scotty** Tcl értelmezőt használja, amely a Tcl alatt lehetővé teszi a TCP, az UDP, az ICMP, a DNS és az SNMP protokoll elérését (lásd a 14.5. szakaszban).



10.14 ábra. A tkined

# TERMÉKTÁMOGATÁS ÉS SÚGÓ

**G**yakran felvettődő érv, különösen az ingyenes és a szabadon terjeszthető szoftverekkel szemben, a terméktámogatást és **forródrót** szolgáltatást nyújtó jogosult szervezet hiánya. Az USA-ban néhány vállalat felismerte ezt a hiányt, és felvállalta az **ingyenes szoftvertermékek** kereskedelmi támogatását. Hasonló kezdeményezések Európában is megfigyelhetők (lásd a Mellekítményben).

Az ilyen támogatáson kívül konkrét problémákra számos módon információt és segítséget lehet kapni az **Interneten** keresztül vagy magából a Linux-rendszerből. Ez a fejezet ezeket a lehetőségeket foglalja össze, valamint ismerteti a Linuxról rendelkezésre álló dokumentációt.

## 11.1 MAN, XMAN

A legtöbb UNIX-rendszerhez hasonlóan a legegyszerűbb információforrás az **elektronikus dokumentáció**. Ez olyan fájlokból áll, amelyek egy-egy parancsot, egy C könyvtári rutint, egy eszközt vagy egy konfigurációs fájlt írnak le. Az elektronikus kézikönyv oldalainak megjelenítésére szolgáló parancs a **man**. Éppen ezért ezeket a fájlokat Manual oldalaknak vagy egyszerűen **man** oldalaknak nevezik.

A **Manual** oldalakat tartalmuk szerint szakaszokra osztják. Ezeket a szakaszokat 1 és 9 közötti számokkal, valamint az n és az l betűvel jelölik az alábbiak szerint:

- 1 – felhasználói parancsok
- 2 – rendszerhívások
- 3 – C könyvtári rutinok
- 4 – a / dev könyvtárban található eszközfájlok
- 5 – fájlformátumok (rendszerint konfigurációs fájlok)
- 6 – játékok
- 7 – mindenféle
- 8 – rendszeradminisztrálás
- 9 – rendszermag-rutinok
- n – új **Manual** oldalak
- l – helyi **Manual** oldalak

A **Manual** oldalak különböző könyvtárakban találhatók, és itt az egyes szakaszokhoz tartozó fájlok alkönyvtárakban helyezkednek el. Erre a célra általában az **/usr/man** és az **/usr/local/man** könyvtár használatos. A következő példa az **/usr/man** könyvtár szakaszokhoz tartozó alkönyvtárait mutatja be.

```
nicetry:/usr/man# ls
Makefile          cat8@      man5/
README           cat9@      man6/
cat1@           catn@      man7/
```

```

cat28          man-pages-1.8.Announce   man8/
cat38          man-pages-1.8.1sm       man9/
cat48          man1/                  mann/
cat58          man2/                  preformat/
cat68          man3/                  whatis
cat78          man4/                  whatis
nicetry:/usr/man#

```

A man parancs verziójától és konfigurációjától függően más könyvtárak és alkönyvtárak is használhatók.

A rendszer adminisztrátora a makewhatis parancs segítségével indexfájlt készíthet a Manual oldalakhoz. Az indexfájl a parancsok nevét és egysoros leírását tartalmazza. Ez a fájl a legtöbb Linux-változat esetében már rendelkezésre áll. A felhasználó a whatis és az apropos parancs segítségével kulcsszavak alapján kereshet ezekben.

```

nicetry:/usr/man# whatis gcc
gcc, g++ (1)      - GNU project C and C++ Compiler (v2.6)
gcc, g++ (1)      - GNU project C and C++ Compiler (v2.7)
gcc, g++ (1)      - GNU project C and C++ Compiler (v2.7)
nicetry:/usr/man# whatis groff
groff (1)         - front end for the groff document formatting system
groff (1)         - front end for the groff document formatting system
groff (1)         - front end for the groff document formatting system
nicetry:/usr/lib/X11/fvwm# apropos pbm
g3topbm (1)       - convert a Group 3 fax file into a portable bitmap
hp2pbm (1)        - Translate HPLJ codes to PostScript or PBM or G3
pbmtog3 (1)       - convert portable bitmaps (PBM) into G3 fax files
g3topbm (1)       - convert a Group 3 fax file into a portable bitmap
hp2pbm (1)        - Translate HPLJ codes to PostScript or PBM or G3
pbmtog3 (1)       - convert portable bitmaps (PBM) into G3 fax files

```

### A Manual oldalak formátuma

Az elektronikus Manual oldalai gyakran két vagy három különböző formátumban rendelkezésre állnak. Az eredeti formátum általában az nroff forráskód, és ez a man1–man8 könyvtárban található. Ez a formátum nem alkalmas a közvetlen megjelenítésre, ezt előbb az nroff vagy a groff szövegformázó programmal le kell fordítani (lásd a 12.4. szakaszban).

Az összetettebb Manual oldalaknál ez az eljárás meglehetősen időigényes, ezért a fájlokat a fordítás után olvasható formátumban a cat1–cat8 könyvtárban tárolják. Ennek végrehajtásához azonban a felhasználónak írás jogosultsággal kell rendelkeznie ezekhez a könyvtárakhoz. Szükség szerint ezek a fájlok a compress programmal tömöríthetők is, ami ugyan lényegesen növeli a megjelenítésükhez szükséges időt, de így jóval kevesebb tárolóhelyet igényelnek.

Ha a Manual oldalak telepítése nem az /usr/man könyvtárba történt (hanem például az /usr/local/man könyvtárba), akkor a Manual oldalak elérési útvonala a MANPATH környezeti változóban adható meg. A következő példa a Manual oldalak útnévénél megadását mutatja be Bourne rendszermag esetében:

```
export MANPATH=/usr/man:/usr/openwin/man:/usr/local/man
```

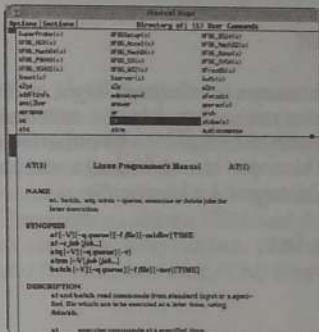
Az alábbi példa a Manual oldalak útnevének megadását C shell esetében mutatja be:

```
setenv MANPATH /usr/man:/usr/openwin/man:/usr/local/man
```

A fenti beállítások természetesen végrehozhatók a megfelelő indítófájlban is (.profile vagy .login).

### xman

Az X11 lehetővé teszi egy némképp kényelmesen használható segédprogram, az xman használatát (lásd a 11.1. ábrán). Ebben a felhasználó előbb kiválasztja a Manual megfelelő szakaszát. Ezután a segédprogram a szakaszhoz tartozó összes Manual oldal áttekintését jeleníti meg. Ebből a felhasználó az egérrel kijelölheti a megjeleníteni kívánt Manual oldalt.

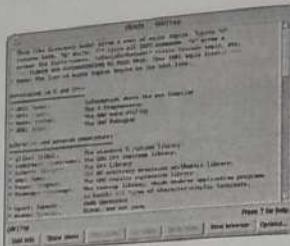


11.1 ábra. Az xman

Ha probléma adódik egy parancssal vagy egy program konfigurálásával, az első segítségnyújtási hely a megfelelő Manual oldal lehet. Természetesen a man parancs használatát ismertető Manual oldal is megtalálható, amelyet a felhasználó a man man befrásával hívhat.

## 11.2 AZ INFO FORMÁTUM

A legtöbb FSF program dokumentációja GNU Info formátumban is rendelkezésre áll. A GNU Emacs használja ezt a formátumot, amellyel a dokumentumban a hypertexthez hasonló ugrások valósíthatók meg. Az Emacs szerkesztő újabb verzióiban, ilyen például a Lucid Emacs vagy az Emacs 19, X11 alatt az egérrel vezérelhető a dokumentumban való mozgás. Olyan programok is rendelkezésre állnak, ilyen az info, xinfo vagy a Tcl/Tk alapú tkinfo, amelyek kifejezetten az információs fájlok megjelenítésére készültek (lásd a 11.2. ábrát).



11.2 ábra. A tkinfo

Az Emacs szerkesztő belül a felhasználó kiválaszthatja az Info üzemmódot (<Ctrl-H><I>), majd a rendelésre álló dokumentumokból kijelölheti a kívánt szöveget. A megjelenő szövegen a kulcsszavak szerint valósítható meg a keresés.

Az Info dokumentumok példáit a GNU C fordító és a GNU AWK segédprogram dokumentációja tartalmazza:

- comp.os.linux.announce (c.o.l.a.)
  - comp.os.linux.help
  - comp.os.linux.misc
  - comp.os.linux.admin
  - comp.os.linux.development.apps
  - comp.os.linux.development.system
  - comp.os.linux.hardware
  - comp.os.linux.networking
  - comp.os.linux.setup
  - comp.os.linux.x
  - comp.os.linux.advocacy
- A comp.os.linux.announce moderált hírcsoport, amely csak az új programok és a Linux új terjesztések bejelentései foglalkozik. A moderált csoportba szánt üzenetek megjelenése nem közvetlen, előbb egy moderátor szűrőjén kell áthaladniuk. A moderátor a hírcsoportra megállapított szabályok és irányelvek betartásáért felelős. A többi hírcsoportban semmiféle korlátozás sem áll fenn: itt tetszőleges részvétő üzeneteket vagy kérdéseket lehet megírni. A felhasználóknak meg kell fontolniuk, hogy minden, egy hírcsoportnak szánt üzenet világszerte elérhető.
- Jelenleg ezeken a hírcsoportokon 100-nál több közlemény jelenik meg naponta. Ez meglehetősen bonyolulttá teszi a naprakész tájékozódást a hatalmas adatmennyiségen: néhány naponta gondosan át kellene olvasni a közményeket. Megoldást jelenthet egy hírcsoportfigyelő a közleményekre adott válaszok tanulmányozásával.

## 11.4 FAQ ÉS HOWTO ISMERTETŐK

Egy további, a hírcsoportokhoz szorosan kapcsolódó információforrás a FAQ (frequently asked questions, gyakran felmértilő kérdések) közlemények. Ez a közleményekben gyakran felmerülő kérdések listáját, és ami még fontosabb, az ezekre adott válaszokat tartalmazza.

A FAQ közlemények a téma körök széles tartományát érintik. Ezeket általában a hírcsoportok aktív résztvevői megszerkesztett formában teszik közzé, így elkerülhető az azonos kérdések ismétlődése. A FAQ ismertetők így a kérdések mellett a hírcsoportok avatott tagjainak válaszait is tartalmazzák.

Az eredeti Linux FAQ közlemények egy része HOWTO (hogyan fogunk hozzá) dokumentumokká alakult át. Ezek hasonlítanak a FAQ dokumentumokra, de még részletesebb magyarázatot tartalmaznak.

A FAQ és a HOWTO közlemények egyszerű szöveges fájlok, így elolvashatók és kinyomtathatók tetszőleges szövegszerkesztővel, a UNIX more parancsával, vagy akár a DOS rendszeren is. Alapszabály, hogy a FAQ és a HOWTO közlemények nem foglalkoznak általános UNIX kérdésekkel. Erre a célra külön hírcsoport létezik (`comp.unix.questions`) saját FAQ közleményekkel. A UNIX új felhasználóinak tehát ebben is érdemes tallózniuk. (Másik alapszabály, hogy a dokumentáció, FAQ, HOWTO elolvasása után illik csak kérdést feltenni a newsban – a lektor.)

### FAQ források

Tetszőleges típusú FAQ közlemények legjobb hírforrása egy speciális hírcsoport, a `news.answers`. Számos csoport rendszeres időközönként megjelenteti saját FAQ közleményeit itt. A Linux, a UNIX és a programozási nyelvek mellett egy új böngésző segítségével a számítógépes kérdésekben kívül eső területekről kikereshetők a téma körök.

Kifejezetten Linux FAQ közlemények rendszeresen megjelennek a `comp.os.linux.announce` hírcsoporthban is. Természetesen számos FTP kiszolgálón is találhatók aktuális FAQ közlemények, általában a Linux alatti `doc` könyvtárban. Például a `ftp.funet.fi` FTP kiszolgálón ezek tárolása a `/pub/OS/Linux/doc/FAQ` könyvtárban történik.

## 11.5 WWW

A World Wide Web (WWW) jelenleg az egyik legvonzóbb és leggyakrabban használt szolgáltatás az Interneten (lásd a 10.11. szakaszt). Számos WWW kiszolgáló létezik a Linux számára is, és ezek FAQ és HOWTO közleményeket, valamint kézikönyveket és egyéb információt is közzétesznek.



11.3 ábra. Linux információkra hivatkozó WWW oldal

## 11.6 LEVELEZÉSI LISTÁK

A Linux-rendszer aktív közreműködői számára számos levelezési lista áll rendelkezésre. Ezek a listák elsősorban az új fejlesztések, a felmerülő problémák, ötletek és új programváltozatok cseréjét segítik. A levelezési listák címére küldött üzeneteket összegyűjtik, és naponta többször továbbítják a lista tagjainak.

A levelezési listához való csatlakozáshoz a felhasználó a `help` parancsot is tartalmazó üzenetet küld a `Majordomo@vger.rutgers.edu` címre, és a válasz részletes utasításokat tartalmaz a listaszerver használatáról. A legegyszerűbb esetben a következő parancs elegendő a segítségkéréshez:

```
echo help | mail Majordomo@vger.rutgers.edu
```

Mivel ezek a levelezési listákat elsősorban a Linux fejlesztőinek szánták, a kezdők nem tehetnek fel kérdéseket a szokásos csatornákon.

## 11.7 EGYÉB DOKUMENTUMOK

A Linux Documentation Project (LDP) számos szerzője állított össze a Linuxról kézikönyveket. A *Linux User's Guide* és a *Linux Installation Guide* mellett ezek a dokumentációk összetettebb kérdezésekkel is érintenek, ilyen például a *Network Administration Guide* (NAG), amely a Linux alatti hálózati telepítés szinte valamennyi kérdését tárgyalja. A *Kernel Hacker's Guide* (KHG) a rendszermag felépítésével és a Linux alatti eszközkezelők fejlesztésével foglalkozik. Ez a dokumentáció elsőrangú áttekintést nyújt a Linux belső működéséről is.

Ezek a kézikönyvek az FTP kiszolgálón keresztül szerezhetők be, és kinyomtathatók. A könyvesboltokban a fenti dokumentációk kinyomtatott és bekötött példányai is kaphatók. Az elérhető fájlok a `sunsite.unc.edu` FTP kiszolgáló `/pub/Linux/docs/LDP` könyvtárában állnak rendelkezésre.

## 11.8 EGYÉB FORRÁSOK

Az eddig említett források mellett még további számos Internet-szolgáltatás és -szervezet kínál információkat és dokumentációkat. Ilyenek a következők.

### WAIS

A WAIS (Wide Area Information System) olyan Internet-szolgáltatás, amely WAIS kiszolgálókon tetszőleges téma-körű dokumentációkat keres kulcsszavak segítségével. A WAIS rendszeren keresztül FAQ közlemények és kézikönyvek kérhetők le. Az ügyfélsofтверek, a WAIS kiszolgálók lista-já és a WAIS részletes ismertetése az [FTP think.com](http://think.com) gazdagépről vagy a [comp.infosystems.wais.hiresportból](http://comp.infosystems.wais.hiresportból) tölthetők le.

### README fájlok

A legtöbb nagyobb programhoz hasonlóan a Linux rendszermag dokumentációját is kibocsátási megjegyzések és README fájlok egészítik ki, amelyek fontos telepítési és működtetési utasításokat tartalmaznak. Ezek a fájlok általában abban a könyvtárban találhatók, amelyben a rendszer vagy a rendszer forráskódja.

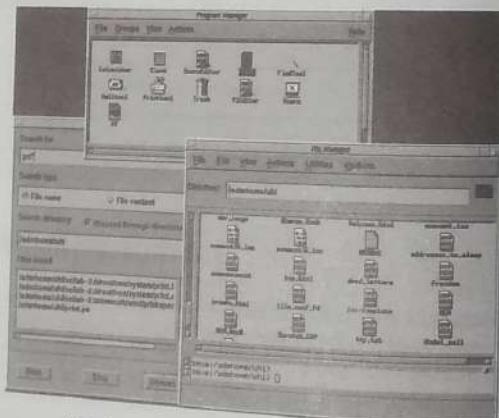
Az `/usr/src` könyvtárhoz például számos, a rendszerelemeket és a segédprogramokat tartalmazó alkönyvtár tartozik. Szinte valamennyi ilyen alkönyvtár saját README fájlokat is tartalmaz. Az `/etc/lilo` könyvtárban általában a Linux Loader (Linux betöltő, LILO) található, és ott lévő README fájl részletesen ismerteti a telepítést és a konfigurálást.

# ALKALMAZÁSOK

**A**z olyan operációs rendszer, amelyhez nem tartoznak alkalmazások, csak kevesek érdeklődésére tarthat számot. A Linux már régóta nem ilyen operációs rendszer. Az Internetről szabadon letölthető nagyszámú alkalmazás mellett a kereskedelmi forgalomban is széles választékuk kapható. Emellett a PC UNIX verziókra írt alkalmazások többsége az iBCS2 emulációval a Linux alatt is futtatható. A rendelkezésre álló alkalmazások közül csak egy kis válogatást tartalmaz a könyv.

## 12.1 PROGRAM- ÉS FÁJLKEZELŐK

A mai számítógépes rendszerek a fájlok és a programok kezeléséhez általában grafikus környezetet biztosítanak a felhasználóknak. A Linuxhoz ingyenesen kapható fájlkezelők általában a kereskedelmi verziók szűkített változatai. A *Freedom Desktop*, amely az összes általánosan használt UNIX platformon futtatható, Linux verziójával is rendelkezik (lásd a 12.1 ábrán). A program a legkényesebb igényeket is kielégíti. A jobb áttekinthetőség kedvéért a programkezelő lehetővé teszi a fájlok, programok és a shellszkriptek csoportosítását. A fájlokkal végezhető műveleteket a fájlkezelő teszi egyszerűvé. A programkezelő a fájlkezelő is lehetővé teszi az egérrel való áthúzás (*drag&drop*) használatát. A Freedom csomag számos egyéb segédprogramot is tartalmaz, így a *find* parancs grafikus kereprogramját és egy nyomtatáskezelőt.



12.1 ábra. A Freedom Desktop Linux változata

## 12.2 SZERKESZTŐK

A számítógépes rendszerek egyik legfontosabb eleme a szövegszerkesztő program. A Linux alatt közel egy tucat ilyen alkalmazás áll rendelkezésre. Sok ezek közül minden UNIX-rendszeren megtalálható, míg mások kifejezetten valamely adott rendszerhez tartoznak.

### **vi**

Minden UNIX-rendszerben általánosan használt szerkesztőprogram a **vi**. Ez a meglehetősen régi program a felhasználók kiszolgálása területén kétség kívül kivánnivalókat hagy maga után. Sok felhasználó nemtetszését váltja ki, hogy a **vi** különbséget tesz a beviteli és a parancsmód között. Ennek a megoldásnak persze előnyei is vannak. Például megismételhető a legutolsó parancs, és egy lépéssel lecserélhető a következő három szó egy új szóra.

Mivel a **vi** forráskódja nem áll rendelkezésre, egyre inkább egy **vi** klón használata terjed el a Linux alatt. Itt mindenki két változatról beszélhetünk, ezek az **elvis** és a **vim**, amelyek az eredeti szinte valamennyi parancsát emulálják, és emellett számos bővítményt is tartalmaznak.

### **sed**

A **sed** a szokásos értelemben nem szerkesztő, hanem vezérlőfájlban elhelyezett parancsokkal adatfolyam módosítására szolgál. A **sed** parancsai nagyon hasonlítanak a **vi** parancsaihoz, de ezeket nem interaktív módon viszik be, hanem a rendszer egy fájlból vagy a parancssorból olvassa be. A **sed** gyakran használt UNIX shellszkriptek vagy igen nagy méretű, más szerkesztőkkel nem beolvasható fájlok kezelésére.

### **joe**

A **joe** a „Joe's own editor” rövidítése. Ez a szerkesztő a DOS rendszerből áttérők körében meglehetősen népszerű, mivel parancsbillentyűinek kiosztása nagyon hasonlít a népszerű PC-s szövegszerkesztő, a **Wordstar** és a Turbo Pascal szerkesztőjének kiosztásához. A parancs- és a szerkesztési üzemmód nincs különválasztva. A kijelölt szövegblokk inverz módon jelenik meg a képernyón, ami a UNIX alatt nem általános. A **joe** szerkesztőben a képernyő több ablakra osztható fel, a bekezdések formázhatók, és elválasztási üzemmód is használható.

Ha a Wordstar parancsbillentyűhez szokott felhasználók hatékonyabb szövegszerkesztőt szeretnének használni, érdemes megismerkedniük a GNU **Emacs** szerkesztő Wordstar üzemmódjával.

### **xedit**

Az **xedit** az X Window System csomag része. A fenti szerkesztőkkel ellentétben az **xedit** nem korlátozódik az ASCII környezetre. Bár az **xedit** grafikus felhasználói kezelőfelület alatt fut, az elváthatóhoz képest nem nyújt elegendő kényelmet: a rendelkezésre álló parancsok az Athena szövegelemelek lehetőségeire korlátozódnak. Bár a szerkesztő sohasem fog igazán széles körben elterjedni, kisebb feladataik végrehajtására megfelelő.

### **axe**

Az **axe** egy újabb olyan szerkesztő, amely kizártlag az X Window System alatt fut. Ez is az Athena szövegelemeket használja, de az **xedit** szerkesztőnél lényegesen több szolgáltatást biztosít. A felhasználónak lehetősége van különböző ablakokban tetszőleges számú szövegfájl megnyitására, egy fájlkijelölő párbeszédpanelen interaktív módon megadhatók a megnyitni vagy men-

teni kiáltott fájlok, szövegrészletek kereshetők és cserélhetők, valamint a bekezdések formázása is végrehajtható. A rendelkezésre álló parancsokról egyszerű súgóleírások jeleníthetők meg. Az axe sokoldalú, felhasználóbarát szerkesztőnek bizonyult a Linux alatt X Window rendszeren.

### **xcoral**

Az **xcoral** szerkesztőhöz is X Window System környezet szükséges (lásd a 12.2 ábrán). Bár az **xcoral** nem szabványos eszközökészletet használ, menüsora és a görgetősáv kényelmes használatot biztosít. Az **xcoral** különösen a C és a C++ programozók érdeklődésére tarthat számot, mivel beépített függvény- és osztálykeresővel rendelkezik. Az **xcoral** elindításakor átvizsgálja az összes C/C++ fájlt, és ábécé sorrendben megjeleníti a bennük található azonosítókat. A tallózóból a felhasználó ezután egyszerűen átérhet a forráskód megfelelő helyére. Ez különösen megkönyvíti a programozóknak a már létező forráskód áttekintését. Új programok írásához a programozó olyan üzemmódot választhat, amely egységes formázást használ, és amelyben függvény- és osztályablakok hozhatók létre. Az **xcoral** billentyűparancsai nagyon hasonlítanak az Emacs szerkesztőben használhatókhöz.



12.2 ábra. Az **xcoral**

### **asedit**

Az **asedit** a Motif elemkészletet használja, amely elsősorban egységes kezelőfelületével tűnik ki. Bár a szerkesztő rendelkezik a legfontosabb parancsokkal (szövegfájlok betöltése és mentése, keresés és csere), túl sok kényelmet nem nyújt. Külön említtést érdemel viszont a hypertext rendszerű súgója.

### **Az Emacs és változatai**

A legnépszerűbb UNIX alatti szerkesztő minden bizonnal az Emacs, illetve ennek változatai. Ennek legfőbb oka az, hogy egyszerű ASCII terminálon és grafikus X11 környezetben is futtatható.

A **Emacs** a Lucid vállalatnál kifejlesztett népszerű Emacs változat. Alapjául az Emacs 19 egy korai verziója szolgált. A **Emacs** szerkesztőt elsősorban megjelenése és különleges Lisp függvények szintaxisának kezelése emeli ki.

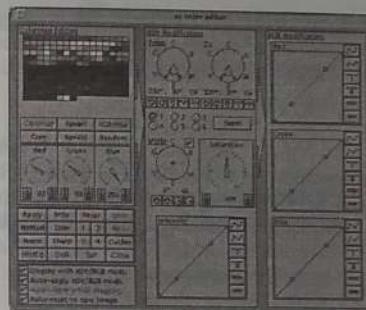
Az Emacs 19-es verziójának megjelenésével az egyéb Emacs verziók, például **epoch**, jelentősége erősen csökkent. A GNU Emacs részletes ismertetését a következő fejezet tartalmazza.

### 12.3 GRAFIKUS PROGRAMOK

Az X Window System alatt számos grafikus program áll rendelkezésre. Funkcióik a vektorgrafikus rajzolástól a képfeldolgozásig és képkonvertálásig terjednek. Csak kevés programnak van különleges Linux változata, többségük közvetlen Linux alatti fordítással használható. A Linux-változatok többsége tartalmazza a legfontosabb grafikus programokat.

**xv**

J. Bradley xv programja, amely minden típusú grafika megjelenítésének és kezelésének hatékony eszköze, nagy népszerűségnek örvend a UNIX világában. Az szokásos FTP kiszolgálókon érhető el, szabadon terjeszthető, de nem ingyenes program. Számos hatékony szolgáltatása ellenére könnyen kezelhető, amihez jól megtervezett kezelőfelülete is hozzájárul. Az xv kezeli tudja az összes elterjedt grafikus formátumú fájlt, így a GIF, a TIFF, a PostScript, a JPEG és a PBM formátumot (lásd a 12.3 ábrát).



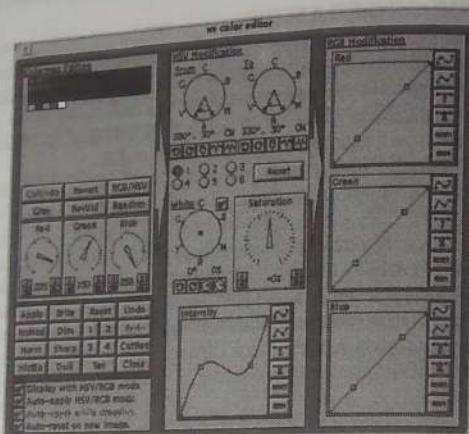
12.3 ábra. Az xv

A betöltendő grafika egy fájlkijelölő menüből, illetve a 3.00-s verzió óta egy integrált fájlkezelő rendszerből választható ki (utóbbiban a kijelölt fájlok kisméretű képe is megtekinthető). A betöltött grafika szabadon méretezhető. A grafikus műveleteket tartalmazó menü számos paraméter, így a fényerő, a kontraszt és a tónus megváltoztatását lehetővé teszi gombok és beállító csúszka segítségével, valamint az egérrel módosítható görbékkel.

Az xv programba betöltött grafika számos módszerrel módosítható, így például előre definiált algoritmus szerint is, ilyen az „Oil Painting” (festés) vagy az „Emboss” (domborítás). A grafika számos módon megjeleníthető háttérként is, és ez a programból való kilépés után is a képernyőn maradhat.

#### **xfig**

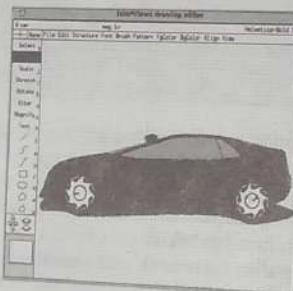
Az xfig, a Linux alatt vektorgrafikus ábrák létrehozására szolgáló program a szokásos rajzolási cszközöket kínálja, emellett különböző betűtípusú szöveg beágyazására is lehetőséget nyújt (lásd a 12.4 ábrán). Egyik érdekes szolgáltatása a grafikák importálása különböző formátumban. Saját formátuma mellett használhatók a szabványos formátumok is, így a PostScript, a HPGL és a LaTeX változatok.



12.4 ábra. Az xfig rajzolóprogram

### **idraw**

A Stanford Egyetem InterViews csomagja tartalmaz egy hatékony, vektorgrafikus rajzolóprogramot is: ez az **idraw** (lásd a 12.5 ábrán). A szokásos grafikus elemek (vonalak, téglalapok, körök és Bézier görbék) mellett a különböző betűtípusú szöveg forgatását is lehetővé teszi.



12.5 ábra. Az idraw

Ez a program csak a PostScript fájiformátum használatát teszi lehetővé, és a szükséges hardver szempontjából meglehetősen igényes (386-os gépeken nem fut elvezethető módon).

### **xpaint**

Az **xpaint** az Apple Macintosh MacDraw alkalmazáson alapuló, grafikák létrehozására és kezelésére szolgáló program. Az **xpaint** különböző ablakokban több grafika kezelésére alkalmas. Különböző rögzítőkőzei segítségével tetszőleges ábra kialakítható: nem csak egyszerű elemek (vonalak és körök), hanem szabadkézi görbék rajzolására is használható (lásd a 12.6 ábrán).

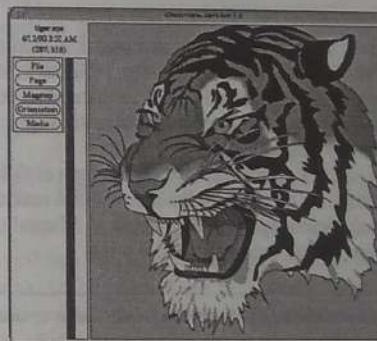


12.6 ábra. Az xpaint

A kitöltő mintázatok szerkesztőjével egyéni mintázatok alakíthatók ki. A beállítható nagyítás gyakorlati jelentőségű: a grafika adott részletének nagyított képe külön ablakban jeleníthető meg. Fontos jellemző, hogy a rajzeszközök a nagyítási ablakban is használhatók. Az xpaint betűkészlet tallózója számos font használatát kínálja, köztük az X11 R5 és R6 alatt rendelkezésre álló méretezhető fontokat is.

### **Ghostscript/Ghostview**

A Linux-rendszer fontos eleme a Ghostscript (gs) PostScript értelmező. A Ghostview grafikus kezelőprogram segítségével PostScript fájlok kényelmes megjelenítését teszi lehetővé (lásd a 12.7. ábrát). A Ghostscript nemcsak a képernyőre képes dolgozni, hanem mátrix vagy tintasugaras nyomtatókat is képessé tesz PostScript fájlok kinyomtatásra.



12.7 ábra. A PS grafikát megjelenítő Ghostview

Amikor színes grafika megjelenítésére kerül sor monokróm eszközön, a Ghostscript árnyalási technikával éri el a megfelelő minőséget. A Ghostscript jelen verziójában a PostScript Level 2 használható.

A szabványos Adobe betűkészletek elemeit tartalmazó szöveg kezelésére a Ghostscript hasonló betűkészlet-csomagot tartalmaz, amelynek minősége azonban nem megfelelő. A Ghostscript egyébként lehetővé teszi az eredeti betűkészletek telepítését és használatát is.

### Az MPEG videolejátszó

Az Interneten rendelkezésre álló nagy mennyiségű MPEG videoanyag megtekintése a Linux alatt a Kaliforniai Egyetemen, Berkeleyben kifejlesztett MPEG Player segítségével történhet (lásd a 12.8 ábrán). Fontos, hogy ehhez nincs szükség külön hardverre, és a kimenet másik képernyőre is átirányítható. A lejátszás sebességét viszont nagy mértékben befolyásolja a számítógép teljesítménye és a grafikus kártya.



12.8 ábra. Az MPEG videolejátszó

Az Interneten néhány videofájl az Apple Quicktime formátumát használja. Ezek lejátszása az xanim programmal lehetséges.

## 12.4 SZÖVEGFELDOLGOZÁS

A Linux alatti legtöbb szövegfeldolgozó rendszer nem igazi szövegszerkesztő. A szöveg formázása általában a megfelelő helyen az ASCII formátumú szövegekbe beszűrt vezérlőszekvenciákkal történik. Első látásra ez a megoldás kényelmetlennek tűnik, de főleg nagyobb méretű dokumentumoknál számos előnye is lehet.

### groff

A UNIX alatt formázott dokumentumok előállításának klasszikus módja az ASCII szövegfeldolgozása az `nroff` parancssal. Ez tulajdonképpen szöveg, táblázatok, képletek és egyszerű grafikák formázására szolgáló csomag. A felhasználó ASCII formátumban beírja a szöveget, míg a formázást és az elrendezést megfelelő parancsok segítségével hajtja végre. A kimenet lehet csak szöveg megjelenítésére szolgáló vagy grafikus eszköz.

A UNIX Manual oldalak készültek ily módon. A segédprogram elsődleges felhasználási területe a Manual oldalak megjelenítése. Az `nroff` szolgáltatásai különböző makrókkal és a különböző feladatokra rendelkezésre álló makrócsomagokkal bővíthetők. A Manual oldalak formázására külön makrófájl (`.an`) áll rendelkezésre. A parancs a következő lehet:

```
nicetry:~$ zcat /usr/man/man1/ls.1.gz | nroff -man | more
```

A Linux az eredeti `nroff` helyett egy bővíst tartalmaz. Az `nroff` parancs csak egy szkript, amely az ASCII kimenet számára megfelelő paraméterekkel a `groff` parancsot hívja. Ha a `groff` hívása e paramétereink nélkül történik, akkor a kimenet PostScript szöveg lesz. Ez hasznos lehet a Manual oldalak nyomtatásakor PostScript kompatibilis nyomtatókon, mivel a szöveg megfelelően formázott és különböző betűtípusokat tartalmaz.

```
nicetry:~$ zcat /usr/man/man1/ls.1.gz | groff -man >ls.ps
```

## TEX

A Donald E. Knuth által kifejlesztett TeX (kiejtése „teh”) kiadványszerkesztő külön világot alkot a hasonló rendszerek körében. A korszerű WYSIWYG („azt kapod, amit látsz”) szövegszerkesztő rendszerekhez szokott felhasználó az első találkozás alkalmával megdöbbeni, és úgy hiszi, hogy a számítógépes kőkorszakba került vissza. Mindemellett azonban kétségtelen, hogy számos szolgáltatása a szövegfeldolgozók elvonalaiból emeli. A rendszer egyik legnagyobb előnye, hogy szinte az összes számítógépes platformon **rendelkezésre áll**, így a létrehozott szöveg „hordozható”.

A TeX ASCII formátumú különleges formázóparancsokat tartalmazó szöveget dolgoz fel. A hatékony program különösen alkalmas matematikai szöveg szedésére és tördelésére, de a szokásos szöveg minősége is általában meghaladja a többi szövegfeldolgozó/szövegszerkesztő programmal kapott eredményt. A TeX az ASCII bemeneti fájlból eszközfüggetlen DVI (device independent) fájlt készít, amely ezután megjeleníthető a képernyón vagy nyomtatható. A DVI fájl formátuma az összes számítógépes rendszeren azonos, így a TeX fájlok problémamentesen átvihetők a rendszerek között. Ezek a fájlok akár Linotronic levilágítóra is küldhetők.

A TeX kiegészítéseként számos makrócsomag és segédprogram áll rendelkezésre Linux alatt. Ezek közé tartozik egy grafikus **megtekintő** (`xdvif`), egy (bibliografikus-) indexfájlok rendezésére szolgáló segédprogram, és a hiányzó betükészletek előállítására szolgáló program. Számos kezelőprogram létezik a DVI fájlok PostScript rendszerre konvertálásához (`dvi2ps`), illetve a fájlok nem PostScript (mátrix, tintasugaras, lézer) nyomtatóra küldéséhez. A grafikák LaTeX parancsokkal vagy külsőleg létrehozott PostScript fájlokkal ágyazhatók.

Az egyik legismertebb makrócsomag a **LaTeX** – készítője Leslie Lamport –, amely jelentősen megkönyíti a munkát a TeX programmal. Természetesen a nemzeti karaktereket tartalmazó szövegkezelésre is születtek megfelelő verziók. A LaTeX segítségével automatikusan előállítható a dokumentumok tartalomjegyzéke és tárgymutatója, valamint egyszerűsödik a táblázatokkal és a listákkal való munka is. A következő példa egy LaTeX bemeneti fájlt mutat be:

```
\documentstyle{article}
\topmargin -15mm \headsep 0mm \textwidth 16cm
\textheight 26cm \oddsidemargin 0cm \parindent 0mm

\begin{document}
\thispagestyle{empty}

\centerline {(\Huge Sz\"ovegform\'az\'as \TeX\ rendszerrel)}
\vspace{1cm}

A \TeX\footnote{kiejt\'ese megfelel a magyar technika sz\'o el\H{o} h\H{a}rom bet\H{H} uj\'enek}\ program \'es a \LaTeX\ makr\'ocsomag
seg\'i ts\'eg\'evel nyomda\H{H} os\H{H} min\H{H} os\H{H} u k\'eziratok
k\'esz\'i thet\H{H} ok. Spe\H{c}i\'al\'lis lehet\H{H} os\H{H} egi egyszer\H{H} uv\'e
teszik cikkek, k\'onyvek, levelek \'es matematikai
t\'argy\'u anyagok k\'esz\'i t\'es\H{v}et. Ezen bel\H{v}ul a \LaTeX\
seg\'i ts\'eg\'evel egyszer\H{H} uen lehet k\'epleteket, t\'abl\'azatokat
\'es list\'akat megform\'azni. A tartalomjegyz\'ek \'es a
tudom\'anyos ig\'enyeknek megfelel\H{H} o fejezetek\H{H} amoz\'as is automatikusan
k\'esz\'ul. M\'eg a l\'abjegyzetek kezel\'ese sem okoz probl\'em\H{v}at.
T\'obbf\'ele bet\H{H} uk\'eszlet \'es el\H{H} ore defini\'alt st\'iluslap
seg\'i ti a sz\"ovegr\'eszek kiemel\'es\H{v}et:
```

```
\begin{center}
(\rm Roman), (\bf Bold Face), (\tt Typewriter), (\it Italic),
(\sl Slanted), (\sc Small Caps), (\sf Sans Serif)
\end{center}
```

A szó "oveg m'ere te v'altoztathat'o:

```
\begin{center}
(\tiny legkisebb), (\scriptsize nagyon kicsi), (\footnotesize kisebb),
(\small kicsi), (\normalsize normál), (\large nagy), (\Large m'eg nagyobb) \\
(\LARGE legnagyobb), (\huge legeslegnagyobb), (\Huge gigantikusan nagy)
\end{center}
```

A matematikai képletek v'i zl'esesen kiszéphetők:

```
\begin{displaymath}
\int_0^{\infty} f(x) dx \approx \sum_{i=1}^n w_i e^{x_i} g(x_i)
\end{displaymath}

\begin{displaymath}
\sqrt[n]{\frac{x^n - Y^n}{1 + u^{(2n)}}}
\end{displaymath}
```

Egy példa k"ovetkezik a lista:

```
\begin{itemize}
\item Hardver
\begin{itemize}
\item számítógép \item billentyűzet \item monitor
\end{itemize}
\item Szoftver
\begin{itemize}
\item operaciós rendszer \item felhasználói interfész
\item felhasználói program
\end{itemize}
\end{itemize}
```

```
\LaTeX -ben k"ülönösen egyszerűtől azatokat készíti teni:
\begin{center} \begin{tabular} {l|c|c|c|c} \hline
Helyezés & Csapata & M & NY & D & V & Gölörök & Pont \\ \hline \hline
1. & Bayern Munich & 33 & 19 & 13 & 1 & 66:31 & 51:15 \\ \hline
2. & Hamburg & 33 & 18 & 9 & 6 & 65:37 & 45:21 \\ \hline
3. & Bor. M'Gladbach & 33 & 17 & 7 & 9 & 70:44 & 41:25 \\ \hline
4. & Bor. Dortmund & 33 & 14 & 10 & 6 & 66:50 & 38:28 \\ \hline
5. & Werder Bremen & 33 & 16 & 6 & 11 & 63:53 & 38:28 \\ \hline
6. & Kaiserslautern & 33 & 15 & 7 & 11 & 64:47 & 37:29 \\ \hline
\end{tabular} \end{center}
```

```
\newcommand{\kethasab}{%
\begin{minipage}[b]{7.5cm}
A sz\"oveg text box-ba illeszthet\H{o}. Term\'esztesen a \TeX\/
k\'epes automatikus elv\'alaszt\'asra. Ezenk\'i v\'ul a \TeX\ a bet\H{u}ket
egym\'as alv\'a igaz\`i tja, ha sz\"uks\'eges. Ezt az elj\'ar\'ast
(\em kerning)-nek h\'i vj\'ak.
\end{minipage}}%}

\kethasab \hfill \kethasab

\begin{center}
{\Huge W\orld V\at V/A W\orld}
\end{center}

\begin{center}
{\Huge World Vat VA World}
\end{center}
\end{document}
```

A fordítás után az eredmény az xdvi parancssal jeleníthető meg; a fenti fájl eredménye a 12.9. ábrán látható.

## Szövegformázás $\text{\TeX}$ rendszerrel

A  $\text{\TeX}$ <sup>1</sup> program és a  $\text{\LaTeX}$  makrócsomag segítségével nyomdakész minőségű kéziratok készíthetők. Speciális lehetőségei egyszerűvé teszik cikkek, könyvek, levelek és matematikai tárgyú anyagok készítését. Ezen belül a  $\text{\LaTeX}$  segítségével egyszerűen lehet képleteket, táblázatokat és listákat megformázni. A tartalomjegyzék és a tudományos ígyenyelek megfelelő fejezetszámozás is automatikusan készül. Még a lábjegyzetek kezelése sem okoz problémát. Többféle betükészlet és előre definiált stíluslap segíti a szövegrészeken kiemelést:

Roman, **Bold Face**, *Typewriter*, *Italic*, Slanted, SMALL CAPS, Sans Serif

A szöveg mérete változtatható:

legkisebb, nagyon kicsi, kisebb, kicsi, normál, nagy, még nagyobb  
legnagyobb, legeslegnagyobb, gigantikusan nagy

A matematikai képletek ízlésesen kiszedhetők:

$$\int_0^{\infty} f(x) dx \approx \sum_{i=1}^n w_i e^{x_i} g(x_i)$$

$$\sqrt{\frac{x^n - Y^n}{1 + u^{2n}}}$$

Egy példa következik a listára:

- Hardver
  - számítógép
  - billentyűzet
  - monitor
- Szoftver
  - operációs rendszer
  - felhasználói interfész
  - felhasználói program

LaTeX-ben különösen egyszerű táblázatokat készíteni:

Helyezés	Csapat	M	NY	D	V	Gólarány	Pont
1.	Bavaria Munich	33	19	13	1	66:31	51:15
2.	Hamburg	33	18	9	6	65:37	45:21
3.	Bor. M'Gladbach	33	17	7	9	70:44	41:25
4.	Bor. Dortmund	33	14	10	9	66:50	38:28
5.	Werder Bremen	33	16	6	11	63:53	38:28
6.	Kaiserslautern	33	15	7	11	64:47	37:29

A szöveg text box-ba illeszthető. Természetesen a TeX képes automatikus elválasztásra. Ezenkívül a TeX a betüköt egymás alá igazítja, ha szükséges. Ezt az eljárást *kerning*-nek hívják.

A szöveg text box-ba illeszthető. Természetesen a TeX képes automatikus elválasztásra. Ezenkívül a TeX a betüköt egymás alá igazítja, ha szükséges. Ezt az eljárást *kerning*-nek hívják.

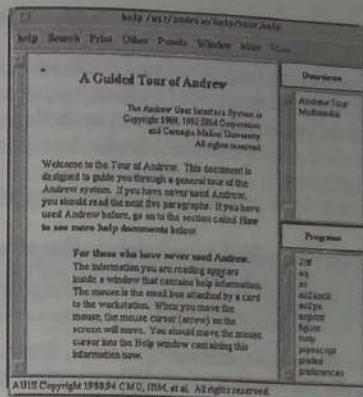
World Vat VA World  
World Vat VA World

\*kiejtése megfelel a magyar technika szó előző három betűjének

12.9 ábra. Egy TeX fájl eredménye

## 12.5 AZ ANDREW MULTIMÉDIÁS KÖRNYEZET

A Carnegie-Mellon Egyetem és az IBM több év óta folyó fejlesztése egy hatékony, elosztott multimédiás felhasználói környezet létrehozására irányul; ennek elnevezése **AUIS** (Andrew User Interface System). Ez számos egyedi alkalmazást foglal magában, amelyek multimédiás dokumentumokfeldolgozását leírhatják (lásd a 12.10 ábrán).



12.10 ábra. Az Andrew csomag

Az alkalmazások közé tartozik egy egyszerű szövegfeldolgozó (ez), egy táblázatkezelő (table), egy rajzolóprogram (figure) és egy festőprogram (image). Az egyes elemeket egy hypertext kialakítású súgó (help Andrew) dokumentálja.

Az elkészített dokumentumok elküldését egy multimédiás levelezőprogram teszi lehetővé. Szükség szerint egy dokumentumon belül az egyes elemek (szöveg, grafika, táblázatok, animáció) kombinálására is lehetőség van. Ezt az eljárást gyakran objektumok beágyazásának nevezik. Az AUIS alatt az egyedi objektumok elnevezése inset (betét).

## 12.6 ADATBÁZISOK

A Linux alatt futó adatbázisok itt következő listája csak egy kis válogatás a folyamatosan bővülő csoportból. Még a SCO UNIX számára kifejlesztett rendszerek – például a Foxpro for UNIX (Microsoft) és egy kereskedelmi Ingres verzió – is futtathatók a Linux alatt az IBCS2 emulátor segítségével. Természetesen számos, kifejezetten a Linux számára kidolgozott termék is rendelkezésre áll. A Clipper-kompatibilis Flagship azok számára lehet érdekes, akik a PC-k területéről szeretnének szoftvert átvinni a Linux alá.

### MSQL

Az MSQL (mini SQL vagy Minerva SQL) fejlesztője az ausztráliai Bond Egyetemről David J. Hughes. Ez egy minimális adatbázismag az ANSI SQL szabvány részhalmazának megvalósítására. A rendelkezésre álló adattípusok jelenleg a karaktersorozatokra és az egész vagy lebegőpontos számokra korlátozódnak. Sajnos az MSQL jelenleg nem ismeri fel a Nézet (View) táblákat. A rendszer azonban valószínűleg elegendő kisebb alkalmazások kifejlesztéséhez. Teljesítményége lényegesen jobb az egyéb szabadon letölthető adatbázis-rendszereknél. Emellett az MSQL kiszolgáló TCP/IP kapcsolaton keresztül is elérhető, ami lehetővé teszi valódi ügyfel/kiszolgáló architektúrák felépítését. Tcl/Tk kiegészítés szintén rendelkezésre áll.

**OBST**

Az OBST objektumorientált adatházist (object-oriented database) a STONE (a Structured and Open Environment) projekt keretében fejlesztették ki a Forschungszentrum Informatik (FZI) intézetben, a németországi Karlsruheban. Jóllehet a rendszert eredetileg nyelvfüggetlen környezetre szánták, az adatházist leíró nyelv erősen a C++ felé irányul. A programfejlesztés azonban végre hajtható a C, a C++ és a Tcl nyelven is.

Emellett az alapvető adminisztrációs feladatokra egy egyszerű grafikus adatbázisszerkesztő is rendelkezésre áll.

Egy karlsruhei szoftvercég jelenleg az OBST kereskedelmi változatának fejlesztésén dolgozik.

**Ingres és Postgres**

Az Ingres adatbázisrendszer és utódja, a Kaliforniai Egyetemen, Berkeleyben kifejlesztett Postgres a Linux alatt is rendelkezésre áll. Nem szabad azonban összetéveszteni az Ingres ezen verzióját a kereskedelmi SQL adatbázisrendszerrel. A Linux verzió lényegesen régebbi, és lekérdezési nyelve is nemileg más, mint az SQL rendszeré.

A Postgres objektumorientált adatházis, de eltérően a hasonló rendszerektől, nem a C++ nyelven és állandó objektumokon alapul. Ez inkább a klasszikus, relációs megközelítés továbbfejlesztésének tekintethető.

Mindkét rendszer érdeklődésre tarthat számot az oktatásban, de az Ingres alkalmazás-fejlesztési feladatokra már csak korlátozottan használható, míg a Postgres legújabb verziója, a Postgres95, amelyet szabvány SQL kereső nyelvvel látta el, alkalmas lehet komoly rendszerekhez is.

**YARD**

Akinck a Linux alatt teljes SQL adatbázisra van szüksége, érdemes megfontolnia a németországi Kölönbén, a Yard Software által kifejlesztett termékek használatát. A YARD-SQL az ANSI szabványnak megfelelő teljes adatbázisrendszer. Az ESQL-C portot kivéve biztosítja a megbízható tranzakciókat, a hivatkozási integritást és a BLOB (Binary Large Object) adattípus használatát.

A grafikus szerkesztő az adatbázis tervezésében használható fel. Az ODBC port az MS-Windows alatt PC-s ügyfelek csatlakozását biztosítja.

**Just Logic SQL**

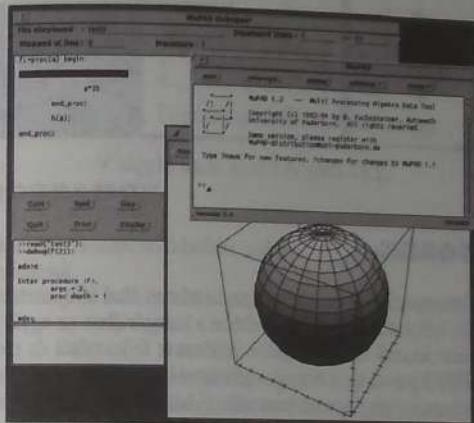
A Just Logic Technologies az olcsóbb kategóriába eső ügyfél/kiszolgáló rendszerű SQL adatbázist fejlesztett ki. A kiszolgáló az összes általánosan használt PC-s UNIX platformon futtatható. Az ügyfélsoftverek DOS, Windows és OS/2 operációs rendszeren is rendelkezésre állnak. Az ODBC port és egy Apple ügyfél fejlesztés alatt áll.

**POET**

Egy másik Linux alatti kereskedelmi adatbázisrendszer a POET. Ez az adatbázisrendszer a C++ nyelven alapul és objektumorientált; emellett az összes általánosan használt operációs rendszeren megtalálható (UNIX, OS/2, MS-Windows), és ügyfél/kiszolgáló rendszerek megvalósításában is jól használható. Mivel a POET rendszert elsősorban C++ programozóknak szánták, felhasználói kezelőfelülete meglehetősen kezdetleges.

## 12.7 MATEMATIKAI ALKALMAZÁSOK

Az egyetemi hallgatók számára különös érdeklődésre tarthat számot a MuPAD programcsomag (lásd a 12.11. ábrán). A számítógépes algebrai rendszert a Paderborn Egyetemen (Németország) fejlesztették ki, és a nem profitorientált intézmények ingyenesen használhatják. A MuPAD a felhasználót a különböző típusú matematikai problémák megoldásában segíti. Integrált programozási nyelve lehetővé teszi a felhasználó saját algoritmusának a bevitelét. A MuPAD rendszert elsősorban párhuzamos problémák megoldására terveztek.



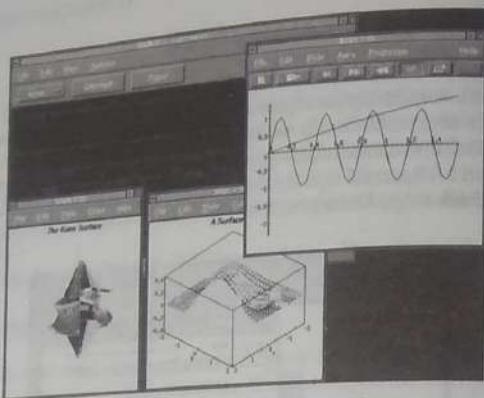
12.11 ábra. A MuPAD számítógépes algebrai rendszer

A MuPAD grafikus kezelőfelülettel rendelkező saját hibakeresőt tartalmaz. A MuPAD önmagában egy egyszerű parancssor-orientált felhasználói kezelőfelület, azonban egy OpenLook- vagy egy Motif-alapú kezelőfelület lényegesen egyszerűbbé teszi a használatát. A MuPAD fejlett grafikus kimeneti szolgáltatásokkal rendelkezik (például háromdimenziós függvények felrajzolása).

A hypertext rendszerű súgó az összes szükséges információt és függvénylefrást biztosítja. A MuPAD részletes kézikönyve a Birkhäuser kiadónál jelent meg.

### Maple V

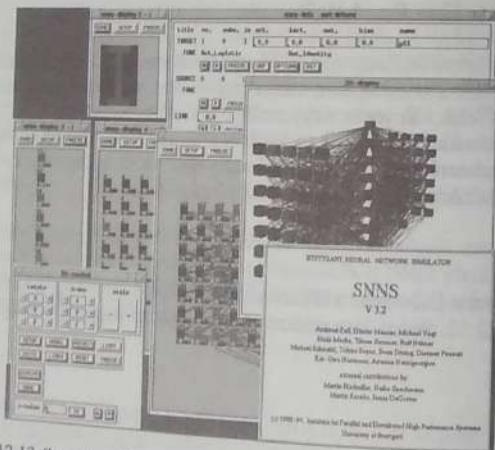
A Waterloo Software elkészítette a közkedvelt Maple V programcsomag 3-as verziójának Linux változatát (lásd a 12.12. ábrán). Ez a rendszer számos felhasználó igényeit kielégíti, de meglehetősen drága.



12.12 ábra. A Linux alatti Maple V

## 12.8 SZIMULÁTOROK

A Stuttgarti Egyetem (Németország) számítógépes szakemberei által kifejlesztett **SNNS** (Simulator for Neural Networks, neurális hálózatok szimulátora) ebben a kategóriában az egyik legjobb rendszer (lásd a 12.13. ábrán). A grafikus kezelő az összetettebb hálózatok fejlesztését és elemzését is lényegesen egyszerűbbé teszi. Az SNNS nemcsak a neurális hálózatokban végbemenő folyamatok bemutatására alkalmas, hanem lehetővé teszi a ténylegesen használható hálózatokon való gyakorlást is.

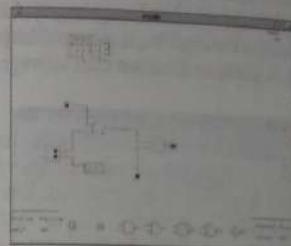


12.13 ábra. Neurális hálózatok szimulációja az SNNS rendszerrel

### Áramkör-szimuláció

A digitális és analóg elektronikus áramkörök szimulációja hajtható végre a Caltech Electronic CAD programjával (lásd a 12.14. ábrán). A digitális áramkörök felépítése és ellenőrzése a **diglog**

programmal valósítható meg. A programhoz alkatrészek és áramköri elemek széles választékát tartalmazó könyvtár is tartozik.



12.14 ábra. Digitális áramkörök szimulációja

## 12.9 JÁTÉKPROGRAMOK

A kemény napi munka utáni kikapcsolódáshoz számos, a Linux alatt is rendelkezésre álló játékprogram közül választhatunk.

### Xteddy

Az Xteddy nem a szokásos értelemben vett játékprogram (lásd a 12.15. ábrán). Különösen népszerű a család azon tagjai körében, akiket (még) nem fertőzött meg teljesen a számítógép.



12.15 ábra. Az elektronikus mackó

### Tetris

A Tetris igen népszerű játékprogram, amely rendkívül vonzó formában a Linux alatt is használható. A játékosnak a lehulló különböző mértani elemeket kell úgy elrendeznie, hogy a halom ne nőjön túl magasra. Ez persze hamar bekövetkezne, ha a teljes sorok alul nem tűnnének el.

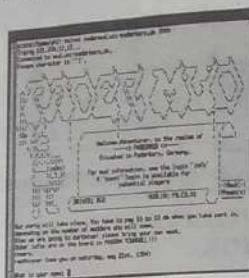
### A GNU sakk

Nagyobb szellemi kihívást jelent az xboard és a GNU Chess együttese. A program képessége lenyűgöző: sakiversenyeken a GNU Chess számos sakkprogramot legyőzött már.

Önmagában a GNU Chess nem túlzottan felhasználóbarát, mert a gyakorlati játékhoz az xboard grafikus kezelőfelület is szükséges. Az xboard segítségével a GNU Chess kétszer is elindítható, így az egymás elleni játék is megvalósítható. További lehetőség: sakkjáték az Interneten keresztül. Itt megkereshető a megfelelő partner bárhol a világban, éjjel és nappal, minden időpontban (lásd a 10.3. szakaszat).

### MUD és Crossfire

A többrésznevős, hálózaton keresztül játszható játékok nagy népszerűségnek örvendenek. Ezek egyik széles körben elterjedt formája a MUD (Multi-User Dungeons, lásd a 12.16. ábrán). minden játékos a Telnettel keresztül bejelentkezik az azonos kiszolgálóra, ahol használhatja például a következő parancsokat: say, get vagy go, és mozoghat. A játékban általában csak szöveges bevitel és kivitel lehetséges.



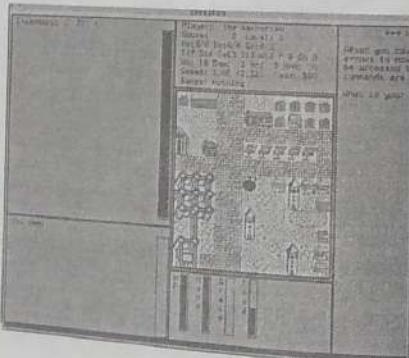
12.16 ábra. MUD

A Telnettel például a következő címeken érhető el a játék (lásd a 10.3. szakaszt):

- morgen.cs.tu-berlin.de, Port 7680
- padermud.uni-paderborn.de, Port 3000
- pascal.uni-muenster.de, Port 4711
- infosgi.rus.uni-stuttgart.de, Port 3333
- mud.uni-muenster.de, Port 4711

További információ a témáról szóló számos hírcsoportból vagy FAQ kiadványokból szerezhető be (lásd a 11.4 szakasz).

A szöveges MUD játéknál egy lépéssel tovább megy a többfelhasználós **Crossfire**, amelyhez speciális ügyfélszoftver szükséges, amely azt a képerletbeli világot ábrázolja, amelyben a játékos mozoghat (lásd a 12.17. ábrán). A játék forráskódja az <ftp://ifi.uio.no/FTP/kiszolgalo/pub/crossfire> könyvtárából letölthető.



12.17 ábra. A Crossfire

# A GNU EMACS

**H**a megkérdeznénk több gyakorlott UNIX felhasználót, hogy melyik szerkesztőprogramot használják, legtöbbjük a vi vagy az Emacs szerkesztőt említené. A legtöbb vi szimpatizáns választásáról alapozná, hogy már 20 éve használja a programot, és kiemelné, hogy az összes UNIX platformon elérhető. Az Emacs felhasználói viszont szerkesztőjük nagysokú rugalmasságát és funkcionálisitását hangsúlyozná.

Ez a fejezet betekintést nyújt az Emacs alkalmazásába, és részletesen ismerteti jellemzőit és szolgáltatásait. Külön szakasz szól az Emacs Lisp programozási kérdésekről; ebben példák segítségével ismertjük a konfigurálást. Az utolsó szakasz talán nehezebben lesz érthető azok számára, akik még nem szerezték programozási ismereteket más programozási nyelvekben. Az ennél részletesebb ismertetés azonban messze meghaladná e könyv kereteit és lehetőségeit. Ezt a részt tehát az alapvető programozási ismeretekkel rendelkezőknek szánjuk.

## 13.1 ÁTTEKINTÉS

Az Emacs fejlesztője Richard M. Stallman, az FSF alapítója. A program szinte az összes UNIX platformon, DOS rendszeren és VMS alatt rendelkezésre áll. Az Emacs nem csak egyszerű ASCII terminálokon, szöveges üzemmódban, hanem az X Window System alatt is használható. Ez a rendszer legördülő menüket, parancsgombokat és görgetőszávokat tartalmaz, valamint számos szín és betűkészlet is rendelkezésre áll. Az X11 alatt már megszokott módon az egérrel kijelölt szöveg másolható vagy beszúrható.

Az Emacs a kezdők számára **oktatónyelvet** tartalmaz, emellett a felhasználókat súgó segíti, amely az összes parancsot, billentyűkombinációt és belső változót ismerteti. A teljes dokumentáció a hypertexthez hasonló formátumban lehívható.

Az Emacs szerkesztőben egyidejűleg több szövegen dolgozhat a felhasználó különálló vagy felosztott ablakokban. Rendszeres időközönként elkészít a megnyitott dokumentumok biztonsági másolatát. Különös említést érdemel a végrehajtott műveletek szinte korlátlan szintű visszavonási lehetősége. A felhasználó egyetlen billentyűvel, megadott határig tetszőleges számú változtatást visszavonhat.

A vi és a többi közismert szerkesztővel ellentétben az Emacs nem pusztán szerkesztőprogram. Saját Lisp-alapú nyelvéhez tartalmaz egy értelmezőt is, amely a felhasználók rendelkezésre áll. Szintén valamennyi, a nem egyszerű szerkesztési műveletek körébe tartozó funkciót ezen az Emacs Lisp (vagy röviden Elisp) nyelven valósítottak meg. Létezik például olyan Emacs Lisp program, amely levelezési előfeldolgozót, egy hírolvasót vagy egy teljes fejlesztői környezetet valósít meg.

Képességeinek megfelelően az Emacs programcsomag meglehetősen nagy méretű. A szokásos telepítése mintegy 20 MB területet igényel a merevlemezen. Ez a telepítés azonban nemcsak számos Emacs Lisp programot, hanem a teljes dokumentációt is tartalmazza. Emellett az Emacs viszonylag sok memóriát is igényel, így kisebb kiépítésű gépen a használata nem olyan kényelmes.

Az Emacs számos rendelkezésre álló fő üzemmódjával a felhasználó adott feladatokra optimalizálhatja a szerkesztőt. A fő üzemmódok a programozás vagy a szerzőt segítő például a (formázási) szöveg formázásában. A C és a Lisp üzemmód az adott szinteknek megfelelő automatikus behúzásokat tartalmaz. Amikor a felhasználó egy záró zárójelet ír be, a megfelelő nyitó zárójel

automatikusan kiemelten jelenik meg. A kulcsszavak, a megjegyzések és az egyéb nyelvi elemek különböző betűtípusnal és színnel jeleníthetők meg. Ezek az üzemmódok az összes elterjedt programozási nyelvre és fájltípusra rendelkezésre állnak, és egyszerűen tesztre szabhatók.

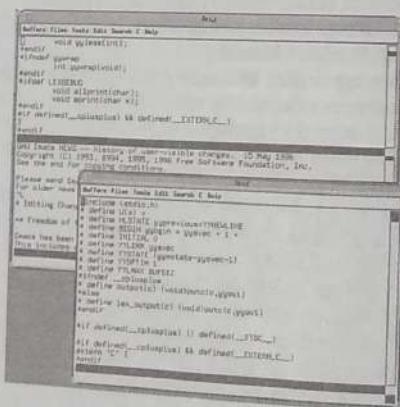
## 13.2 ALAPFOGALMAK

Az Emacs a dokumentációban és a függvények nevében olyan fogalmakat is használ, amelyek más szerkesztőkben nem léteznek, vagy jelentésük eltérő. Ezeket a későbbi félreértelesek elkerülése érdekében itt röviden ismertetjük.

A szövegen a felhasználó aktuális pozíciója (kurzor) és az előző karakter közötti hely elnevezése *pont* (*point*). Emellett van még egy kitüntetett pozíció, amelyet a felhasználó megadhat: ennek neve *jelölés* (*mark*). A *tartomány* (*region*) a pont és a jelölés közötti terület. A tartomány az egyéb szerkesztőkben egy kijelölt blokknak felel meg, és ez törlhető, kivágható vagy másolható. Elterően a többi szerkesztől, az Emacs programban a tartomány mindenkor az aktuális kurzorpozícióhoz kötött.

A billentyűzet adott konfigurálását az Emacs programban *billentyű-hozzárendelésnek* (*key binding*) nevezik, mivel ez a billentyűzet-székvencia és egy adott funkció közötti kapcsolatot definiálja. A többszörös hozzárendelések tárolása együttesen történik egy *billentyűterképben* (*keymap*). Amikor a felhasználó az Emacs programban egy fájlon dolgozik, akkor a munka egy *r pufferben* (*buffer r*) történik. Egy puffer általában a fájl tartalmát tartalmazza, de tartalmazhat eztől teljesen eltérő dolgokat is. Jó példa lehet erre az Emacs programból meghívott *gdb* GNU hibakereső bemenneti és kiimeneti tartalma, vagy egy könyvtár tartalma, amikor a felhasználó a *directory* üzemmódban van (lásd később).

Ha a puffer megjelenített, akkor egy *ablakban* látható. Az Emacs szóhasználatában azonban az ablak nem az X11 alatti ablaknak felel meg, hanem sokkal inkább az aktuális ablaknak egy olyan elkölnölt részét jelenti, amelyben egy puffer került megjelenítésre. Az X11 alatt egy különálló ablak elnevezése ebben az esetben *keret* (*frame*), és ez több ablakot tartalmazhat (lásd a 13.1. ábrán).



### 13.3 MŰKÖDTETÉS

Számos szövegszerkesztőhöz hasonlóan az Emacs indítható egy paraméterként megadott fájlnévvel is. Ha például a home könyvtárban lévő `.cshrc` fájlon szeretne dolgozni, az Emacs indítása a következő:

```
emacs ~/.cshrc
```

Az X11 alatt az Emacs az indításkor új ablakot nyit, amelyen grafikus menüsor is látható. Ha az indítás nem X11 alatt, hanem szövegmódban történik, nem jelenik meg menüsor, és a program a terminál teljes képernyőjét használja.

A felhasználó a működtetés során a kurzorvezérlő billentyűket, a funkcióbillentyűket és a különleges `<ctrl>` vagy `<meta>` billentyűkombinációkat használja. A funkcióbillentyűk konfigurálását a 13.8. szakaszban ismertetjük.

Néhány munkaállomás billentyűzetén megtalálható a `<meta>` billentyű, de a PC-ken általában használt MF2 típusú és az egyszerű ASCII billentyűzeteken nem. A `<meta>` billentyű helyett használható az `<esc>` billentyű. Például a `<meta-x>` helyett az `<esc>` billentyűt, majd az `<x>` billentyűt kell megnyomni a `<meta>` nélküli billentyűzeten. Az X11 alatt az `<alt>` billentyűt definiálják `<meta>` billentyűként, és ilyenkor az `<esc>` használata fölösleges.

A szokásos billentyűtérfelületeken a legfontosabb hozzárendelések a következők (a funkcióbillentyűk nélkül):

Kurzormozgatás

<code>ctrl-f</code>	egy karakterrel előre
<code>ctrl-b</code>	egy karakterrel vissza
<code>ctrl-n</code>	következő sorba
<code>ctrl-p</code>	előző sorba
<code>ctrl-a</code>	sor elejére
<code>ctrl-e</code>	sor végére
<code>ctrl-v</code>	egy oldallal előre
<code>meta-v</code>	egy oldallal vissza

Törlés

<code>del</code>	a cursor előtti karakter törlése
<code>ctrl-d</code>	a cursor alatti karakter törlése
<code>ctrl-k</code>	törlés a sor végéig; a törlött szöveg a vágólapon ( <i>kill ring</i> ) tárolódik, és később beilleszthető

Kijelölés, törlés és beillesztés

<code>ctrl-szököz</code>	a jelölést a pont aktuális pozíciójánál állítja be
<code>meta-w</code>	a tartományt (a jelölés és a pont közötti szöveget) a vágólapra másolja
<code>ctrl-w</code>	a tartomány tartalmát a vágólapra másolja, és azt törl a szövegből
<code>ctrl-y</code>	a vágólap tartalmát az aktuális kurzorpozícionál beilleszti

## Keresés

ctrl-s	keresés (fokozatos, lásd később)
ctrl-r	keresés visszafelé (fokozatos)
ctrl-g	a keresés vagy az utolsó parancs megszakítása

## Egyéb

ctrl-x f	fájl betöltése
ctrl-x ctrl-c	kilépés az Emacs programból
ctrl-x o	ugrás a következő ablakra
ctrl-x b	ugrás másik pufferbe

A **<ctrl-g>** gyakran használt és fontos billentyűkombináció. A kezdők gyakran úgy érzik, hogy valami elveszett az Emacs több billentyűt használó kombinációjában. Ha például a felhasználó véletlenül megnyomja a **<ctrl-x>** billentyűket, az Emacs egy újabb bevitelt vár, amely az előző billentyűkombinációt parancsra egészíti ki. Például ez a parancsbevitel szakítható meg a **<ctrl-g>** lenyomásával.

A felhasználó az Emacs billentyű-hozzárendeléscinek áttekinthető és viszonylag teljes listáját az Emacs referenciakártyán nézheti meg, amely az Emacs etc könyvtárában található. Ha például az Emacs telepítése az /usr/lib/emacs/19.28 könyvtárba történt, akkor az etc könyvtár a következő: /usr/lib/emacs/19.28/etc. A referenciakártya egy PostScript vagy TEX fájl, és neve refcard.ps vagy refcard.tex.

Emellett a felhasználó bármikor megjelenítheti az összes billentyű kiosztását a **<ctrl-h>b** (a „b” a hozzárendelésre (binding) utal) billentyűkombinációval, illetve egy adott billentyűkombináció jelentését a **<ctrl-h>k< kérdéses billentyűkombináció>** parancsal kérheti le (a „k” a billentyű (key) utal).

Az Emacs fokozatos keresés funkciója a többi szövegszerkesztőben általában nem található meg. Ez azt jelenti, hogy az Emacs nem várja meg, míg a keresésnél a felhasználó a teljes keresett szöveget beviszi, hanem az első karakter beírása után azonnal megkeresi ennek legközelebbi előfordulását, és erre helyezi a kurzort. Amikor a felhasználó beírja a következő karaktert, akkor az Emacs e két karakter következő előfordulását jelzi. Ily módon a felhasználó a keresett szöveget lépésenként közelíti meg, és általában kevesebb billentyűleütéssel elérheti.

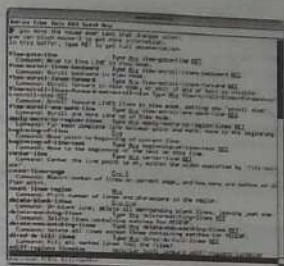
Az X11 alatt a legfontosabb függvényekhez legörökítő menü tartozik. Emellett az összes művelet közvetlenül is hivható, függetlenül attól, hogy ez adott billentyűkombinációhoz rendelt-e vagy sem. Ehhez a **<meta-x>** billentyűket kell megnyomni. Az Emacs akkor a kívánt függvény nevénél bevételét kéri. Automatikus névkiterjesztés a **<tab>** billentyűvel

Kérhető: ez általában az Emacs szerkesztőben. Ha a hivandó függvényről bővebb információra van szükség, a minipufferben keresés hajtható végre. A ritkábban használt függvényeket érdemes a **<meta->** segítségével hívni, különösen akkor, ha a felhasználó nem tudja, hogy egyáltalán van-e ehhez billentyűkombináció rendelve.

### 13.4 DOKUMENTÁCIÓ ÉS SÚGÓ

Az Emacs oktató gyors bevezető ismertetést tartalmaz az Emacs működéséről. Az oktató szöveges magyarázat, amely a **<ctrl-h>< >** billentyűkombináció hatására automatikusan betöltődik. Utasításokat tartalmaz, amelyek interaktív módon, lépésről lépésre elmagyarázzák a legfontosabb szer-

keszítési parancsokat. A tényleges súgó számos részleges függvényt tartalmaz. A felhasználó egy függvény leírását a `<ctrl-h><f>` billentyűkombinációval kérheti, míg a `<ctrl-h a>` (az „a” az apropos rövidítése) az összes függvény nevét tartalmazó listában keres egy részleges karakter-sorozatot (lásd a 13.2. ábrán). Ha például a felhasználó közvetlenül egy adott sorra ugrató függvényt keres, akkor a `<ctrl-h a>line` az összes olyan függvény listáját adja, amely nevében tartalmazza a `line` szót, valamint megadja az esetleges billentyű-hozzárendeléseket is. A lista tartalmazni fogja például a `previous-line` (előző sor) és a `next-line` (következő sor), valamint a keresett `goto-line` (ugrás adott sorra) függvényt is.



13.2 ábra. Az apropos függvény

A `<ctrl-h b>` billentyűkombinációval az aktuális **billentyű-hozzárendelések** listája jeleníthető meg, míg a `<ctrl-h m>` az aktivált üzemmódok leírását adja (lásd alább). Az összes súgó-függvény az X11 alatt a legörökítő help menütben is rendelkezésre áll.

<code>ctrl-h a</code>	apropos – az összes olyan függvény listáját adja, amelynek nevében szerepel a megadott szöveg
<code>ctrl-h c</code>	egy billentyűkombináció rövid ismertetése
<code>ctrl-h k</code>	egy billentyűkombináció bővebb ismertetése
<code>ctrl-h ctrl-k</code>	átérés info üzemmódba, és ugrás arra az oldalra, amely a billentyűkombinációhoz rendelt parancs ismertetését tartalmazza
<code>ctrl-h i</code>	átérés info üzemmódba
<code>ctrl-h m</code>	üzemmód – az aktivált üzemmódok leírása
<code>ctrl-h f</code>	egy Emacs Lisp függvény leírása
<code>ctrl-h v</code>	egy Emacs Lisp változó leírása
<code>ctrl-h h</code>	az összes <code>ctrl-h</code> parancs rövid áttekintése

#### A legfontosabb súgó-függvények

Hasonlóan a többi FSF programhoz, az Emacs szerkesztő átfogó dokumentációja is eredetileg `texinfo` formátumban áll rendelkezésre. Az Info fájlok létrehozása a telepítés során automatikusan megtörténik. A `texinfo` fájlból nemcsak Info fájlok, hanem TeX fájlok is készíthetők. Ez azt jelenti, hogy a dokumentáció kinyomtatható (lásd a 12.4. szakasz). Az Info fájlok hierarchikus felépítésűek, és kereszthivatkozásokat tartalmaznak.

A felhasználó az Emacs szerkesztőben az Info üzemmódban (lásd a 13.3. ábrán) tekintheti meg ezeket, vagy megjelenítésükhez más Info-olvasót, például `tkinfo`, is használhat.



13.3 ábra. Az Info üzemmód az Emacs szerkesztőben

Az Info üzemmóhoz külön billentyűzetkiosztás tartozik, így a felhasználó egyszerű billentyűparancsokkal vagy az egérrel követheti a kereszthivatkozásokat, és ily módon bejárhatja a hierarchikus felépítésű Info fájlt. Az X11 alatt a kereszthivatkozások kiemelése eltérő betűtípusnal és színnel történik.

Az Emacs programról az Info üzemmódban megjeleníthető adatok megegyeznek az FSF-től beszerezhető nyomtatott Emacs kézikönyv tartalmával. Az Info üzemmódban használható legfontosabb billentyűk a következők:

középső egérgomb	az egérmutató alatti kereszthivatkozás követése
d	ugrás az összes Info fájl áttekintését tartalmazó tartalomjegyzékre
Enter	a kurzor alatti kereszthivatkozás követése
l	ugrás vissza az utolsó megtekintett oldalra
n	ugrás a következő Info oldalra
p	ugrás az előző Info oldalra
u	ugrás a hierarchiában az aktuális oldal fölött lévő Info oldalra
s	szöveg keresése

A szokásos Emacs kézikönyv mellett az Emacs Lisp dokumentációja is rendelkezésre áll `texinfo` formátumban. Ez a kinyomtatva több száz oldalas kézikönyv átfogóan ismerteti a szerkesztő belső részleteit és programozását. A kézikönyv a legtöbb nagyobb FTP kiszolgálóról letölthető. A fájl általában az Emacs forráskódjával és más GNU segédprogramokkal együtt a gnu könyvtárban található.

## 13.5 ÜZEMMÓDOK

Minden pufferhez hozzárendelt egy fő-, és egy vagy több alüzemmód. A főüzemmód általában új billentyű-hozzárendelést és gyakran új függvényeket is tartalmaz, s ezzel az Emacs szerkesztő adott feladatn teszi alkalmassá. Igy külön főüzemmód tartozik az általánosan használt programozási nyelvekhez, de a konfigurációs fájlok és a normál szöveg szerkesztéséhez is. Néhány üzemmód nem fájlok szerkesztésére szolgál, hanem az Emacs szerkesztővel megvalósítható egyéb célokra. Ilyen például a `gnus`, az `rmail` és a `direc` üzemmód.

Az alüzemmódok a kijelölt főüzemmód funkciói mellett hívható egyéb szolgáltatásokat biztosítják. Példa erre a **font-lock** üzemmód, amely a programozási nyelvük kulcsszavait és a megjegyzések elérő betűtípusával és színnel jelenti meg.

A következő két táblázat néhány példát mutat be a fő- és az alüzemmódra a legfontosabb szolgáltatások rövid ismertetésével.

### Főüzemmódok:

C, C++	a forrásszöveg automatikus behúzása
lisp	a forrásszöveg automatikus behúzása, súgó, kifejezések kiértékelése
tcl	a forrásszöveg automatikus behúzása, a Tcl alatti tartományok kiértékelése

### Alüzemmódok:

font-lock	a megjegyzések, a kulcsszavakat, a karakterszorozatokat és az egyéb szövegetűleteket elérő betűtípusával és színnel jelentő meg
outline	különböző hierarchikus szinteken szöveget szűr be vagy távolít el
auto-fill	automatikus szöszétválasztás

Az Emacs a fájlok **betöltésekor** automatikusan kiválasztja a megfelelő főüzemmódot. A fő-üzemmód neve a megfelelő ablak üzemmód sorában jelenik meg. Az üzemmód kiválasztásnak szempontjából a fájlnév vagy a fájlkiterjesztés (például .c, .cc vagy .tcl) szolgál. Ezek hozzárendelése a megfelelő üzemmódokhoz egy táblázatban történik. Emellett az Emacs az üzemmódot az első sor tartalma alapján is meg tudja határozni. Ebben az esetben az egyedi megjegyzéseket veszi figyelembe, ilyenek lehetnek például a következők: #! //usr/bin/wish, # -\*-Tcl-\*- vagy # -\*-Mode: Tcl;-\*- . A felhasználó az üzemmódot maga is beállíthatja. Ehhez a <meta-x> billentyűkombináció használható, majd ezután az üzemmód nevét kell beírni, például: <meta-x>tcl-mode.

## 13.6 EGYÉB PROGRAMOK ÉS BŐVÍTMÉNYEK

Az üzemmódok mellett további általános funkciók megvalósítására Emacs Lisp programok is rendelkezésre állnak. Ezek a Lisp értelmező segítségével lényegében független alkalmazások végrehajtására alkalmasak. Ebben a szakaszban néhány ilyen bővítményt ismertünk.

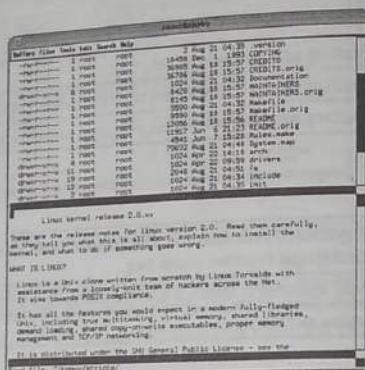
### saveplace

A **saveplace** a munkafájl bezárásakor menti a kurzor aktuális pozícióját. A fájl legközelebbi megnyitásakor az Emacs a kurzort automatikusan abba a pozícióba helyezi, amelyben a fájl előző bezárásakor volt. A felhasználó ezt a függvényt általában az összes fájdra vagy csak adott fájlokra alkalmazhatja. A megfelelő bejegyzés az Emacs konfigurációs fájlból a következő:

```
(load "saveplace")
(define-key ctl-x-map "p" 'togle-save-place)
```

**dired**

Jóllehet a **dired** tényleges üzemmód, de nem sok hasonlóságot mutat a szokásos fájlfeldolgozásra szánt főüzemmódokhoz. Amint az a nevéből is sejthető, a **dired** a könyvtárakon végrehajtható munkát teszi lehetővé. Segítségével fájlok másolhatók, átnevezhetők vagy törlhetők (lásd a 13.4. ábrán).

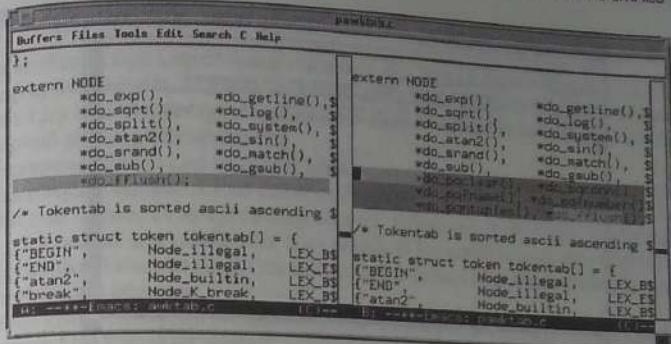


13.4 ábra. A **dired** üzemmód az Emacs programban

Az **E** billentyű lehetővé teszi az alkönyvtárak kezelését, illetve fájl betöltését egy másik pufferbe. A program automatikusan hívja a **dired** üzemmódot, amikor fájl helyett könyvtár „betöltésére” kerül sor. A felhasználó közvetlenül a <meta-x>**dired** segítségével állíthatja be.

**ediff**

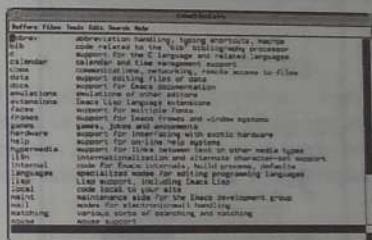
Az **ediff** szövegfájlok különböző verzióinak összehasonlításában jelent nagy segítséget. Ez az Emacs Lisp program két fájlt nyit meg, amelyeket választható módon egymás mellett vagy egymás alatt jelenít meg. A két fájl közötti különbségeket kiemeléssel (eltérő betűtípus vagy szín) jeleníti meg a program: ezek egyszerű billentyűparancsokkal hajthatók végre az egyik fájlból a másikba (lásd a 13.5. ábrán). Az **ediff** menüből választható, vagy közvetlenül a <meta-x>**ediff** segítségével hívható.



### 13.5 Az ediff az Emacs programban

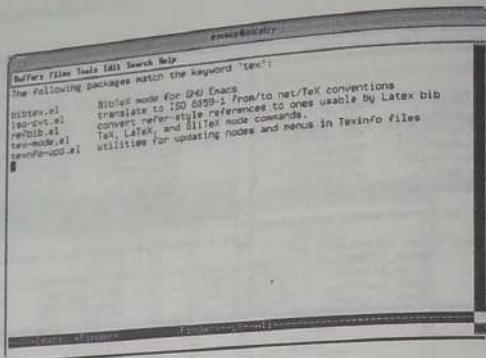
### **Sínder és lispdir**

A felhasználóknak nem könnyű eligazodni a számtalan Lisp program és Emacs témamód között. Az Emacs Finder (kereső) program ebben segít. Ez az Emacs változat Lisp programjait csoportosítja kulcsszavak szerint, és ezeket egy **választéklistában** jeleníti meg. Ha a felhasználó a <szóköz> billentyűvel kijelöl egy kulcsszót, akkor az ennek megfelelő programok jelennek meg rövid ismertetővel együtt (lásd a 13.6. ábrán). A Finder indításához először a <meta-x>load-library befrásával kell tölteni. Ezután a program a <meta-x>finder-by-keyword befrásával indítható.



13.6 ábra. A Finder

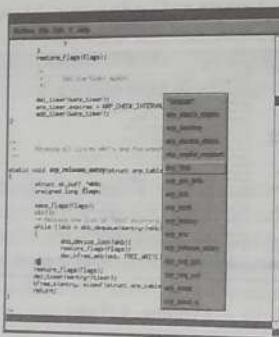
A Finder csak azokat a Lisp programokat ismeri, amelyeket az adott Emacs programcsomag már tartalmaz. Az egyéb Emacs programok áttekintése a GNU Emacs Lisp Code Directory segítségével történhet. Ez a lista 800-nál több különféle Emacs Lisp programot tartalmaz, amelyekben egy speciális programmal lehet keresgálni. A lista aktuális verziója az archive.cis.chi-state.edu kiszolgáló /pub/gnu/emacs/elisp-archive könyvtárából tölthető le. Alispdir.el.z keresőprogram ugyanebben a könyvtárban található (lásd a 13.7. ábrán).



13.7 ábra. Egy liskdir keresés eredménye

### func-menu

A func-menu program a C nyelvű programozásnál hasznos segítség. Az egérrel való kattintásra elemzi az aktív pufferben lévő C forráskódot, és a definiált függvények nevét tartalmazó feltároló menüt jelenít meg (lásd a 13.8. ábrán). Ha a felhasználó kiválaszt egy függvénynevet a menüből, az Emacs a függvény definíálási helyére ugrik a pufferben.



13.8 ábra. A func-menu egy C forráskódban

### Függvényjelzők

A függvényjelzők kezelése kevésbé kényelmes feladat, hatékonysága azonban jelentős. Az etags parancs olyan fájlt hoz létre, amelyben az összes függvény neve szerepel, valamint megtalálható az a fájl és sor, amelyben definíálása történik. A listában ezután a felhasználó Emacs függvényekkel, például tags-search és tags-apropos, keresgélhet.

## 13.7 AZ EMACS LISP

Az Emacs szerkesztő saját Lisp értelmezőt tartalmaz, és a legtöbb Emacs függvény nem C, hanem Emacs Lisp nyelven készült. Az Emacs működtetéséhez vagy a billentyűzet átdefiniálásához nem szükséges a Lisp ismerete, de függvények írásához vagy összetettebb konfiguráció kialakításához már az alapvető jártasság nélkülözhetetlen.

Természetesen az Emacs Lisp teljes körű leírása vagy részletes bevezetés a Lisp programozásba meghaladja e fejezet kereteit. A Free Software Foundation (FSF) által kiadott Emacs Lisp kézikönyv több száz oldalas, és az egyéb Lisp könyvek terjedelme is általában meghaladja az 500 oldalt. Ez a fejezet csak arra vállalkozhat, hogy példákon keresztül bemutassa a Lisp legfontosabb elemeit, így az olvasó könnyen megértheti az egyszerűbb Emacs Lisp programokat, és kisebb függvények megírására is képes lesz.

A témaiban jobban elmélyedni kívánóknak az FSF eredeti kézikönyvét, az Emacs Lisp Info fájlokat és az egyéb Lisp könyveket ajánljuk.

### Általános ismertető

A „Lisp” név a LISt Processor (listafeldolgozó) rövidítése. A rosszmájú kritikusok szerint a betűszó inkább a Lots of Irritating Single Parentheses (nagyszámú bosszantó zárójel) szavakból eredeztetető. Tény, hogy egy Lisp program legfontosabb alapeleme egy zárójelekkel tagolt lista. Ez azt jelenti a gyakorlatban, hogy a kifejezések gyakran öt vagy még több szint mélységgig egymásba ágyazott zárójelezést tartalmaznak.

Mindez azonban nem annyira elrettentő, mint első pillanatban látszik. Az elrendezés áttekinthető tethető: az Emacs Lisp üzemmódban a kifejezések automatikus behúzása, valamint a nyitó és a záró zárójelpárok kiemelése után egyáltalán nincs szükség a zárójelek számolgatására.

### Szimbólumok

A szimbólumok különleges szerepet játszanak a Lisp nyelvben. A szimbólum egyedi nevet képvisel, amelyet adatok, függvények és tulajdonságlisták elérésére használnak. (A tulajdonságlistákkal itt nem foglalkozunk részletesen.) A szimbólum kisméretű tartályként fogható fel, amelyben számos rekesz található: egy-egy rekesz a név, az érték, a függvénydefiníció és a tulajdonságlista számára.

név
érték
függvény
tulajdonságlista

A szimbólumok értékéhez bizonyos típus tartozik, amelyet nem kell előre deklarálni (szemben sok eljárással nyelvvel, pl. Pascal), hanem azt az érték implicit módon tartalmazza (mint a BASIC és a Tcl nyelvben). Az Emacs Lisp ismeri minden érték típusát. Ez azt jelenti, hogy hibás típus használata csak a futás idején derül ki.

Az Emacs Lisp nyelvben a legfontosabb primitív adattípusok a következők:

- Integer (egész)
- Floating point (lebegőpontos)
- Character (karakter)
- Array (tömb)
- Strings (karaktersorozat, egydimenziós tömb)

- Symbol (szimbólum)
- Function (függvény)
- Macro (makró)
- Primitive function (primitív függvény)
- Byte-code function (fordított függvény)



13.9 ábra. Részlet az Emacs Lisp típus-hierarchiájából

A fenti adattípusok mellett, amelyek a többi Lisp nyelvjárásban is megtalálhatók, az Emacs Lisp tartalmaz szerkesztési funkciókra használt egyedi adattípusokat is. Ezek közül a legfontosabbak a következők:

- buffer (puffer)
- window (ablak)
- frame (keret)
- process (processz)
- stream (adatfolyam)
- keymap (billentyűtérféjl)

Az Emacs Lisp nyelvben az adattípusok részlegesen átfedik egymást, egy érték egyidejűleg több típusba tartozhat. Ez azért van így, mert néhány típus egy másik típus **speciális változata**. A string adattípus például az array típus speciális változata, amely viszont a sequence különleges esete.

Az összehasonlítások vagy logikai kifejezések eredményeként adódó true (*igaz*) és false (*hamis*) érték kifejezésére a Lisp nyelvben a t és a nil szimbólum szolgál. A nil az () üres lista szinonimája, így az atomic (atomi) elem és lista is.

### Az értelmező

A Lisp értelmező olvasás–értelmezés–eredménykiadás ciklust hajt végre. Ez azt jelenti, hogy bemeneti adatot olvas be, megkíséri az értelmezni azt, majd kiadja az eredményt. Az értelmező által kiértékelhető kifejezések elnevezése *form* (forma) vagy *s-expression* (s-kifejezés). Az utóbbi az atomi és a listaelemek (számok, karaktersorozatok és szimbólumok) összefoglaló elnevezése.

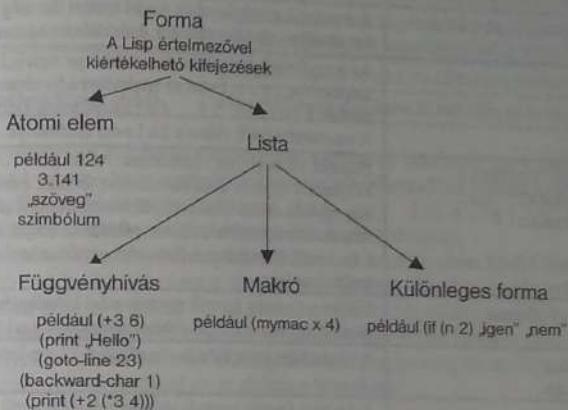
Ha az értelmező értéket talál, ilyen például egy karaktersorozat vagy egy szám, akkor a kiértékelés eredménye maga az érték lesz. Ha az értelmező szimbólumot talál, az eredmény a rekesz tartalma lesz a szimbólum értékével együtt. A következő példa több bemeneti értéket és a Lisp értelmező ezekhez tartozó eredményét mutatja be:

```

1
1
"test"
  
```

```
test
fill-column
78
```

Ha az értelmező listát talál, akkor a kiértékelés a lista első szimbólumától függ. A szimbólum kiértékelése nem történik meg, ez az s-kifejezés fajtáját határozza meg. Ha ez egy függvény neve, akkor a lista egy függvényhívás. Az értelmező ezután megkíséri kiértékelni a maradék elemeket (az első kivételével a többöt), és a függvényt a kiértékelés eredményével hívja (lásd a 13.10. ábrát).



13.10 ábra. Emacs Lisp kifejezések

A függvényhívásként értelmezett listák mellett makrók és különleges formák is léteznek. Ezeknél a maradék listaelemek kiértékelése nem történik meg. Különleges formák például a feltételek (*if* vagy *cond*).

A függvényhívások általános alakja a következő:

```
(functionName Argument Argument ... )
```

Ez a matematikából ismert prefixkódos ábrázolás. A következő példa függvényhívásokat és az értelmezőtől kapott eredményeket mutat be:

```
(+ 1 4)
5
substring "abcdefg" 2 3
"C"
*(+ 2 3) 3
15
```

A függvényhívások egyes argumentumai lehetnek szimbólumok vagy listák is, amelyek szintén függvényhívásokat kerülnek kiértékelésre. Az ilyen lista kiértékelése tehát rekurzív.

A Lisp függvények részletesebb ismertetése előtt érdemes megismerkedni néhány különleges formával. Ezek, többek között, feltételek és ciklusok megvalósításában és függvények definíálásában lehetnek hasznosak.

### Különleges formák

(defun fname (P1...Pn) mej Form1 ... Formn)	A P1 ... Pn paraméterekkel és a Form1 ... Formn függvénydefinícióval rendelkező fname nevű függvényt definiálja.
(defvar symbol érték mej)	A symbol szimbólumhoz rendel értéket (ha még nincs értéke). Ezt általában globális változók definíálására használják.
(if exp true falsel ... falsen)	Az exp formát értékeli ki. Ha eredménye nem nil, a forma kiértékelése true, és ezt az eredményt adja vissza. Ellenkező esetben a falsel ... falsen formák kiértékelése történik. A visszatérési érték ekkor a falsen eredménye.
(cond (exp1 forms1) (exp2 forms2) ... (expn formsn))	Hasonlít a Pascal case utasításához. Az exp1 ... expn kifejezések egymás utáni kiértékelése történik mindenkorábban, míg egyikük értéke nem nil. Ebben az esetben az ehhez tartozó formák kiértékelésére kerül sor, például az exp2 esetében a forms2 formákra. A visszatérési érték az utolsó kapcsolt forma eredménye lesz.
(while cond forms)	Ciklus: a forms formák egymás utáni kiértékelése történik mindenkorábban, míg a cond feltétel eredménye a nil értéktől eltérő.
(quote list) vagy 'list	A visszatérési érték kiértékelés nélkül a list lista.
(progn forms)	A forms formák egymás utáni kiértékelése kerül végrehajtára, és az utolsó eredmény lesz a különleges forma eredménye.
(setq sname nvalue)	Az új nvalue értéket rendeli az sname szimbólum értékrekeszéhez.
(let (Var1 ... Varn) forms)	A formákon belül helyi változóként használható Var1 ... Varn változóhoz új csatolásokat hoz létre. Var1 ... Varn lehetnek szimbólumok, és ekkor ezek a nil értéket kapják, vagy lehetnek Var form típusú listák, és ekkor a Var szimbólum csatolódik a form eredményéhez.
(save-excursion forms)	A forms formák egymás utáni kiértékelése történik (mint a progn esetében), de a szerkesztő aktuális állapota (azaz az aktuális puffer, a kurzorpozíció és a jelölő pozíciója) is mentésre és visszatöltésre kerül forms kiértékelés után.

A további különleges formák ismertetésére akkor kerül sor, amikor azokat példában használjuk. Az Emacs Lisp összes különleges formáját az Emacs Lisp Info dokumentuma tartalmazza.

### Példák a különleges formák használatára

```
(defvar TestVariable 123
  "This variable is only for test purposes.")
(setq TestVariable (+ TestVariable 2))
```

Ebben a példában a `TestVariable` szimbólum globális változóként használatos. Először a változóhoz a 123 érték kerül hozzárendelésre, majd a `setq` forma 2-vel növeli az értéket. A forma eredménye a `TestVariable` új értéke lesz, azaz 125.

Az itt közölt egyszerű példákat egyszerű **kipróbálni** az Emacs szerkesztőben. Indítsa el a programot fájlnév megadása nélkül. Az Emacs a szokásos verziószámot és a súgó parancsokat jeleníti meg. Tetszőleges gomb megnyomására eltűnik, és a `*scratch*` nevű puffer rendelkezésre áll. Az aktuális üzemmód a „Lisp interaction” lesz, amely azt jelenti, hogy a Lisp értelmező kész a munkára. Az üzemmód a következő billentyű-hozzárendeléseket tartalmazza:

<code>tab</code>	az aktuális sor beágyazottságától függő megfelelő behúzása.
<code>meta-tab</code>	az aktuális szimbólum kiegészítése. A függvénynevek, változók és a különleges formák automatikus kiterjesztése megtörténik.
<code>meta-ctrl-x</code>	a kurzortól balra lévő defun különleges forma kiértékelése.
<code>lfd</code> (soremelés billentyű vagy <code>ctrl-j</code> )	a kurzortól balra lévő forma kiértékelése és az eredmény kiadása.

A PC-s billentyűzeteken sajnos általában nincs soremelés billentyű. Ez a funkció azonban a `<meta-x>local-set-key` segítségével valamelyik billentyűhöz hozzárendelhető. Az Emacs ekkor a billentyűkombináció megadásához a „Set key locally: -“ üzenetet jeleníti meg. Ekkor például beadható a `<meta-return>` billentyűkombináció.

Az Emacs ezután a megadott billentyűkombinációhoz hozzárendelni kívánt függvényt kérdezi. Itt az `eval-print-last-sexp` megadása szükséges. Ezzel a `<meta-return>` billentyűkombinációhoz az `eval-print-last-sexp` Emacs függvényt rendeltük, amely a kurzortól balra lévő formát értékeli ki, és kiadja az eredményt. Egy forma bevitelle után tehát egyszerűen a `<return>` helyett a `<meta-return>` billentyűket kel megnyomni, és az Emacs Lisp értelmező kiértékeli a formát. A fenti példánál ez az eljárás a következő:

```
(defvar TestVariable 123
  "This variable is only for test purposes.")
TestVariable
(setq TestVariable (+ TestVariable 2))
125
```

A következő példában egy egyszerű függvényt definiálunk, amely két szám összegét 2-vel szorozza. Az új függvényt ezután a `print` függvényben hívjuk.

```
defun doublesum (a b)
  "simple test function"
  (* 2 (+ a b))
doublesum
(print (doublesum 2 3 ))
10
10
```

A Lisp az eredményt **kétszer adja ki**: először a `print` függvény hatására, majd annak a formának az eredményeként, amelyben a `print` függvény hívása történt.

Mindkét előző példában megjegyzésként karaktersorozat szerepelt. Ez a megjegyzés (szokásos elnevezése még doc string) a definícióval együtt mentésre kerül. Ezek a megjegyzések később a describe-function és a describe-variable előre definiált függvények olvashatók el (ezek a <ctrl-h><ctrl-f>, illetve a <ctrl-h><ctrl-v> billentyűkombinációval hívhatók). Ez azt jelenti, hogy ugyanazok a segítségi függvények hívhatók az egyéni függvényekhez is, mint az előre definiáltakhoz.

A programozás során az egymásra következő lépésekben átmeneti eredmények tárolására is szükség lehet: ez helyi változókkal oldható meg. A Lisp nyelvben erre a let különleges forma szolgál, amint ez a következő példában is látható:

```
(defvar square 0
  "global value of square"
square
(defun f (n)
  "example function using let"
  (let ((square (* n n)))
    (print square)
    (print (- square 2))))
f
(f 3)

9
7
7

square
0
```

A let forma egy új hozzárendelést hoz létre a square szimbólum számára. Ez a szimbólum globális értékét jelenti. Az új hozzárendelés csak helyileg, a let formán belül érvényes. A let formát elhagyva a square globális értéke – ami nem változott – lesz érvényes újra.

A Common Lisp hozzárendelési módszerével szemben az Emacs Lisp dinamikus hozzárendelést használ. Ez azt jelenti, hogy az új let változók értékei azokból a függvényekből érhetők el, amelyeket a let formán belül hívnak. A következő példa ezt világítja meg:

```
(defvar a 100)
a
(defvar b 200)
b
(defvar c 300)
c
a
100
(defun fact1 ()
  "Test function using let"
  (print "in fact1"
```

```

(print (format "a=%s, b=%s" a b c))
(let ((a 1) (b (+ 2 3)) c)
  (print "in fct1 in let")
  (print (format "a=%s, b=%s, c=%s" a b c)))
  (fct2))
  (print "in fct1 again outside of let")
  (print (format "a=%s, b=%s, c=%s" a b c)))
fct1
(defun fct2 ()
  "Test function, invoked by fct1 and accessing variables"
  from fct1"
  (print "in fct2")
  (print (format "a=%s, b=%s, c=%s" a b c)))
fct2

(fct1)

"in fct1"

"a=100, b=200, c=300"

"in fct1 in let"

"a=1, b=5, c=nil"

"in fct2"

"a=1, b=5, c=nil"

"in fct1 again outside of let"

"a=100, b=200, c=300"

"a=100, b=200, c=300"

```

## Tipusellenőrző függvények

Az előre definiált függvények az Emacs Lisp jelentős elemei. Nagy számauk miatt itt csak egy kis részüket mutathatjuk be. A rendelkezésre álló függvényekről azonban jó áttekintést kapható az Emacs súgó függvényei vagy az Emacs Lisp Info fájljai segítségével.

Az a tény, hogy a Lisp nyelvben egy szimbólum vagy egy függvény visszatérési értékének típusa gyakran csak a futás idején határozható meg, szükségesse teszi a **tipusellenőrző** függvényeket. A következő táblázat ilyen függvényekre mutat be példákat. Ezek azt vizsgálják, hogy átadott kifejezés adott típusú-e, és eredményül a t vagy a nil értéket adják vissza. Nevük általában a típus nevéről származik kiegészítve a p betűvel (predikátum). A Lisp gyakran használja ezt a kiterjesztést vizsgálatot végrehajtó, és eredményül a t vagy a nil értéket adó függvényeknél.

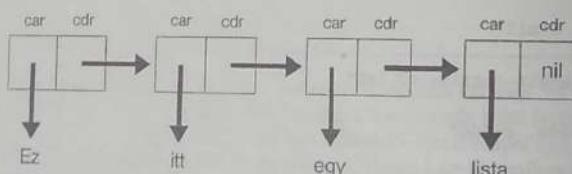
arrayp	ellenőrzi, hogy az átadott érték tömb-e
bufferp	ellenőrzi, hogy egy puffer értéke beállított-e
consp	ellenőrzi, hogy az érték cons cella-e (a Lisp nyelvben ez a listák építőeleme)
listp	ellenőrzi, hogy az érték lista-e
numberp	ellenőrzi, hogy az érték numerikus-e

Példa:

```
(setq a "this is a string"
      "this is a string"
      (arrayp a)
      t
      (stringp a)
      t
      (numberp a)
      nil)
```

### Listakezelő függvények

A Lisp számos, listák feldolgozására szolgáló függvényt tartalmaz. E függvények megértéséhez fontos ismernünk a listák ábrázolását a Lisp nyelvben. A listák *cons* cellák összekapcsolásával képződnek. Ezek két mutatót tartalmazó cellák. Az első mutató – elnevezése *car* – értékre hivatkozik, míg a második mutató – elnevezése *cdr* – a következő listaelem cons cellájára hivatkozik. Egy lista utolsó cons cellájának *cdr* értéke *nil*. A 13.11. ábrán lévő példa egy lista (Ez itt egy lista) cons cellákkal való ábrázolását mutatja be.



13.11 ábra. Egy lista belső felépítése

Ha egy listaelém *car* mutatója egy további cons cellára mutat, egymásba ágyazott listát kapunk. A következő táblázat a legfontosabb listakezelő függvényeket sorolja fel:

car	egy lista első elemét adja vissza (az első cons cella car mutatója)
cdr	a lista első eleme után következő elemeket adja vissza (az első cons cella cdr mutatója)
cons	új cons cellát hoz létre
nth	egy lista egy elemét adja vissza
list	új listát hoz létre
append	hozzáfűzéssel két listát egyesít
reverse	listát invertál
sort	listát sorba rendez

A szokásos Emacs Lisp függvények mellett egy speciális kiterjesztés betöltése után a Common Lisp függvényei is használhatók. A listafeldolgozást segítő két makró: a `push` a lista elejére felvesz egy elemet, míg a `pop` a lista elejéről eltávolít egy elemet.

Példa:

```
(setq l1 '(This is a list)
      (This is a list)
      (car l1)
      The
      (cdr l1)
      (is a list)
      (cons 'This (cdr l1))
      (This is a list)
      (append '(and this too) (cdr l1))
      (and this too is a list)
      (nth 2 l1)
      a)
```

### Karaktersorok kezelő függvények

A karaktersorozat-típus a Lisp nyelvben a tömbtípus különleges esete. Ez azt jelenti, hogy az összes tömbkezelő függvény karaktersorozatokra is használható. A következő táblázat a legfontosabb ilyen függvényeket sorolja fel.

<code>make-string</code>	karaktersorozatot hoz létre
<code>substring</code>	rész-karaktersorozatot ad vissza
<code>concat</code>	karaktersorozatokat fűz össze
<code>string=</code>	két karaktersorozatot hasonlít össze
<code>format</code>	karaktersorozatot formáz (hasonlóan, mint a <code>sprintf</code> a C nyelvben)

Példa:

```
(setq a "this is a string")
      "this is a string"
      (stringp a)
      t
      (substring a 4 10)
      " is a "
      (make-string 10 32)
      "          "
```

### Kurzormozgatás és szövegfeldolgozás

Az Emacs Lisp legfontosabb függvényei természetesen azok, amelyek egy pufferben lévő szöveg feldolgozását hajtják végre, a kurzort mozgatják, szöveget szúrnak be vagy törölnek. Ezek

Ichetővé teszik a szerkesztési feladatok automatizálását. A következő táblázat ilyen függvényeket mutat be.

point	a kurzor aktuális pozícióját adjá vissza
goto-char	a kurzort adott pozícióba helyezi
forward-char	a kurzort egy helyel jobbra viszi
forward-word	a kurzort egy szóval jobbra viszi
search-forward	adott karaktersorozatot keres a pufferben
insert	a pufferben a kurzor aktuális pozíciójánál karaktersorozatot szűr be
delete-region	a pufferben törli az aktuális tartományt

### Interaktív függvények

Egy függvényt a felhasználók akkor hívhatnak a hozzárendelt billentyűkombinációval vagy a `<meta-x>` segítségével, ha a függvény tartalmazza az `interactive` függvényhívást. Ez a hívás hasonlít egy azt meghatározó deklarációhoz, hogy mely argumentumokat kell átadni a függvénynek a hívásakor. A legegyszerűbb eset az argumentumok nélküli függvényeké, mert ekkor elegendő a függvény elején az `interactive` hívása. A következő példa egy ilyen egyszerű függvényt mutat be, amelyet a `<shift-jobbra>` billentyűkombinációhoz rendelhetünk, amely a kurzort egy karakterrel jobbra viszi, és kijelöli a szöveget.

```
(defun mark-move-right ()
  "move right and set the mark if it is not already active"
  (interactive)
  (if (not mark-active)
      (push-mark nil nil t))
  (forward-char))
```

Lehetséges olyan függvény definiálása is, amely paramétereit a felhasználói minipufferen keresztül kapja: ez szintén az `interactive` függénnel valósítható meg. Paraméterként egy kódkaraktér és egy azt követő promptot tartalmazó karaktersorozat kerül átadásra az `interactive` függvénynek. A következő példa egy interaktív függvényt definiál, amely egy „-” karakterekből álló sort szűr be a szövegbe (ehhez előbb a sor hosszát kell megállapítani).

```
(defun line (len)
  "draw a -- line with length len"
  (interactive "nLength")
  (insert (make-string len 45)))
```

Az itt használt kódkarakter az „n”, amely numerikus bevitelt jelent. Ha több paraméter kerül átadásra, ez ugyanabban a karaktersorozatban történik, de határolóként a „\n” használatos. A következő példa az előző továbbfejlesztése: ebben lehetőség van a sorkötöltő karakter megadására is. Egyedi karakter bevitelének kódkaraktere a „c”.

```
(defun line (len lchar)
  "draw a -- line with length len"
  (interactive "nLength ")
  (insert (make-string len lchar)))
```

Az összes kódkarakter részletes ismertetése a súgóban (**<ctrl-h >interactive**) vagy az Emacs Lisp Info fájlokban található.

### A programozási stílus

A Lisp funkcionális nyelv, amely azt jelenti, hogy a legfontosabb eleme a függvény. A Pascal nyelvben használt tiszta eljárások, vagy az értéket vissza nem adó szubrutinok nem léteznek a Lisp nyelvben. A funkcionális programozás azt jelenti, hogy ahol csak lehet, kerülni kell a változók hozzárendelését, hogy így közvetlenül a függvények visszatérési értékeivel vagy a kifejezések értékével lehessen dolgozni.

## 13.8 KONFIGURÁLÁS

Az Emacs programban a billentyűzet tesztreszabása vagy további függvények és programok betöltsése általában a felhasználó home könyvtárában található .emacs fájl segítségével történik. Az indításkor az Emacs betölti ezt a fájlt, majd az Emacs Lisp értelmező kiértékeli. Így ezt  *inicializációs fájlnak* is nevezik.

Ezek a beállítások globálisan az összes felhasználó számára a site-start.el és a default.el fájlból is végrehajthatók. Ezeknek a fájloknak abban a könyvtárban kell lenniük, amelyben az Emacs automatikusan a Lisp fájlokat keresi. Ilyen könyvtár a site-lisp, amelynek elérési útvonala a Linux alatt általában a következő: /usr/lib/emacs/site-lisp. Más UNIX rendszerek az Emacs szerkesztő általában az /usr/local könyvtárba telepítik. Ekkor a fájl elérési útvonala a következő: /usr/local/lib/emacs/site-lisp.

Ha a site-start.el fájl létezik, akkor egy felhasználói inicializációs fájl értékeli ki. A default.el fájl szintén nem kötelezően használt, és az alapértelmezett inicializációs adatokat tartalmazza. Betöltésére a felhasználói inicializációs fájl után kerül sor, ha ebben az inhibit-default-init változó a nil értéktől eltérőt tartalmaz. Így tehát három inicializációs fájl létezhet, egyikük használata sem kötelező, és hivásukra a következő sorrendben kerül sor:

- site-start.el – a felhasználó inicializációs fájltól függetlenül végrehajtott globális beállításai
- ~/.emacs – felhasználói beállítások
- default.el – globális beállítások, amelyek alkalmazására akkor kerül sor, ha a felhasználó nem tiltotta ezeket az inhibit-default-init változóban.

A konfigurálás ezekben a fájlokban történik az Emacs Lisp segítségével. A fájlokban lévő bejegyzések Emacs Lisp függvényhívások. Amint ezt a 13.7. szakaszban látuk, ezek zárójelű kifejezések, amelyekben az első listaelem jelenti a függvénynevet, míg a többi elem a paraméterek. Ha a paraméterek szimbólumok, kiértékelést kerülnek, ha előttük nem az aposztróf (') karakter áll.

A konfigurációs fájlokban néhány elérési útvonalban az Emacs verziószáma is szerepel. Ez a szám a verziónak megfelelően változik. Ebben a szakaszban a 19.28 verziót ismertetjük (1996 augusztusában a 19.33-as verzió volt a legújabb ismert Emacs verzió).

A billentyűzetkiosztás mellett a konfigurációs fájlok globális konfigurációk változókat is beállítanak, valamint további programokat is aktivizálnak. Egy összetettebb Emacs konfigurációs fájl a következő szakaszokat tartalmazza:

- Globális változók beállítása adott funkciók beállítására vagy módosítására.
- A betűtípusok és a színek beállítása az X11 alatt futó Emacs számára.
- További függvények és programok betöltése.
- Olyan parancsok definiálása, amelyek hívása automatikusan betölti a megfelelő programsomagot vagy Lisp fájlt.
- Üzemmód vagy program indításakor végrehajtandó parancsok definiálása.
- Billentyűzetkiosztás megadása.

A fenti szakaszokat példákkal mutatjuk be.

### Globális változók és beállítások

Egy .emacs fájl következő részlete először a Common Lisp nyelvi kiterjesztést (c1) tölti be, amely egyszerűbbé teszi a konfigurálást. Emellett egy kis Lisp makró definíálására kerül sor a hibák figyelmen kívül hagyására szolgáló condition-case különleges forma rövidítéseként. Ez a makróról itt nem ismertetjük. A Lisp makrókról részletes leírás található a Lisp könyvekben és az Emacs Lisp Info fájlokban.

```
;: Load the Common Lisp language extension
(require 'c1)

;; macro to abbreviate condition-case
;; if all errors are to be ignored
(defmacro ignore-errors (&rest forms)
  "short from for a condition case"
  (list 'condition-case 'nil
        (cons 'progn forms)
        '(error nil)))

;; Path specification for Lisp programs
(push "/usr/lib/emacs/site-lisp/auctex" load-path)
(push "/usr/lib/emacs/site-lisp/swi" load-path)

;; Default mode for unknown file types
(setq default-major-mode 'text-mode)

;; Text width
(setq default-fill-column 78)

;; Make region visible
(setq transient-mark-mode t)

;; Permit eval with <Meta-Esc>
(put 'eval-expression 'disabled nil)
```

A `load-path` változó karaktersorozatként útvonal-definíciók listáját tartalmazza. Egy Emacs Lisp fájl betöltésekor ezekben az elérési útvonalakban megadott helyeken kersegél a rendszer. A szokásos Linux telepítésnél, ahol nem került sor az Emacs módosítására, indításkor ez a lista elsőként a hivatalos Lisp fájlok `/usr/lib/emacs/19.28/lisp` elérési útvonalát és az Emacs `site-lisp` könyvtár `/usr/lib/emacs/site-lisp` elérési útvonalát tartalmazza.

Ha kiegészítő Lisp programcsomagok is – ilyen például az AUCTEX csomag – telepítésre kerülnek más könyvtárakba, akkor a változót bővíteni kell. Mivel az adatstruktúra lista, a push Comon Lisp függvény használható egy elem hozzáfűzésére a lista elején.

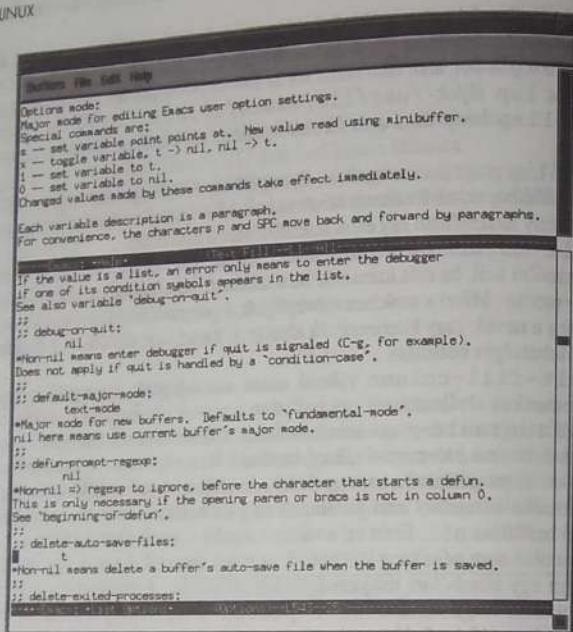
A `default-major-mode` változó annak a főüzemmódnak a nevét tartalmazza, amelyet egy fájlnál használni kell, ha más üzemmód nem található. Az alapértelmezett üzemmód a `fundamental-mode`. Mivel a szokásos szövegfájlok, a programfájlokkal ellentétben, nem különböztethetők meg a nevük vagy kiterjesztésük alapján, a `text-mode` (szöveg-üzemmód) lehet a változó másik lehetséges beállítása.

A `default-fill-column` változó annak az oszlopnak a megadását teszi lehetővé, amelyben automatikus elválasztásnál egy pufferben a szavak elválasztása kezdődhet. Egy másik hasonló változó a `default-tab-width`, amely a tabulátorközök beállítását teszi lehetővé.

A `transient-mark-mode` változó kapcsoló. Ha a változó a `nil` értéktől eltérő értéket tartalmaz, akkor ez megszünteti a szöveg aktuális kijelölését a puffer módosításakor. Ily módon azonban az aktuális tartomány nem jeleníthető meg kiemeléssel, ha a `transient-mark-mode` változó beállítása `nil`. Ezért ezt a változót minden t értékre kell beállítani.

A szakasz utolsó sora némi képp különbözik az előző soroktól. Ez nem egy általános változót módosít, hanem egy szimbólum tulajdonságlistáját olvassa be. A `<meta-esc>` vagy az `<esc-esc>` billentyűkombináció valójában az `eval-expression` függvényt jelenti, amely egy Lisp kifejezés kiértékelését teszi lehetővé közvetlenül a minipufferben. A kezdőket és a korábban a vi szerkesztő használókat, akit egy téves billentyűkiadás lezárásához gyakran használtak az `<esc>` billentyűt, kissé megzavarhatja ez a függvény. Éppen ezért ez a szolgáltatás eredetileg ki van iktatva, és a `<meta-esc>` megfelelő üzenetet jelenít meg. Azokat a felhasználókat viszont, akit már jobban ismerik az Emacs szerkesztőt, és tudják, hogy az Escape billentyű nem az `<esc>`, hanem a `<ctrl-g>`, éppen a megjelenő üzenet fogja zavarni. A `<meta-esc>` normál működése a disabled tulajdonság `nil` értékre állításával állítható vissza. Egy változó leírását legegyszerűbben a `<ctrl-h v>` segítségével kaphatjuk meg. Ennek kiadása után a minipuffer a változó nevének megadását kéri, és az Emacs ennek leírását egy új ablakban jeleníti meg.

Az `edit-options` függvény jó áttekintést nyújt az Emacs szerkesztőben használt összes globális változóról. Ekkor egy új ablakban az összes változó, valamint aktuális értékük és leírásuk jelenik meg. Az értékek ebben az ablakban közvetlenül megváltozthatók (lásd a 13.12. ábrán).



13.12 ábra. Az edit-options függvény

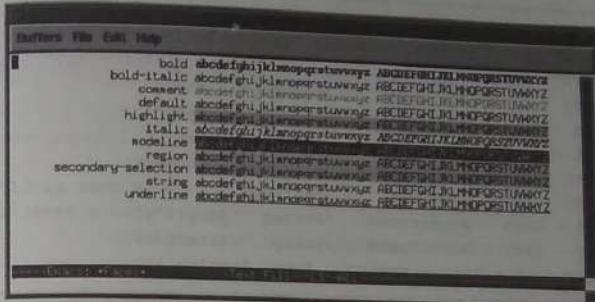
### Betűtípusok és színek

A 19-es verziósámu program kibocsátása óta a GNU Emacs az X11 alatt saját, legördülő menütet tartalmazó ablakot nyit, az egérrrel kezelhető, és egy pufferben különböző betűtípusokat és színeket jeleníthet meg. Ez a szakasz a betűtípusok és a színek konfigurálását mutatja be, az egérmóveletekről és a legördülő menüköről később, a billentyűzet elrendezésével kapcsolatban lesz szó.

A GNU Emacs a különböző betűkészleteket és színeket *faces* (*betűtíjak*) segítségével kezeli. A betűtíjak a következő jellemzőket tartalmazzák: betűtípus, előtérszin, háttérszin és aláhúzott-e vagy sem. Az ily módon definiált betűtíjak ezután a szövegterületeken megfelelően alkalmazhatók.

Ha a *transient-mark-mode* aktív, az Emacs a *region* betűtíjak alkalmazásával jelenti meg az aktuális tartományt. A különböző főüzemmódok szintén betűtíkokat használnak a szöveg megjelenítésére. Az Info üzemmódban például az egyéb oldalakra vagy lehetőségekre való hivatkozások szintén egyedi betűtíakkal történik. Programfájlok szerkesztésekor a front-lock alüzemmód választói ki, amelyben a kulcsszavak, a függvénynevek, a megjegyzések és a karakterszorozatok különböző betűtíkokkal jeleníthetők meg.

Az Emacs Lisp nyelvben a betűtíjak létrehozására általában a *make-face* vagy a *copy-face* függvényt, míg módosításukra a *set-face-font*, a *set-face-foreground*, a *set-face-background* és a *set-face-underline-p* függvényt használják. A *list-faces-display* függvény segítségével az összes definiált betűtíjak megtékinthető (lásd a 13.13. ábrán).



13.13 ábra. A list-faces-display függvény eredménye

A következő példában több betűalak módosítására kerül sor.

```
;;
;; fonts and faces
;; = = = = =
;; Set fonts in Emacs
(when window - system
  (ignore - errors
    (set - face - font 'bold
      " - adobe - courier - bold - r - normal - * ")
    (set - face - font 'italic
      " - adobe - courier - medium - o - normal - * ")
    (set - face - font 'comment
      " - adobe - courier - medium - o - normal - * ")
    (set - face - font "bold - italic
      " - adobe - courier - bold - o - normal - * ")))
;; Set faces for font - lock mode
(when window - system
  (setq font - lock - function - name - face 'bold)
  (setq font - lock - comment - face           'italic)
  (setq font - lock - string - face           'default)
  (setq font - lock - doc - string - face     'default))
;; Set attribute for faces
(when window - system
  (ignore - errors
    ;; Region
    (set - face - foreground 'region "black")
    (set - face - background 'region "grey90")
    (if (not (x - display - color - p))
        (set - face - underline - p 'region t))
    ;; bold face
    (set - face - foreground   'bold   "black")
    (set - face - background   'bold   "white")
    (set - face - underline - p 'bold   nil)))
```

```
;: Additional faces make sense for color monitor
(when (and window - system (x - display - color - p))
  (ignore - errors
    (copy - face 'default 'comment)
    (set - face - foreground 'comment "grey60")
    (set - face - background 'comment "white")
    (setq font - lock - comment - face - "comment)
    (copy - face 'default 'string)
    (set - face - foreground 'string "grey10")
    (set - face - background 'string "white")
    (setq font - lock - string - face 'string)))
```

### Bővítmények betöltése és aktivizálása

A bővítmények betöltésére a load Lisp függvény szolgál. Ez a függvény hibaüzenetet ad, ha a betölteni kívánt fájl nem létezik, éppen ezért az ignore-errors makrót használjuk, amely hiba esetében folytatja a feldolgozást. A func-menu csomagban az X11 alatt és a C üzemmódban az aktuális fájlban definiált összes függvényt tartalmazó menü jeleníthető meg. Ha ebben a menüben kijelölünk egy függvényt, a kurzor a függvény definíciójára kerül. Ez így egy egyszerű de hatékony forráskód-tallózót valósít meg.

```
;: func - menu
(when window - system
  (ignore - errors
    (load "func - menu")
    (define - key global - map [S - down - mouse - 3] 'function - menu)))
```

Az egyes üzemmódokat érintő beállítások módosításához **kapcsokat** (hook) használunk. A kapcsok az egyes üzemmódokban rendelkezésre álló változók, amelyekben Lisp kifejezések tárolhatók. Ezek a kifejezések – a kapocs típusától függően – adott időpontban végrehajtásra kerülnek; általában ez az üzemmód inicializálása egy új pufferhez. Ez azt jelenti, hogy bizonyos függvények egy puffer adott üzemmódban való megnyitásakor minden alkalommal végrehajthatók.

Egy Emacs inicializációs fájl következő részlete ezt a mechanizmust használja a line-numbers és a font-lock alüzemmód aktivizálására a C és az Emacs Lisp üzemmódban lévő pufferek számára.

```
;: line - numbers and font - lock for c
(add - hook 'c - mode - hook
  '(lambda ()
    (line - number - mode 1)
    (if window - system
        (font - lock - mode t)))
;: line - numbers and font - lock for emacs - lisp
(add - hook 'emacs - lisp - mode - hook
  '(lambda ()
    (line - number - mode 1)
    (if window - system
```

```
(font - lock - mode 1)))))

;; line - numbers and font - lock for tcl
(add-hook 'tcl-mode-hook
  '(lambda ()
    (line-number-mode 1)
    (if window-system
        (font-lock-mode t)))))

;; Auto fill in text mode
(add-hook 'text-mode-hook
  '(lambda () (turn-on-auto-fill))))
```

Az Emacs számos olyan üzemmódot is tartalmaz, amely konfigurálás nélkül nem kerül aktiv állapotba. Ezeket használatukhoz be kell tölteni az Emacs autoload mechanizmusával, amely lehetővé teszi speciális Lisp fájlok betöltését csak akkor, amikor ténylegesen szükség van rájuk. Az autoload függvény lehetővé teszi ilyen specifikációk megadását. Paraméterként az autoload egy függvénynevet és egy fájlnévét kap. Amikor a függvény-paraméter hívására kerül sor, az Emacs tudni fogja, hogy a megadott fájlt előbb be kell tölteni.

Adott kiterjesztésű fájl szerkesztésekor automatikusan új üzemmód aktivizáláshoz megfelelő bejegyzést kell írni az auto-mode-alist listába, amely párosaval előforduló bejegyzésekben egy mintát és egy függvényt tartalmaz. Ezek azt határozzák meg, hogy a mintával egyező fájl betöltenekor az adott függvény hívására kerül sor. A következő példa az Ada fájlokra végzi el ezt a beállítást. Ekkor az .ada végű fájlok betöltésekor az ada-mode függvény hívása történik. Ez a függvény az ada.el fájl betöltésével kezdődik.

```
;; Ada mode
(autoload "ada-mode" "ada"
  "Ada major mode." t)
(pushnew '("\\.ada$" . ada-mode)
  auto-mode-alist)

;; Smalltalk mode
(autoload 'smalltalk-mode "st.el" "" t)
(pushnew '("\\.st$" . smalltalk-mode)
  auto-mode-alist)

;; Modula-3 mode
(autoload 'modula-3-mode "modula3.el" "" t)
(pushnew '("\\.m3$" . modula-3-mode)
  auto-mode-alist)
(pushnew '("\\.I3$" . modula-3-mode)
  auto-mode-alist)

;; liskdir -- Search for/retrieve additional
;; packages in the Emacs Lisp Archive dir
(autoload 'lisp-dir-apropos "lispdir" nil t)
```

A pushnew függvény olyan Common Lisp makró, amely egy elemet fűz egy lista elejére, ha az elem még nem szerepel a listában. Az autoload mechanizmus a fenti részben is szerepel a lisp-dir-apropos függvény betöltésére, amely a Lisp kódtárból hajt végre keresést (lásd a 13.6. szakaszban).

A következőkben három kisebb Lisp függvényt definiálunk, amelyekhez később majd billentyűvel is rendelünk. A mark-and-do függvényt a shift billentyűvel együtt használjuk. A <shift-jobbra>

billentyűkombináció a kurzort egy karakterrel jobbra viszi, és kiterjeszti a kijelölést, vagy létrehozza azt, ha még nem volt kijelölt rész. A függvény általában megkíséri a érzékelni a megnyomott kombináció shift nélküli billentyűjéhez tartozó függvényt, és a kijelölés feldolgozása után ezt hívja. Itt a `this-command-keys` függvényt használja, amely a függvényt hívó billentyűkombinációt adja vissza.

```
(defun S - Key (keysym)
  "return the keysym without S-"
  (let ((name (symbol - name keysym)))
    (vector
      (make - symbol
        (substring name 2 (length name))))))
(defun mark - and - do ()
  "set the mark if not active and
  do command without shift"
  (interactive)
  (if (not mark - active)
    (push - mark nil t))
  (let ((key (aref (this - command - keys) 0)))
    (call interactively
      (key - binding (S - Key)))))
(defun mouse - describe - function (event)
  "describe function under the mouse - cursor"
  (interactive "e")
  (save - excursion
    (mouse - set - point event)
    (let ((fn (function - called - at - point)))
      (describe - function fn)
      nil)))
```

Magának a billentyű-hozzárendelésnek a megadása viszonylag egyszerű: a billentyű és a hívandó függvényt kell paraméterként átadni a `define-key` vagy a `global-set-key` függvénynek. Ezután a `define-key` újból hívására kerül sor a billentyűtérképpel, amelybe a hozzárendelést kell bevinni. A `function-key-map` nevű billentyűtérkép különös szerepet játszik itt. Ez a billentyűszekvenciákat más billentyűkkel vagy szimbólumokkal helyettesíti, mielőtt ezeket más billentyűtérképek feldolgoznák. A `function-key-map` bejegyzéseit általában az ASCII szekvenciákat funkcióbillentyűket küldő terminálokra illesztésre használják. Az egyszerű kurzormozgató billentyűk kódjai általában már megtalálhatók a rendszer `termcap` vagy `terminfo` adatházból, és így az Emacs már ismeri ezeket.

```
;: Special cursor Keys on Linux Console
(define - key function - key - map "\e[1 " [home])
(define - key function - key - map "\e[4 " [end])
(define - key function - key - map "\e[2 " [insert])
;: Control - cursor
(global - set - key [C - right] 'forward - word)
(global - set - key [C - left] 'backward - word)
```

```

(global - set - key [C - prior]  'beginning - of - buffer)
(global - set - key [C - next]   'end - of - buffer)

;; Shift - cursor
(global - set - key [S - right]  'mark - and - do)
(global - set - key [S - left]   'mark - and - do)
(global - set - key [S - up]     'mark - and - do)
(global - set - key [S - down]   'mark - and - do)
(global - set - key [S - end]    'mark - and - do)
(global - set - key [S - prior]  'mark - and - do)
(global - set - key [S - next]   'mark - and - do)

;; Function Keys
(global - set - key [f1]         'info)
(global - set - key [f2]         'save - buffer)
(global - set - key [f3]         'find - file)

(global - set - key [f5]         'goto - line)
(global - set - key [S - f5]    'what - line)
(global - set - key [f6]         'tags - search)
(global - set - key [S - f7]    'visit - tags - table)

(global - set - key [f9]         'compile)
(global - set - key [M - f8]    'next - error)
(global - set - key [f10]        'next - error)
(global - set - key [f12]        'advertised - undo)

;; Redefine home and end
(global - set - key [home]      'beginning - of - line)
(global - set - key [end]        'end - of - line)

;; Redefine backspace and delete
(define - key function - key - map [delete]  [deletechar])
(define - key function - key - map [backspace] [DEL])
(global - set - key [DEL]       'delete - backward - char)

;; divers
(define - key ctl - x - map "p"  'toggle - save - place)
(define - key lisp interaction - mode - map [M - return]
  'eval - print - last - sexp)

(define - key emacs - lisp - mode - map [S - mouse - 1]
  'mouse - describe - function)
(define - key lisp interaction - mode - map [S - mouse - 1]
  'mouse - describe - function)

```

Új menü definiálása hasonló a billentyű-hozzárendelések megadásához, és tárolásuk is a szokásos billentyű-hozzárendelésekkel együtt történik. A következő példa helyi menüt definiál az Emacs Lisp üzemmód számára.

```
(when window system
  ;; New menu in elisp mode
  (define-key emacs-lisp-mode-map [menu-bar elisp]
    (cons "Elisp" (make-sparse-keymap "elisp")))
  (define-key emacs-lisp-mode-map [menu-bar elisp debonenoff]
    '("Cancel Debug on Entry" . cancel-debug-on-entry))
  (define-key emacs-lisp-mode-map [menu-bar elisp debonen])
  ('("Debug on Entry" . debug-on-entry)))
  (define-key emacs-lisp-mode-map [menu-bar elisp debdefun]
    '("Debug Defun" . edebug-defun))
  (define-key emacs-lisp-mode-map [menu-bar elisp evalbuff]
    '("Eval Buffer" . edebug-buffer))
  (define-key emacs-lisp-mode-map [menu-bar elisp evalreg]
    '("Eval Region" . eval-region))
  (define-key emacs-lisp-mode-map [menu-bar elisp evaldef]
    '("Eval Defun" . eval-defun)))
```

A `define-key` első paramétere a billentyűtérféket tartalmazza, míg a második a menü-hierarchiában elfoglalt pozíciót. A `[menu-bar elisp]` új legördülő menüt definiál. A harmadik paraméter a szöveget és a hozzá tartozó műveletet adja meg. A `define-key` első hívásakor az utolsó paraméter nem tényleges műveletet definíál, hanem az új legördülő menü billentyűtérfépet. A további hívások mindenkor egy `cons` cellát tartalmaz, amelynek első eleme a menüszoveget, míg második eleme a hivandó függvény nevét adja meg.

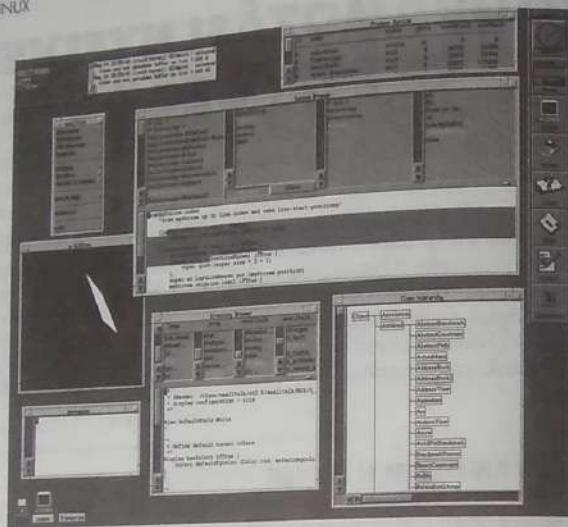
# PROGRAMOZÁSI NYELVEK ÉS ESZKÖZÖK

**A** Linux egyik kiemelkedő jellemzője, hogy a Linux FTP kiszolgálókról bináris és forrás formában számos ingyenesen letölthető programozási nyelv érhető el, és maguk az egyes Linux-változatok is szép számmal tartalmaznak ilyeneket. Ezek az eszközök képességeikkel gyakran elérik a kereskedelmi termékek szintjét. Ez a fejezet a Linux alatt rendelkezésre álló különböző fordítóprogramokat, értelmezőket és szoftverfejlesztő eszközöket mutat be, kiemelve legfontosabb jellemzőiket. (Fontos megjegyezni, hogy a programozási nyelvek jelentős része az FSF Gnu projektjéből származik, ezeket minden GNU programokkal foglalkozó FTP kiszolgálóról le lehet tölteni – a lektor.)

## 14.1 PROGRAMOZÁSI NYELVEK

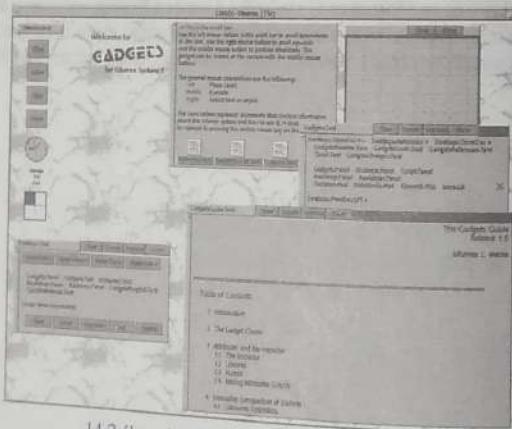
Vitathatatlan, hogy a UNIX-rendszerben a C programozási nyelv a legelterjedtebbet használt. Magukat a mai UNIX-rendszerket, valamint az ezekhez készült legtöbb programot is ezen a nyelven fejlesztették. Így nem meglepő, hogy a UNIX fejlesztői környezet központi eleme egy C fordítóprogram.

A korábbi kereskedelmi UNIX-rendserek szinte kivétel nélkül tartalmaztak egy C fordítóprogramot. Mára ez némiépp megváltozott: az újabb PC-s felhasználói UNIX-változatok C fordítóprogram és hálózati támogatás nélkül kerülnek forgalomba (ez nem csak a PC-s verzióra igaz, az AT&T SVR4 szabvány szerint nem szokás C fordítót adni a Unix rendszerekhez). A teljes verziók tartalmazzák ezeket az elemeket, de lényegesen drágábbak. A Linux alatt használható nyelvek skálája széles: a C, a C++ és az Objective C nyelvtől a Lisp és a Prolog, valamint a Smalltalk (lásd a 14.1. ábrán) és Forth nyelvig sokféle megtalálható. A klasszikus nyelvek, például Fortran vagy APL, szintén rendelkezésre állnak Linux alatt. Még Basic programok is fejleszthetők két értelmező segítségével.



14.1 ábra. A Smalltalk/X a Linux alatt

A fordítóprogramok a Modula-2 és a Modula-3, valamint az Oberon rendszerben (lásd a 14.2. ábrán) használhatók. E fordítóprogramok nemelyike kereskedelmi rendszerek, és más UNIX rendszerek alatt is használható. Az Ada9X számára a New York-i Egyetemen kifejlesztett Ada fordító, a GNAT, Linux alatt is megtalálható. Ez a GCC háttérprocesszort használó rendszer teljes szolgáltatást nyújt. Mivel az Ada9X szabvány végső kidolgozása előtt került forgalomba, hivatalos megfejtése még nem történt meg.



14.2 ábra. Az Oberon System 3 Linux alatt

A jelenlegi fejlesztések ismeretében elmondható, hogy a közeljövőben szinte valamennyi nyelv elérhető lesz Linux alatt is.

## 14.2 C FORDÍTÓPROGRAMOK

A Linux alatt használt C fordítóprogram a Free Software Foundation GNU C fordítóprogramja (`gcc`), amely optimalizált kód állítására képes. Mivel a `gcc` számos UNIX platformon megtalálható, így a szoftver más számítógépekre könnyen átvihető. A GNU C fordítóprogrammal a C, a C++ és az Objective C ANSI és K&R szabványai használhatók. Kiemelést érdemelnek a fordítóprogram szintaktikai hibák nélküli megjelenő részletes hibaüzenetei.

## 14.3 PASCAL, FORTRAN, SIMULA ÉS MODULA-2

A GNU C fordítóprogram a Pascal-, a Simula- és a Fortran-C konvertert is tartalmazza. Ezek a konverterek meglévő Pascal, Simula vagy Fortran forráskódot olvasnak be, és ebből C programot állítanak elő, ami a C fordítóprogrammal fordítható. Ez az integráció a C átmeneti állapotát teljesen észrevélténné teszi a programozók számára. (Ezt már az idő túlhaladta, elérhető Pascal, illetve Fortran kiegészítés a Gnu C-hez, amelyek köztes C állapot nélkül generálnak tárgykódot `gpc`, illetve `g77` néven – a lektor.)

A szabványos Pascal mellett a Pascal-C konverter számos Pascal dialektust is fordít, így a Turbo Pascal változatot az 5-ös verzióig és a Macintosh Pascalt. A Simula-C konverter hasonló elven működik. A Német Matematikai és Adatfeldolgozási Társaság (Gesellschaft für Mathematik und Datenverarbeitung, GMD) a MOCKA Modula-2 fejlesztői környezetét Linux alatt is megvalósította. Érdekesség, hogy maga a fordítóprogram Modula-2 rendszerben készült, és ennek forráskódját a programcsomag tartalmazza. A fenti eszközökkel ellentétben a MOCKA közvetlenül tárgykódot állít elő. A MOCKA lehetővé teszi C rutinok *idegen modulként* való kapcsolását is.

## 14.4 LISP ÉS PROLOG

Mivel az ismeretbázisú rendszerek és a mesterséges intelligencia területén használt programozási nyelvek fontos szerepet játszanak az egyetemeken, számos ilyen megvalósítás áll rendelkezésre. Itt a Lisp kiemelt jelentőségű, mivel a GNU Emacs programozási nyelveként viszonylag széles körben elterjedt.

Az egyik, objektumorientált bővítméssel (a CLOS részhalmaza) rendelkező Common Lisp megvalósítás, a `clisp` a legtöbb Linux-rendszerben megtalálható. Az átfogó GNU Common Lisp a Common Lisp új ANSI szabványára összpontosít.

A `swi.psy.uva.nl` FTP kiszolgáló az Amszterdami Egyetem átfogó Prolog megvalósítását tartalmazza, amely könnyen fordítható Linux alatt. A Warren Abstract Machine (WAM) rendszeren alapul, és jól használható súgóit is tartalmaz. Érdekessége az XPCE objektumorientált rendszerhez való kapcsolata, amely szintén Amszterdamban készült. Ez a rendszer így grafikus programozási környezetet biztosít a Prolog számára.

## 14.5 A TCL NYELV

Függetlenül attól, hogy szöveget, adatbázist vagy modemet kezelő programról van szó, a jó szoftverhez szükség van egy kiegészítő parancsnyelvre, amelyen a felhasználók makrókat vagy szkripteket írhatnak. Ezek műveletek automatikus elvégzésére vagy a rendszer egyszerű bővítésére használhatók. Ilyen jól ismert nyelvek például a következők: Emacs Lisp, elk és a Microsoft Visual Basic for Applications (VBA).

Az 1988-ban John Ousterhout ilyen nyelvként fejlesztette ki a **Tcl** (Tool Command Language) nyelvet. A Tcl egyszerű szintaxisával rendelkezik, és egyetlen adattípust (karaktersorozat) tartalmaz. A követelményektől és a környezettől függően ezeket a karaktersortokat a program számként, listaként vagy egyéb adattípusként értelmezi. A nyelv erősségeit előszörban a számos magas szintű függvény és a Lisphez hasonlóan az a tény jelenti, hogy nem tesz különbséget az adatok és a program között. Maga a Tcl kód kerül tárolásra, és ezt értelmezi karaktersortokat. Ez egyszerű véteszi a nyelvi szerkezeti bővítek vagy az egyéni kód megvalósítását.

A nyelv C könyvtárként valósult meg és a meglévő programokkal összekapcsolással és néhány függvény végrehajtásával kombinálható. Független programokhoz közvetlenül végrehajtható értelmezőkkel rendelkezik, amelyek egyetlen bemenetet olvasnak, ezt átadják a Tcl értelmezőnek, majd szolgáltatják az eredményt. A következőben a Tcl legfontosabb nyelvi elemeinek bemutatása néhány egyszerű példát ismertetünk. A Tcl értelmező a **tcl** vagy a **tclsh** beírásával hívható.

### A nyelv

A Tcl nyelvben a lista olyan karaktersort, amelynek elemeit szóközökkel, tabulátorokkal vagy sortöréssel választják el egymástól. Egy Tcl utasítás olyan lista, azaz olyan karaktersort, amelyben az első elem a hivandó függvény neve. A lista többi eleme a függvény argumentumaként kerül átadásra.

```
nicetry:-$ tclsh
% set a 5
% set a 5
5
% puts $a
% puts $a
5
% puts "The value of a is $a"
% puts "The value of a is $a"
The value of a is 5
%
```

A fenti példában a tcl értelmező először a **set a 5** sort értékeli ki. A **set** függvény értékét rendel egy változóhoz (a példában ez a). A Tcl nyelvben a változókat nem kell deklarálni: egy változó létrehozásához egyszerűen értékét kell rendelni hozzá. A következő sor az a aktuális értékét adja vissza. A Tcl programkód egy sorának kiértékelése előtt az értelmező helyettesítéseket hajt végre, amelyeket különleges karakterekkel vezet be. A \$ karakter például azt jelzi, hogy az utána következő nevet változóként kell kezelni, és ezt értékével helyettesíti. Ha a megadott néven nem létezik változó, hibaüzenetet kapunk.

Ez a változókiértékelési mechanizmus a programsor minden részén érvényesül, kivéve, ha különleges idézettel ezt le nem tiltjuk. A következő példa azt mutatja be, hogy még egy függvény-név sem kivétel ez alól.

```
% set fkt puts
% set fkt puts
puts
% set arg "The function $fkt was invoked here"
% set arg "The function $fkt was invoked here"
```

```
The function puts was invoked here
% $fkt $arg
% $fkt $arg
The function puts was invoked here
```

Az utolsó sor a `puts` függvényt hívja. A függvény argumentuma a „`The function $fkt was invoked here.`” („Itt a `$fkt` függvény hívására került sor.”) karakterszorozat.

A kiértékelés egy másik típusa a szöglletes zárójelcs ([ ]) visszatérési értékkal hívható. A zárójel közé tett szöveget Tcl függvényhívásként értékeli ki az értelmező, majd a függvény visszatérési értékével helyettesíti. A következő példa a `lindex` függvényt használja, amely egy karakterszorozatot listaként értelmez, és ennek meghatározott elemét adja vissza. A Tcl nyelvben a lista elemeinek számozása a 0 értékkel kezdődik.

```
% set l "This is a list."
% set l "This is a list."
This is a list.
% puts "The 2nd element is: [lindex $l 1]"
% puts "The 2nd element is: [lindex $l 1]"
The 2nd element is: is
```

A Tcl nyelv egyik érdekes eleme a tömb, amelynek indexéreit nem korlátozódnak numerikus értékekre, hanem tétszőleges karakterszorozatok megengedettek itt. A következő példában a `grade` tömbként használt, míg az indexek kitalált hallgatók nevei.

```
% set grade(sam) 2.3
% set grade(sam) 2.3
2.3
% set grade(fred) 1
% set grade(fred) 1
1
% set grade(peter) 4
% set grade(peter) 4
4
% puts $grade(fred)
% puts $grade(fred)
1
% set name sam
% set name sam
sam
% puts "The grade average for $name is $grade($name)"
% puts "The grade average for $name is $grade($name)"
The grade average for sam is 2.3
```

A Tcl az eljárásos nyelvek szokásos vezérlőelemeit nyújja. Az `if`, a `switch`, a `for` és a `while` mellett a `foreach` ciklus is megtalálható, amely egy lista minden egyes elemére egy utasításblokkot hajt végre. Ez lehetővé teszi utasítások megismétlését egy asszociatív tömb minden elemére, amint ezt a következő példa is mutatja:

```

% set students [array name grade]
% set students [array name grade]
peter fred sam
% foreach name $students puts "The grade average for $name is
$grade($name)"
% foreach name $students puts "The grade average for $name is
$grade($name)"
The grade average for peter is
The grade average for fred is 1
The grade average for sam is 2.3

```

Az array függvény létező tömbökről szolgáltat információt. A name s argumentummal használva a tömb azon elemeinek indexét adja vissza, amelyek értéket tartalmaznak.

A Tcl a reguláris kifejezések (lásd a 2.8. szakasz) kezelésére is tartalmaz függvényeket. A regexp függvény egy reguláris kifejezést hasonlít össze egy karakterszorozattal, és I a visszatérési érték, ha a kifejezést megtalálta. Az is lehetséges, hogy a megadott kifejezéssel vagy annak al-kifejezésével egyező al-karaktersorozatokat változókhöz rendeljük.

```

% set s "123abchello6677plist7zz"
% set s "123abchello6677plist7zz"
123abchello6677plist7zz
% set pattern hello.*?([a-z]+)7
% set pattern hello.*?([a-z]+)7
hello.*?([a-z]+)7
% regexp $pattern $s all sub1
% regexp $pattern $s all sub1
1
% puts $all
% puts $all
hello6677plist7
% puts $sub1
% puts $sub1
plist

```

A kapcsos zárójelek {} az idézőjelekhez hasonló funkcióval rendelkeznek, azonban ebben az esetben nem lehetséges a további belső helyettesítés. A fenti példában a reguláris kifejezést kapcsos zárójelek közé kellett tenni, mivel egyébként az [a-z] al-karaktersorozatot függvényhívásként értelmezte volna a Tcl.

A Tcl nyelvben az egyéni függvények a proc függvény hívásával definiálhatók. Az új függvény argumentumai egyedi változókként vagy listaváltozókként adhatók meg. Az első esetben a paraméterek száma kötött, míg a második esetben dinamikusan változhat.

```

% proc myFunction arg1 arg2 puts "The arguments are $arg1 and $arg2" ;
return 1
% proc myFunction arg1 arg2 puts "The arguments are $arg1 and $arg2" ;
return 1
% myFunction a b

```

```
% myFunction a b
The arguments are a and b
1
% proc funct2 args puts "The arguments are $args"; return bye
% proc funct2 args puts "The arguments are $args"; return bye
% funct2 a b c d e
% funct2 a b c d e
The arguments are a b c d e
bye
```

A Tcl függvényekben szereplő változók általában **lokálisak**. Globális változók létrehozásához cze-  
ket előbb egy **global** sorban meg kell adni.

```
% set name Sam
% set name Sam
Sam
% proc Test name puts "name is $name"
% proc Test name puts "name is $name"
% Test
% Test
no value given for parameter "name" to "Test"
% proc Test global name; puts "Name is $name"
% proc Test global name; puts "Name is $name"
% Test
% Test
Name is Sam
%
```

Tcl szkriptek végrehajtható fájlként való használatához a Tcl értelmezőt az első sorban a #! után shellprogramként kell definiálni. Ha a felhasználók jogosultak a szkriptfájl végrehajtására, akkor a szkript nevének megadásával hívhatják azt. A következő példa egy szkriptfájl indítását mutatja be:

```
#!/usr/bin/tcl
puts "Hello world!"
```

## Alkalmazások és kiterjesztések

A Tcl nyelvvel alapvetően két módon lehet alkalmazásokat írni. Az első esetben egy C nyelven írt alkalmazás a Tcl értelmezővel egy szkript segítségével terjeszthető ki. A szkriptnyelv alkalmazás-hoz csatolása további Tcl parancsok definiálásával történik. Ez a megközelítés elsősorban meglévő programknál lehet hasznos. Új fejlesztések nélkül még a főalkalmazás is kifejleszthető a Tcl-ben. A rendelkezésre nem álló függvények Tcl kiterjesztésként hozhatók létre a C nyelvben, és a Tcl nyel-ven keresztül érhetők el.

A **Tcl kiterjesztése** igen egyszerű. Egy C függvényt kell definálni, amely hasonlóan egy C-beli main függvényhez az argc és az argv függvényen keresztül kapja a paramétereit. Ezután a Tcl értelmező inicializációs üzemmódjában a **Tcl\_CreateCommand** függvényt kell hívni, amely bejegyzi az új függvényt, és a Tcl-en belül nevét rendel hozzá.

Számos Tcl nyelvi kiterjesztés ingyenes rendelkezésre áll. Ezek a kiterjesztések lehetővé teszik SQL adatbázisok elérését vagy UNIX rendszerek hívását. Például a *Tcl-dp* hálózati programozási függvényeket biztosít (foglalatok és RPC), míg az *itcl* a Tcl nyelvet osztályokkal és objektumorientált programozási metódusokkal bővíti. Ezek mellett egy hibakereső és egy osztálytallázó is rendelkezésre áll.

A TCP/IP foglalatok és a Sun RPC-k mellett a *scotty* kiterjesztés a DNS eléréséhez és az SNMP-n kereszttüli hálóztafelülethez is biztosít függvényeket. A következő példa egy scotty kiterjesztéssel írt egyszerű kiszolgálót mutat be. Ez egy TCP foglalatot nyit meg, és a kapcsolatokra vár. Az ügyfél részére a Telnet program az egyik lehetséges kapcsolatfelvételi lehetőség. Amikor egy ügyfél a kiszolgálónak kapcsolódik, a kiszolgáló az aktuális dátumot és időpontot küldi az ügyfelnök, majd az ügyfél válaszára vár.

```
#!/usr/local/bin/scotty -f

set port 1371

proc handleConnection lsock
    set socket [tcp accept $lsock]
    # send date and time to client
    puts $socket [exec date]
    # wait for reply from client
    puts "Msg from Client: [gets $socket]"
    close $socket

set lsocket [tcp listen $port]
addinput -read $lsocket "handleConnection %E"
puts "Waiting for connections on port $port..."
```

Az ügyféloldal a következőképpen néz ki:

```
nicetry:~$ telnet localhost 1371
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Fri Aug 30 10:31:03 1996
Hello
Connection closed by foreign host.
```

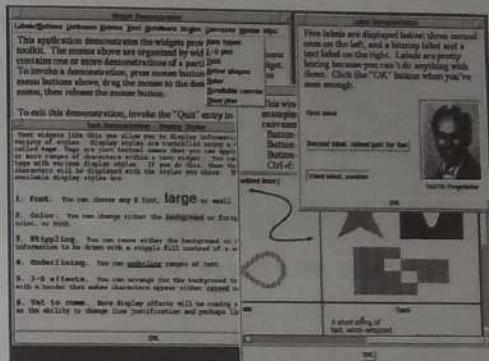
## Tk

A Tk mind a C, mind a Tcl nyelvhez kapcsolatot biztosító elemgyűjtemény. Ily módon különösen jól használható grafikus felhasználói felületen Tcl programok előállítására. A rendelkezésre álló lesztísszére egyaránt jól használható.

A *Tcl/Tk* alkalmazások jellemző példái a következők: Zircon, tkined, Picasso, valamint számos más rendszer, amely kiterjeszeti nyelvként a Tcl-t használja.

**XF**

A Tcl/Tk felületfejlesztője, az XF egy Tk felhasználói felület grafikus és interaktív tervezését teszi lehetővé (lásd a 14.3 ábrán). Ez nemcsak a Tk felületek egyszerű megvalósítását biztosítja, hanem meglévő Tcl/Tk programok betöltését, módosítását és kiterjesztését is. Mivel maga az XF is a Tcl/Tk rendszeren került kifejlesztésre, a felhasználói felület már a fejlesztés során ellenőrizhető, és a fejlesztés során kapott megvalósítás pontosan megfelel a végző programnak.



14.3 ábra. Az XF, a Tcl/Tk felületfejlesztője

**Tcl/Motif**

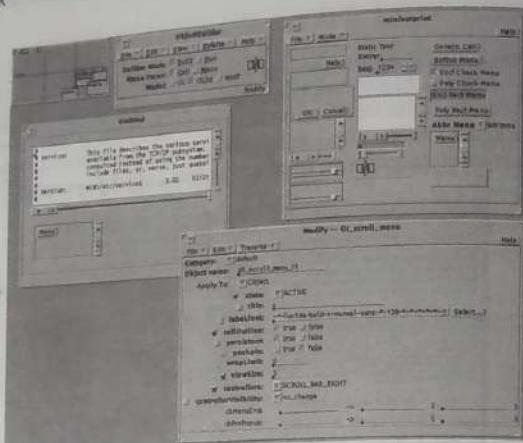
A Tcl és a Motif gyűjtemény előnyeinek egyesítéséhez érdemes kipróbálni a Tcl/Motif alkalmazást. Ez a csomag a szokásos Motif elemgyűjteményt használja a grafikus felületek létrehozásához. Így a Motif felhasználói felületek egyszerűen egy értelmezői környezetben tervezhetők. A programozó szinte az összes szokásos erőforrást használhatja. A Tcl/Motif még az egérrel való húzás mechanizmusát is támogatja.

Az <ftp://ftp.aud.alcatel.com> FTP kiszolgáló számos Tcl kiterjesztést, valamint végleges Tcl/Tk alkalmazásokat is tartalmaz. A következő WWW oldal áttekintést nyújt a Tcl kiterjesztésekről, dokumentációkról és programokról:

<http://web.cs.ualberta.ca/~wade/Auto/Tcl.html>

**14.6 FELÜLETFEJLESZTŐK**

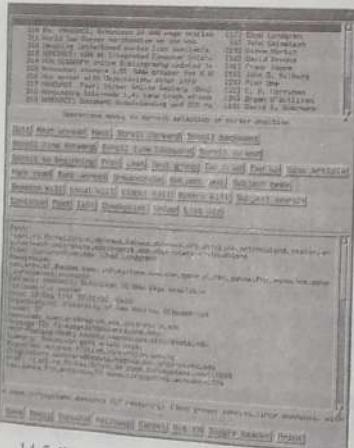
Az X11 alapú alkalmazások előállításának legkényelmesebb módja a grafikus felhasználói felületek interaktív tervezésére szolgáló egyedi szoftverek használata. Ezek a felületfejlesztők általában C forráskód előállítására alkalmasak, amelyet a programozók szükség szerint kiterjeszthetnek. A ParcPlace kínál ilyen felületfejlesztőt, amelynek neve **ObjectBuilder** (lásd a 14.4. ábrán). Ez a grafikus objektumok közvetlen kezelését biztosítja az egérrel, és C++ forráskódot állít elő. Az **ObjectLibrary**val együtt használva a felhasználói felületek az OpenLook vagy a Motif előírások szerint állíthatók elő. A parancssori kapcsolókkal a külalak állítható be. Mindkét termék az összes általánosan használt UNIX platformon rendelkezésre áll, mintegy cser dolláros áron. Egyedül a Linux-verzió tölthető le ingyenesen. Az igényesebb felhasználók számára azonban a ParcPlace forgalmazásában rendelkezésre álló kézikönyvekre is szükség lehet.



14.4 ábra. A ParcPlace felületfejlesztője

### VXP

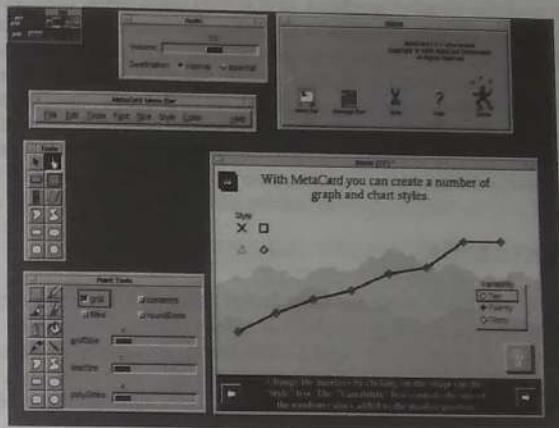
Bár az ingyenes, Young Chen által készített VXP felületfejlesztő még fejlesztés alatt áll, már is egyszerűen használható kisebb OSF/Motif alapú felhasználói felületek létrehozására (lásd a 14.5 ábrán). A ParcPlace felületfejlesztőjével szemben a VXP tisztá Motif forráskódot állít elő. Az összes grafikus objektum az egérrel kezelhető és pozicionálható. A VXP ezután előállítja a kapcsolódó C forráskódot és egy megfelelő make-fájlt. Az alkalmazás funkcionalitása, szokás szerint a Motif alatt, C nyelvű hívörutinokkal került megvalósításra. A VXP a programozó által felvett forráskódot is kezeli. Emellett a fejlesztői környezet elhagyása nélkül a C fordítóprogram is egy egérkattintással elindítható. A VXP által előállított forráskódhoz a Motif könyvtár mellett nincs szükség egyéb könyvtári rutinokra, így az könnyen átvihető egyéb UNIX platformokra.



14.5 ábra. A VXP Motif felületfejlesztő

## 14.7 METACARD

A Metacard **Hypercard** alkalmazások kifejlesztésére szolgáló rendszer (ezeket az Apple Macintosh használja kiterjedten). A Metacard az Apple Hypercard vermeket is képes betölteni és feldolgozni. Az integrált programozási nyelv lehetővé teszi kisebb alkalmazások vagy grafikus felhasználói felületek prototípusainak egyszerű kifejlesztését. A Metacard az összes általánosan használt UNIX platformon rendelkezésre áll, de kereskedelmi termék (lásd a 14.6. ábrán).



14.6 ábra. A Metacard a Linux alatt

## 14.8 AWK, GAWK

Az **awk** olyan hagyományos UNIX-eszköz, amellyel karakterSORozatok és szöveges fájlok feldolgozására szolgáló kisebb szkriptek hajthatók végre. A program neve készítőinek (Aho, Weinberger és Kernigham) kezdőbetűjéből származik. Mivel az **awk** értelmezőnyelv nagyon hasonlít a C nyelvre, használata viszonylag könnyen megtanulható.

Az **awk** nem tesz különbséget a numerikus és a karakteres változók között, és a változók deklarálása sem szükséges. A kisebb munkáknál az **awk** a prototípus előállításának eszköze lehet.

A következő szkript az aktuális könyvtárban lévő fájlok méretét összegzi, majd megjeleníti az eredményt:

```
nicetry:~$ ls -l | awk ' sum += $5 END print "Sum: " sum'
Sum: 7688
nicetry:~$
```

Az eredeti segédprogram nem áll rendelkezésre a Linux alatt, de a Free Software Foundation GNU **awk** (**gawk**) megvalósítása igen. Az **awk** mellett a kereskedelemben kapható UNIX-rendserek gyakran tartalmazzák a **nawk** nevű bővített verziót.

## 14.9 PERL

A Perl is értelmezőnyelv. A sed, az awk és a szokásos UNIX shellprogramok legfontosabb jellemzőit egyesíti, és különösen szövegfájlokfeldolgozására alkalmas.

A Perl lehetővé teszi listák és asszociatív tömbök használatát, ahol az adatstruktúrák méretét egyszerűen rendelkezésre álló memória korlátossa, és így viszonylag nagy mennyiségű adatfeldolgozására nylik lehetőség. Természetesen a Perl rendelkezik a szokásos vezérlőelemekkel is, illetve a ciklusok, a szubrutinok, a rekurziók, kezelhetünk benne reguláris kifejezéseket. Az eredmények kezelésére szolgáló formázó utasítások segítségével áttekinthető táblázatok és jelentések készíthetők. Az integrált hibakereső töréspontok elhelyezését és a program lépésenkénti végrehajtását teszi lehetővé.

A Perl a rendszeralminisztrációs feladatok végrehajtására is alkalmas, mivel a segítségével előállított szkriptek root jogosultsággal végrehajthatók. Említésre méltó továbbá még az is, hogy a szabványos C könyvtár és a UNIX-rendszer mag szinte valamennyi rutinja közvetlenül végrehajtható a Perl alatt, beleértve a hálózati programozási függvényeket is. A Perl új 5-ös verziójá objektumorientált bővítményeket is tartalmaz.

## 14.10 SZERKESZTŐK

Egy fordítóprogram önmagában nem alkot teljes értékű fejlesztői környezetet. Legalább még egy megfelelő szerkesztőprogramra és egy szimbolikus hibakeresőre is szükség van. A GNU Emacs szerkesztő elegendő ezeket az elemeket (lásd a 14.7. ábrán). Ez számos segédprogramot hívhat, ilyen a grep és az RCS, valamint a C fordítóprogram és a GNU hibakereső, így a teljes fejlesztési folyamat végrehajtható a szerkesztőprogramon belül. Az Emacs részletes ismertetését a 13. fejezet tartalmazza.



14.7 ábra. Szoftverfejlesztés az Emacs szerkesztővel

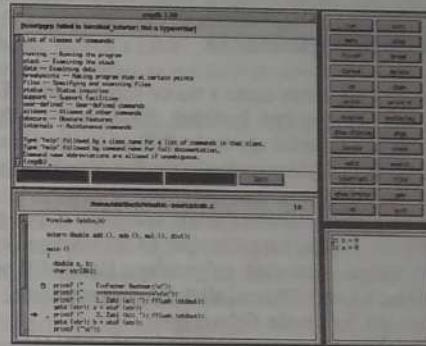
A GNU Emacs mellett a Linux-változatok számos egyéb szerkesztőprogramot is tartalmaznak, ezek áttekintése a 12.2. szakaszban található.

## 14.11 A GNU HIBAKERESŐ (GDB)

A GNU hibakereső (GDB) olyan hatékony hibakereső, amely a C, C++, Fortran, Ada és a Modula-2 nyelvben használható. Ez az összes olyan szolgáltatást tartalmazza, amelyet a fejlesztők egy ilyen eszköztől elvárnak. A programok lépésenkénti végrehajtása is lehetséges. A töréspontok a végrehajtás megadott pontokban való megszakítását teszik lehetővé, míg a figyelőpontok akkor szakítják meg a program végrehajtását, ha a megadott értékek megváltoznak. A kijelölt változók vagy objektumok értékei folyamatosan megjeleníthetők.

A GDB újabb verziói a távoli hibakeresést is lehetővé teszik, amely azt jelenti, hogy a hibakereső a vizsgált programról eltérő számítógépen is futhat. A gépek közötti kapcsolat lehet soros interfész vagy hálózaton keresztüli. A Linux alatt még az operációs rendszer is elemezhető a GDB hibakeresővel, és végrehajtható a futtató processzek hibakeresése.

A GNU hibakereső csak egyszerű, parancsori felhasználói felületet biztosít, de grafikus előfel dolgozókkal a hibakereső kényelmes egérvezérelt működtetése valósítható meg. Ezek az előfeldol gozók a GDB hibakereső második processzként indítják el, és átirányítják a bemenetet és a kimenetet. Ily módon szimulálják a felhasználói bemenetet, és a hibakereső kimenetet különböző ablakba irányítják. Ilyen grafikus kezelőfelület például az xxgdb (lásd a 14.8. ábrán).



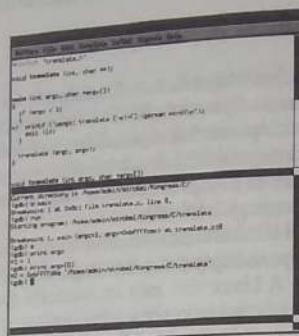
14.8 ábra. Az xxgdb, a GDB grafikus kezelőprogramja

A felső ablaktábla a forráskódban mutatja az aktuális pozíciót. A középső területen az alapvető hibakereső műveleteket hívó parancsok találhatók. Az alsó ablaktáblában a felhasználó szöveges parancsokat írhat be, ha a billentyűzetről bevitt parancsok túlnek hasznosabbnak.

A GDB az alsó ablaktáblában lehetővé teszi a változók értékeitnek folyamatos megjelenítését. A C forráskódban előforduló kifejezések egyszerűen kiértékelhetők: az egérrel ki kell jelölni a forráskódban a kifejezést, ezután a print gombra kell kattintani, és az eredmény megjelenik az alsó ablaktáblában.

A töréspont definíálása hasonlónan történik: a szövegkúrzon a forráskód ablakban a kívánt pozícióba kell helyezni, majd a megfelelő parancsombra kell kattintani. A töréspontot kéz-szimbólum, míg a forráskódban az aktuális pozíciót kék színű nyíl jelzi.

A GNU hibakereső kényelmesebb használatának másik módja integrálása az Emacs szerkesztővel. Az Emacs egyik különleges üzemmódja lehetővé teszi programok hibakeresését egy szokásos szerkesztőablakban, amely két ablaktáblára oszlik. A felső ablaktábla a forráskódot jeleníti meg, és az aktuális pozíciót nyíllal jelzi. Az alsó ablaktábla a hibakeresőt vezéri, és parancsok bevitelét teszi lehetővé.



14.9 ábra. A GDB az Emacs szerkesztőben

A GDB használata az Emacs keretében bizonyos előnyt jelent az Emacs szerkesztő jól ismerők számára, de kétségtelen, hogy nem használható olyan kényelmesen, mint X-alapú előfeldolgozóval.

## 14.12 A MAKE SEGÉDPORGRAM

A C fejlesztői környezet egy másik fontos eleme a `make` segédprogram. Ez a parancs lényegesen egyszerűsíti a több modulból álló projektek fordítását. Ennek megvalósítására a programozó az egyes programrészek közötti kapcsolatokat egy `make`-fájlban tárolja. Ha a programozó módosítja valamelyik ilyen modul forráskódját, akkor csak a módosított részeket és az ezektől függő modulokat kell újra lefordítani.

A Linux alatt használt GNU `make` változat számos, a szokásos `make` segédprogramon túlmutató szolgáltatással rendelkezik. Például használható az RCS (Revision Control System), amely a forráskód verziókezelésére szolgáló parancsok gyűjteménye. Ha egy fájl nem található az aktuális könyvtárban, akkor a GNU `make` az RCS nevű alkönyvtárban megkeresi, és automatikusan beolvassa egy modul újabb verzióját. Ez a `make`-fájlra is igaz.

A GNU `make` több implicit szabályt ismer fel, mint a `make` többi verziója. Egy kisebb, három fájlból álló C programhoz tartozó `make`-fájl például a következő alakú lehet:

```
CFLAGS = -g
LDLIBS = -lm

test: test.o sub1.o sub2.o
```

A GNU `make` szabályokat tartalmaz a megfelelő `.c` fájlok ból `.o` fájlok előállítására, és a tárgyfájlok ból álló `test` program összekapsolására a megadott könyvtárakkal. Az Info rendszerben megtalálható GNU `make` dokumentáció részletes áttekintést tartalmaz az összes szabályról és változóról (lásd még a 11.2. szakaszit is).

### 14.13 IMAKE

A különféle UNIX változatok közötti eltérések gyakran komolyan gátolják a platform-független szoftverek kifejlesztését. Ily módon szükséges lehet a make-fájlok különböző változatainak elkezelése, amelyek mindegyike megadott rendszerhez tartozik. Az X Window Systemből ismert Imake segédprogram egy platform-független Imake-fájlból egy adott platformra jellemző make-fájlt állít elő.

Az Imake hívása általában az `xmkmf` szkriptből történik. A helyes működéshez a parancshoz `sablonfájl` szükséges, amely a rendszerre jellemző adatokat tartalmazza. Ezek a fájlok az `/usr/lib/X11/config` könyvtárba kerülnek telepítésre.

### 14.14 RCS

A számos különböző modult tartalmazó és folytonosan új verziók készítését igénylő nagyobb projektök kifejlesztéséhez nélkülözhetetlen egy verziókezelő rendszer. Az ilyen rendszer fontossága növekszik, ha több programozó dolgozik a projekten egyidejűleg.

Mivel egy program fejlesztési ciklusában az összes verzió archiválásra kerül, szükség szerint a programozók bármelyik korábbi verzióra visszatérhetnek. A UNIX System V Source Code Control System (SCCS) rendszere mellett a Linux a még hatékonyabb Revision Control System (RCS) rendszert ajánlja. A szokásos esetben az archivált adatok tárolása külön könyvtárban (RCS vagy SCCS) történik.

A RCS számos parancsa közül a legfontosabbak a következők:

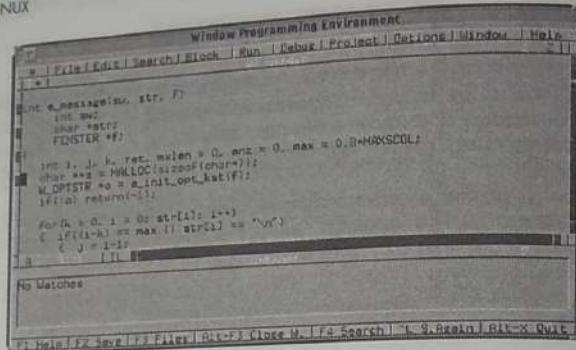
- `c i` (check in): a megadott fájlt az RCS könyvtárba tölti, így rögzíti az aktuális verziót. A teljes verzió nem tárolódik az egyes bevitelkkel, csak az előző verziótól való, a `diff` parancssal meghatározott eltérések. A modul verziószámát a program automatikusan növeli.
- `c o` (check out): a modul aktuális vagy tetszőleges korábbi verziójából nem módosítható másolatot készít. Ha a modult módosítani kívánjuk, ezt a -1 kapcsolóval kell jelezni. Ez megakadályozza, hogy más programozók is módosítani tudják a fájlt, mivel ezt a műveletet csak egy felhasználó és csak egyszer tudja végrehajtani.
- `r log`: különféle adatokat jelenít meg, például egy archív állapotát és a benne lévő modulok verzióit.

Az egyéb parancsok között például olyan is található, amely két verzió egyesítését (összefűlést) teszi lehetővé.

A verziókezelés és a hátralévő fejlesztési munka megfelelő együttműködése érdekében az Emacs szerkesztő újabb verziói az RCS működéséhez külön üzemmóddal rendelkezik.

### 14.15 XWPE

Az `xwpe` fejlesztési környezetként az Emacs érdekes alternatíváját jelenti. Azok a felhasználók, akik a Borland DOS rendszeren működő termékeit (Turbo C, Turbo Pascal stb.) ismerik, hamarosan otthonosan használják ezt a rendszert, mivel az `xwpe` nagyon hasonlít ezekhez (lásd a 14.10. ábrán). Bár az `xwpe` elsősorban szöveges parancsokat elfogadó rendszer, de az egérrel is kezelhető. Az `xwpe` készítői egy színes, X Window System alatt működő terminálemulációs programot is fejlesztettek. Az `xwpe` a szokásos terminálokon is fut, természetesen itt szerényebb megjelenítési lehetőségekkel.



14.10 ábra. Az xwpe fejlesztői környezet

A beépített szerkesztő lehetővé teszi különböző ablakokban több szöveges fájl feldolgozását. Mind a fordítóprogram (`gcc`), mind a hibakereső (`gdb`) a billentyűzetről vagy az egérrel is indítható. A hibализenekek külön ablakban jelennek meg, és a töréspontok közvetlenül a szerkesztőben adhatók meg.

Az xwpe az [ftp.rzrzn.uni-hannover.de](ftp://ftp.rzrzn.uni-hannover.de) FTP kiszolgáló /pub/systems/  
/unix/xwpe könyvtárból töltethető le.

14-16 PÉLDÁK

Az előzőleg ismertetett fejlesztői környezetekben való munkáról szerezhetünk közelebbi benyomásokat, ha áttekinjük a mintaként bemutatott egyszerű C programot. A program több, a bevitt két számot összeadó, egymásból kivonó, összeszorozó és egymással elosztó modulból áll. A főprogram csak a számok bevitelére és az eredmény kiadására szolgáló részből áll. A számításokra különálló modulokban kerül sor. (Természetesen az ilyen egyszerű feladatok esetén fájlban is megoldhatók.)

Az első fájl beviteléhez hívjuk az Emacs szerkesztőt a main.c fájlnévvel mint argumentummal. Mivel a fájl még nem létezik, az Emacs törles puffert nyit meg, amely azonban már C üzeműben van. A következő forrázzöveg bevitelé után a mentés a <Ctrl-x><Ctrl-s> billentyű kombinációval hajtható végre, vagy a parancs a menüből is kiválasztható.

```
#include <stdio.h>

extern double add (), sub (), mul(), div ();

main ()

    double a,b;
    char str[80];

    printf("      Simple Calculator:");
    printf("      1. szam: "); fflush(stdout);
    gets( str); a = atoi(str);
    printf("      2. szam: "); fflush(stdout);
    gets( str); b = atoi( str);
```

```

printf("");
printf(" a + b: %f",add(a,b));
printf(" a - b: %f",sub(a,b));
printf(" a * b: %f",mul(a,b));
printf(" a / b: %f",div(a,b));

printf("Good bye!");

```

## A calc.c fájl

Ezután a többi modul bevitelére kerül sor. A <ctrl-x><ctrl-f> billentyűkombinációval új fájl nyitható meg.

```

double add(double a, double b)

return a + b;

double sub(double a, double b)

return (double)(a - b);

```

## Az addsub.c fájl

```

double mul(double a, double b)

return a * b;

double div(double a, double b)

return a / b;

```

## Amuldiv.c fájl

Ekkor a projekt három fájlból áll, ami a make-fájlban is látható:

```
CFLAGS = -g
LIBS = -lm
OBJS = calc.o addsub.o muldiv.o

calc: $(OBJS)
$(CC) -o $@ $(CFLAGS) $(OBJS) $(LIBS)
```

## A make-fájl

Először a CFLAGS környezeti változó beállítása történik (-g paraméter), amelynek hatására a fordítóprogram hibakereső kódot állít el. Az újonnán definiált OBJS változó a deklarálás után az összes tárgyfájl nevét tartalmazza. A deklarálás csak az áttekinthetőség érdekében került ide, megadása nem feltétlenül szükséges. A lefordított és összekapcsolt program neve calc lesz. A célfájl előállításához három tárgyfájl szükséges. A következő parancs a fordítás után összekapcsolja ezeket a tárgyfájlokat:

```
$(CC) -o $@ $(CFLAGS) $(OBJS) $(LIBS)
```

Egy \* .c fájl \*.o fájiba fordításához nem kell külön szabályt megadni, ezt a make mechanizmus magában foglalja. A make-fájl bevitelkor a link parancsot <tab> karakterekkel kell megelőznie. A make-fájl mentése után a teljes projekt fordítható és összekapcsolható a szerkesztőben kiadott <meta-> compile parancssal. A szerkesztőablak ekkor több részre oszlik. Az alsó ablaktáblában a végrehajtás alatti parancsokat és az előforduló hibaüzeneteket jeleníti meg. A fenti példában a muldiv.c modulban szándékosan kihagyott a pontosvesszőt a div függvénynél. A fordítóprogram a forráskód fordításakor észre fogja venni ezt a szintaktikai hibát.

A <meta-> next-error parancs lehetővé teszi a hiba javítását a szerkesztőben: a forráskódban a hiba előfordulási pozíciójába helyezi a kurzort. A javítás után a <meta-> compile parancssal újra elindítható a fordítóprogram. Természetesen a programozó a fenti parancsokhoz billentyűkombinációkat (parancsbillentyűket) rendelhet, így nem kell bevinne minden alkalommal a teljes parancsot. Az ilyen megadásokat a home könyvtárban található .emacs fájlból érdemes elhelyezni (lásd még a 13. fejezetet is).

Ha a programban logikai hibák fordulnak elő, ezek megtalálásában a GNU hibakereső lehet a segítségünk. Mivel a program fordítása a -g paraméterrel történik, a hibakereső azonnal elindítható.

```
nicetry:~$ xxgdb calc
```

Először a kívánt töréspontokat kell elhelyezni. Ez az egérrel vagy a megfelelő parancs bevitelével történhet. A kurzort a szövegablakban a töréspontnak megfelelő sorba kell helyezni. A hibakereső break parancsgombjára kattintva felvethető a töréspont. Néha egyszerűbb a töréspont beállítása a parancssorból, különösen akkor, ha azt egy függvény legelejére kell felvenni.

```
gdb> break main
```

A fenti parancs leállítja a programot, amint az eléri a main sort.

Az összes töréspont felvétele után a program a run parancssal indítható. Ha a végrehajtás egy töréspontnál megszakad, a futtatás lépésekben üzemmódban folytatható, és a változók értékei megje-

```
gdb> print a
```

A fenti parancs az a változó aktuális értékét jeleníti meg. A `display` parancs a változók értékeinek folyamatos megjelenítését indítja el.

```
gdb> display b
```

A fenti parancs a `b` változó értékét minden egymás utáni `gdb` parancs után megjeleníti. A hibakeresővel ellenőrzött program az Emacs szerkesztőből közvetlenül elindítható. Előtte azonban a `<meta->` shell parancssal egy héjprogramot kell meghyitni. Ezután az összes UNIX parancs használható lesz a szerkesztőn belül. Az Emacs szerkesztőn belül végrehajtott héjprogram előnye az, hogy lehetővé teszi a végrehajtott parancsok egyszerű szerkesztését és az összes szerkesztőparancs végrehajtását (ilyen például blokkok másolása a kimenetre).

## 14.17 SZOFTVEREK ÁTVITELE

A számos FTP kiszolgálón a UNIX-rendszerek számára rendelkezésre álló programok közül csak keveset vittek át kifejezetten a Linux-rendszerre. Mivel a Linux megfelel a létfontosabb UNIX-szabványoknak, a más UNIX-rendszerekre írt programok fordítása általában nem okoz problémát.

### Kibontás

Az FTP kiszolgálókon a programokat általában tömörített tar fájlként tárolják. Az ilyen fájlok a `tar` programmal készültek, és több fájlt és könyvtárt tartalmazhatnak. A fájl neve rendszerint utalást tartalmaz a fájl jellegére. Ha a név `.tar`, `.z` vagy `.Z` végződésű, akkor ez olyan tar archív fájl, amelyet a UNIX `compress` parancssal készítettek. Ha a fájl kiterjesztése `.gz`, `.tgz` akkor ez olyan tar archív fájl, amely tömörítésére a `gzip` programot használták.

Az ilyen fájlokat a Linux alatt a következő módon lehet kibontani:

```
nicetry:~$ tar xvfz <file.tar.z>
```

A `tar` program függetlenül hívja a `gunzip` programot, amely kibontja a `gzip` vagy a UNIX `compress` parancssal tömörített fájlokat.

Az e-mail fájlként elküldött forráskódok általában a shellarchívumokban találhatók, amelyek neve rendszerint `.shar` végződésű. Egy szokásos Bourne shell esetében ilyen archívumból a fájlok az alábbi módon fejthetők ki:

```
nicetry:~$ sh<file.shar>
```

### Dokumentáció

Az archív kifejtése után a felhasználóknak feltétlenül érdemes elolvasniuk a mellékelt `README` vagy `install` fájlokat, amelyek fontos utasításokat tartalmaznak a fordításról. Ezek általában a használható platformokra és az esetleg felmerülő problémákra vonatkoznak.

A programcsomagok gyakran teljes felhasználói dokumentációt is tartalmaznak PostScript vagy TeX fájlként. Legalább egy UNIX hivatkozási Manual oldalt minden képpen mellékelnek.

### **Make-fájl/Imake-fájl**

Egy program forráskódjához mindig mellékelnek egy make-fájlt, amelyben fontos beállítások hajthatók végre, így például a fordítási paraméterek, a könyvtárak elérési útvonalának és a telepítési könyvtár megadása. A make-fájl segítséget jelent a program fordításában és telepítésében is. A make vagy a make install hívása a fordítóprogramot, illetve a szerkesztő programot indítja el, majd telepít a programot. Az összes parancs és beállítási lehetőség részletes ismertetése a README fájlból található, amely szinte minden programcsomagban megtalálható.

Az X Window System szoftverek konfigurálása még egyszerűbb. Az X Window System programai make-fájl helyett általában egy Imake-fájlt tartalmaznak, amely lényegében egy rendszergélekkel megtalálható. Az xmkmf parancs automatikusan létrehozza az Imake-fájlból a megfelelő make-fájlt: a make-fájl a megfelelő rendszer összes lényeges elérési útvonalát és beállítását tartalmazza. Az ilyen programoknál általában elegendő az xmkmf, a make és a make install parancs használata.

### **Konfigurálás**

Az újabb FSF programcsomagok egy configure nevű szkriptfájlt is tartalmaznak. Először ezt a szkriptfájlt kell hívni, amely felismeri az operációs rendszert, és megkeresi a rendszerhez tartozó C fordítóprogramot, az egyéb segédprogramokat és a könyvtákat. Az így megtalált adatokat a fordítás alatt használja a rendszer, így további beállításra nincs szükség.

Az ilyen configure szkriptek általában a GNU autoconf csomaggal hozzájárultak, amely az M4 makróprocesszort használja az olvasható formában a configure.in fájlban lévő adatokból a configure szkript előállítására.

### **Kézi beállítások**

Ha a fordítóprogram a make eljárását hibázenettel fejezi be, akkor a felhasználónak kézi beállításokat kell végreghajtania. Ez gyakran előfordul az olyan programoknál, amelyek nem használják a configure szkriptet. Némi gyakorlattal ezek a beállítások egyszerűen végreghajthatók. Az alábbi példa tipikus hibaüzenetet mutat be:

```
nicetry:~$ gcc bsp.c
bsp.c: In function 'main':
bsp.c: 'errno' undeclared (first use this function)
bsp.c: Each undeclared identifier is reported only once
bsp.c: for each function it appears in.
nicetry:~$
```

Ebben a fordítóprogram nem képes értelmezni valamely szimbólum definícióját. Ennek több oka lehet. Ha a szimbólum állandó vagy makró, akkor ez vagy nem áll rendelkezésre a Linux alatt, vagy a deklarációt tartalmazó fejreszfájl bevétele nem történt meg a #include parancssal. Ugyanez érvényes a C függvényekre is, amelyek prototípusai szintén fejreszfájlokban találhatók. A felhasználónak ekkor a megfelelő Manual oldalon meg kell ismerkednie a függvény működésével és paramétereivel.

Annak meghatározására, hogy egy függvény vagy más szimbólum megtalálható-e a Linux alatt, a rendszerhez mellékelt fájlokat, azaz a Linux C könyvtáranak C fejreszfájait kell megvizsgálni.

Ezek a fejrfájlok az /usr/include könyvtárban találhatók, és a keresés a következő parancssal hajtható végre:

```
find /usr/include -name '*.h' -exec grep 'symbol' -print
```

A find parancs megkeresi az összes fejrfájlt az /usr/include könyvtárban és annak alkönyvtáraiban, majd a find a grep parancsot hívja, amely a szimbólumot keresi az egyes megtalált fájlokban. Ha a definícióatlannak talált szimbólum olyan függvény, amelynek különböző alakjai lehetnek az egyes UNIX-rendszerekben, akkor a forráskód gyakran alternatívákat is tartalmaz, amelyek a -DSYSV vagy a -DBSD fordítási kapcsolóval hívhatók. A forráskódban ez a következő alakú lehet:

```
#ifdef SYSV
    function_a (x,y,z);
#else
    function_b (x,y,z);
#endif
```

Ha a felhasználó módosításokat szeretne végrehajtani a forráskódban, akkor ezeket az #ifdef vagy az #ifndef linux parancsba kell foglalnia. Itt a linux olyan szimbólum, amelyet a fordítóprogram automatikusan deklarál, amikor a Linux alatt hívják.

```
#ifdef linux
    changes
#else
    old code
#endif
```

Ha a fordítóprogram nem jelez hibát, de az összekapcsolt definiálatlan szimbólumokkal kapcsolatos hibaüzenetet ad, akkor az nm program segíthet. Ez a tárgyfájlokban és könyvtárakban deklarált és használt szimbólumok listáját adja meg. A definiálatlan szimbólumok listájában kereső grep parancssal együtt használva ez lehetővé teszi annak megállapítását, hogy egy szimbólum hol használatos és melyik könyvtár tartalmazza azt:

```
nicetry:~$ diff -Nrc linux linux.patched >linux.patch
```

Némi gyakorlatra szert téve a felhasználók néhány perc alatt képesek kisebb programok kezelésére és a fordítás végrehajtására a Linux alatt.

### Archiválás

Egy program forráskódjának megváltoztatása után a módosításokat a diff parancssal fájlba kell menteni. Ezután ezeket a változtatásokat bármikor alkalmazni lehet az credeti programra a patch parancssal, valamint ezeket a módosításokat más felhasználók rendelkezésére lehet bocsátani.

A következő példa a diff használatát mutatja be:

```
hermes:/usr/src# diff -Nrc tcsh-6.05 tcsh-6.05.patched > tcsh.patch
hermes:/usr/src#
```

A változatlan forráskódban a patch/diff fájlból lévő módosítások alkalmazására a patch parancs a következő módon hívható:

```
nicetry:/usr/src/linux# zcat patch-2.0.15.gz | patch -p1
Hmmm... The next patch looks like a unified diff to me...
The text leading up to this was:
-----
|diff -u --recursive --new-file v2.0.14/linux/include/linux/sched.h
linux/include/linux/sched.h
|--- v2.0.14/linux/include/linux/sched.h      Thu Jul 25 20:26:46 1996
|+++ linux/include/linux/sched.h      Sat Aug 24 13:13:15 1996
Patching file include/linux/sched.h using Plan A...
Hunk #1 succeeded at 398.
Hmmm... The next patch looks like a unified diff to me...
The text leading up to this was:
-----
|diff -u --recursive --new-file v2.0.14/linux/kernel/sched.c
linux/kernel/sched.c
|--- v2.0.14/linux/kernel/sched.c      Wed Aug 21 09:18:10 1996
|+++ linux/kernel/sched.c      Sat Aug 24 10:45:10 1996
Patching file kernel/sched.c using Plan A...
Hunk #1 succeeded at 501.
```

# LINUX PARANCSOK

---

## apropos terms fogalmak

A paraméterként átadott fogalmaknak megfelelő parancsleírásokat keresi meg a Manual (kézikönyv) oldalain, majd ezeket megjeleníti. Egyenértékű a man -k parancssal. (Lásd még a whatis parancsot is.)

---

**ar** [-]művelet [argumentumok] [pozíciónev] archív [fájlok]

Egy archív fájlt dolgoz fel, amely általában egy tárgykódkönyvtár. A parancssal tetszőleges bináris fájlok töltethetők egy könyvtárba, vagy fájlok gyűjthetők ki onnan. Csak egy művelet adható meg, de a paraméterek száma nem korlátozott. (A magyar „könyvtár” szót két, az angol szaknyelvben szereplő kifejezésre is használjuk: a lemezek „tartalomjegyzékére” utaló directory és a tárgykódokat tartalmazó bináris könyvtár, library fordításaként. Ez zavaró lehet – a lektor.)

A rendelkezésre álló műveletek:

- d törli a fájlokat az archívóból
- m a fájlokat áthelyezi az archívba (a további argumentumokban megadott pozíciótól)
- p az archívban lévő fájlokat listázza
- q a megadott fájlokat az archív végéhez fűzi
- r az archívban a megadott fájlokat az új fájlokkal helyettesíti
- t az archív tartalmát listázza (a v argumentummal részletes lista kérhető)
- x az összes vagy a megadott fájlok kigyűjtése az archívóból

A rendelkezésre álló argumentumok:

- a a fájlokat az archívban a pozíciónev után helyezi el  
(az r és az m műveletnél használható)
- b a fájlokat az archívban a pozíciónev előtt helyezi el  
(az r és az m műveletnél használható)
- c létrehozza az archívot
- i lásd a b argumentumot
- o megtartja az eredeti fájl dátumát és eredetét
- s az archív fájltáblázatát újra létrehozza
- u csak a módosított fájlokkal végi el a helyettesítést (az r műveletnél adható meg)
- v minden műveletnél részletes listát ad

---

**at** [kapcsolók] időpont

Parancsok hajt végre adott időpontban. A parancsok a szabványos bemeneti cszközön adhatók meg, lezárásuk az EOF (fájl vége, <Ctrl-d>) jellel történik. Az f kapcsoló segtségével a bemenet szkript is lehet. A végrehajtásra azt a shellt használja a rendszer, amelyiket a parancs kiadására használjuk (tehát nem az alapértelmezett bejelentkezési shell).

A **q** kapcsolóval egyedi munkák rendelhetők a különböző várólistákhoz (a-z, A-Z), ahol a hátról álló betűk kisebb prioritást jelentenek.

Az időpont numerikus formában (ÓÓ:PP, ÓÓ:PP) vagy egy kulcsszóval, például noon (dél) teatime (16:00) vagy midnight (éjfél) adható meg. A pontos időpont helyett az eddig eltelő idő is megadható, például now + 3 hour (3 óra múlva). A használható egységek a következők: perc, óra, nap, hét, hónap és év. Ha a munkát adott napon kell végrehajtani, akkor a hónap (Jan, Feb, Mar, ...) és az év (95, 96, ...) is megadásra kerül.

#### *A rendelkezésre álló kapcsolók*

- b egyenértékű a batch parancsal
- d a megadott munkákat törli a várólistáról (atrm)
- f fájl a fájlból megadott parancsokat hajtja végre
- l az aktuális felhasználó munkáit sorolja fel (atq)
- m a felhasználónak e-mail üzenetet küld a parancsok végrehajtása után
- q Q a Q-ban megadott várólistához (a-z, A-Z) rendeli a munkát
- V a verziósztámot jeleníti meg

#### **atq [kapcsolók]**

A felhasználó at parancssal végrehajtandó munkáit sorolja fel.

#### *A rendelkezésre álló kapcsolók:*

- q Q csak a Q-ban megadott várólista munkáit sorolja fel
- V a parancs verziósztámát jeleníti meg
- v a végrehajtott, de még nem törölt munkákat sorolja fel

#### **atrm [kapcsolók] munkák**

Eltávolítja a megadott munkákat. Amunkákat az at vagy az atq parancssal megjelenített azonosítójuk írja le.

#### *A rendelkezésre álló kapcsolók:*

- V a verziósztámot jeleníti meg

#### **awk [kapcsolók] [program] [-v vált=érték ...] [fájlok]**

Az awk olyan egyszerű értelmező, amely a grep és a sed jellemzőit egyesíti. Saját, C-hez hasonló nyelvvel rendelkezik. A parancs különösen jól használható ASCII fájlok kiértékelésében, és a rendszерadminisztrátorok számára ilyenek elkeszítésében.

#### *A rendelkezésre álló kapcsolók*

- f fájl a programot a parancssor helyett a megadott fájlból olvassa
- F c a mezők elválasztó karakterét c-re állítja be
- v vált=érték a vált változóhoz a megadott értéket rendeli

#### **basename útnév [kiterjesztés]**

Elhagyja az útvonalat, és ha megadott kiterjesztésre végződik az útnév, akkor a kiterjesztést is, majd a megmaradó fájlnéveket a szabványos kimenetre küldi. Főleg szkriptekben használatos.

---

**bash** [kapcsolók] [argumentumok]

A Bourne és a Korn shellhez hasonló parancsértelemző.

---

**batch** [kapcsolók] [idő]

Hasonló az a t parancshoz. A megadott parancsokat azonban csak a rendszer alacsony leterhelésénél hajtja végre. (Lásd még az at parancsot is.)

*A rendelkezésre álló kapcsolók:*

- f fájl a fájlból megadott parancsokat hajtja végre
  - m a felhasználónak e-mail üzenet küld a parancsok végrehajtása után
  - q Q a Q-ban megadott várolistához (a-z, A-Z) rendeli a munkát
  - V a verziószámot jeleníti meg
- 

**bc** [kapcsolók] [fájlok]

Interaktív, számításokat végrehajtó program, zsebszámológép. A bc saját nyelvet használ, amely lehetővé teszi például új függvények definiálását.

*A rendelkezésre álló kapcsolók:*

- l a matematikai könyvtár függvényeit teszi elérhetővé
- s POSIX kompatibilis működést tesz lehetővé
- w POSIX típusú üzeneteket jelenít meg

Példa:

```
nicetry:~$ bc
bc 1.03 (Nov 2, 1994)
Copyright (C) 1991, 1992, 1993, 1994 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type 'warranty'.
a=5
b=3
a*b
15
quit
nicetry:~$
```

---

**cal** [kapcsolók] [hónap] év

Az aktuális hónapról, illetve a megadott hónapról vagy évről jelenít meg naptárt. Az évet hosszú formátumban (például 1996), míg a hónapot számként (1-12) kell megadni.

*A rendelkezésre álló kapcsolók:*

- j Julianus naptárt jelenít meg (folyamatosan számozott napok)
- y az aktuális év naptárát jeleníti meg

**cat [kapcsolók] [fájlok]**

Több fájlt beolvas, és ezeket a szabványos kimeneti eszközre küldi. Ha nincsenek megadott fájlok, akkor a parancs a szabványos bemeneti eszközt olvassa. A kimenetet a > használatával gyakran átirányítják.

A rendelkezésre álló kapcsolók:

- b folyamatosan megszámozza a nem üres sorokat
- e a -v kapcsolóval használható: a sor vége (EOL) jelként „\$” karaktert ad ki
- n az összes sort folyamatosan megszámozza
- s az összefüggő üres sorokat egygyel helyettesíti
- u pufferelés nélküli kimenet
- v a vezérlő és az egyéb nem nyomtatható karaktereket is kiküldi
- t a -v kapcsolóval használható: a tabulátor helyett „^I”, a lapdobás helyett „^L” karaktert küld ki

Példa:

```
nicetry:~$ cat >file.txt
Ez van a fajlban.
<Ctrl-d>
nicetry:~$ cat file.txt
Ez van a fajlban.
nicetry:~$
```

**cc [kapcsolók] fájlok**

C fordító (lásd gcc).

**cd [könyvtár]**

Az aktuális könyvtár beállítására szolgál, általában a héj tartalmazza. Ha a könyvtár nincs megadva, akkor az aktuális felhasználó home könyvtára az alapértelmezett. A „-” megadásával az előző lesz az aktuális könyvtár.

**chgrp [kapcsolók] csoportfájlok**

A fájlok csoporttagságát módosítja. Ezt a parancsot a rendszer adminisztrátora vagy a megadott fájlok tulajdonosa hajthatja végre. A csoport numerikus csoportazonosítóként vagy a csoport nevével adható meg.

A rendelkezésre álló kapcsolók:

- c azon fájlok nevét jeleníti meg, amelyek csoporttagsága megváltozott
- f tiltja a hibaüzenetek megjelenítését
- R a fájlok csoporttagságát az alkönyvtárakban is módosítja (rekurzív módon)
- v a módosításokat részletes listában közli

**chmod [kapcsolók] jogosultságok fájlok**

A megadott fájlok jogosultságait módosítja. Ezt a parancsot a rendszer adminisztrátora vagy a megadott fájlok tulajdonosa hajthatja végre. A jogosultságok numerikus (oktális) formában vagy parancssorozattal adhatók meg. A parancssorozatban a következők szerepelhetnek: a tulajdonos megnevezése (u), csoport (g) vagy egyéb (o); a felvétel (+), az eltávolítás (-) vagy a beállítás (=) parancs; az olvasási (r), az írási (w) vagy a végrehajtási (x) jogosultság; különleges jelzőket, felhasználói azonosító beállítása (set user ID, s) és sticky (t), beállító vagy törlő parancs.

*A rendelkezésre álló kapcsolók:*

- c azon fájlok nevét jeleníti meg, amelyek jogosultsága megváltozott
- f tiltja a hibaüzenetek megjelenítését
- R a fájlok jogosultságait az alkönyvtárakban is módosítja (rekurzív módon)
- v a módosításokat részletes listában közli

*Példák:*

chmod u+x fájl

végrehajtási jogosultságot ad a fájl tulajdonosának

chmod go-rwx fájlok

visszavonja az olvasási és a végrehajtási jogosultságot a csoport- és az egyéb felhasználóktól

chmod g+s fájl

a megadott fájl csoportazonosító beállítása (set group ID) jelzőjét állítja be

chmod a=r fájl

a fájl jogosultságait csak olvashatóra állítja mindenkinél

chmod 644 fájl

olvasási és frási jogosultságot biztosít a tulajdonos, míg olvasási jogosultságot minden egyéb felhasználó számára

**chown [kapcsolók] tulajdonos[.csoport] fájlok**

A megadott fájlok tulajdonosát, és ha megadják, csoportját módosítja. A tulajdonos és a csoport numerikus azonosítóval vagy a nevével adható meg.

*A rendelkezésre álló kapcsolók:*

- c azon fájlok nevét jeleníti meg, amelyek tulajdonosa megváltozott
- f tiltja a hibaüzenetek megjelenítését
- R a fájlok tulajdonosát az alkönyvtárakban is módosítja (rekurzív módon)
- v a módosításokat részletes listában közli

**cksum [fájlok]**

A megadott fájlokra kiszámítja a CRC ellenőrző összegeket, és ezeket a fájlmérettel és a fájlnévvel együtt megjeleníti

**clear**

Törli a képernyőt

**cmp [kapcsolók] fáj11 [fáj12]**

Két fájl tartalmát bájtonként összehasonlíta. Ha a két fájl azonos, a visszatérési érték 0, ellenkező esetben 1. Ha fájlnévként „..” a megadott érték, akkor a parancs a szabványos bemeneti eszközölt. Ugyanez történik akkor is, ha a fáj12 nincs megadva.

*A rendelkezésre álló kapcsolók:*

- c megjeleníti az eltérő karaktereket
- l az eltérő bájtok oktális értékét és eltolását jeleníti meg
- s a képernyón semmit sem jelenít meg

**comp [kapcsolók] fáj11 fáj12**

Két, soronként rendezett fájt hasonlít össze. Alapértelmezésben az eredményt három oszlopban adja meg: az első oszlop csak a fáj11-ben található sorokat tartalmazza, a második oszlop az összes olyan sort tartalmazza, amely kizárolag a fáj12-ben fordul elő, végül a harmadik oszlop a közös sorokat jeleníti meg.

*A rendelkezésre álló kapcsolók*

- 1 tiltja az első oszlop megjelenítését
- 2 tiltja a második oszlop megjelenítését
- 3 tiltja a harmadik oszlop megjelenítését

**compress [kapcsolók] [fájlok]**

A Lempel-Ziv módszerrel tömöríti a megadott fájlokat. A tömörítést a fájlnévhez fűzött „Z” jelzi. Az összes többi fájlattribútum változatlan marad.

*A rendelkezésre álló kapcsolók*

- b n a kódolásra használható bitek számát n-re korlátozza
- c az eredményeket a szabványos kimeneti eszközre küldi, a fájlokat nem módosítja
- f végrehajtja a tömörítést, a már létező célfájnlá nem kér jóváhagyást
- r a könyvtárakban lévő fájlokat is tömöríti (rekurzív módon)
- v részletes állapotjelentést közöl
- V a program verziószámát jeleníti meg

**cp [kapcsolók] fáj11 fáj12****cp [kapcsolók] fájlok könyvtár**

A fáj11-et a fáj12-be, illetve a megadott fájlokat a könyvtárba másolja. Ha a célfájl (fáj12) már létezik, azt felülírja (az -i kapcsolóval jóváhagyás kérhető).

*A rendelkezésre álló kapcsolók*

- a a -d, a -p és a -r kombinációja
- b a felülírásra kerülő fájlok rövidítési módot készít
- d a másolás során a szimbolikus és fix csatolásokat megtartja
- f a létező fájlokat a másolásnál felülírja
- i létező fájl felülírása előtt jóváhagyást kér
- l a fájlok másolása helyett fix csatolást hoz létre

- P a fájlokat a könyvtárstruktúrába másolja (a nem létező könyvtárat létrehozza)
- p a fájlok jogosultságait és az utolsó módosítások idejét is másolja
- r alkönyvtákat és tartalmukat másolja (rekurzív módon)
- R lásd -r
- s a fájlok másolása helyett szimbolikus csatolást hoz létre
- S kiterjesztés a biztonsági másolatok kiterjesztését a megadottra módosítja
- u az azonos néven már létező és későbbi dátumú fájlokat nem írja felül
- v másoláskor egyenként megjeleníti a fájlok nevét
- x tetszőleges fájlrendszerben a forrásfájltól eltérő könyvtárat figyelmen kívül hagyja
- V {számoszt, létező, egyszerű} a verziókezelés módját határozza meg:  
számoszt minden létrehoz számoszt biztonsági másolatot  
létező számoszt biztonsági másolatot csak azoknál a fájloknál hoz létre, amelyek már rendelkeznek ilyennel, az összes többi esetben egyszerű biztonsági másolatot készít  
egyszerű minden létrehoz egyszerű biztonsági másolatot

---

### cpio kapcsolók [argumentumok]

Fájlokat másol archívumba, archívum tartalmát jeleníti meg, vagy fájlokat bont ki archívumból. Az archívum lehet mágnesszalagon, merevlemezén vagy hajlékonylemezen. A cpio parancsnak három, kapcsolókkal kiválasztható üzemmódja van: -i (copy in = kibontás), -o (copy out = tömörítés) és -p (copy pass = könyvtárak helyi másolása). A cpio megfelelően együttműködik a find parancsal.

#### A rendelkezésre álló kapcsolók

- 0 az új sor karakterre végződő fájlnevek helyett nullára végződőket fogad el (tömörítés és másolás könyvtárakból üzemmód)
- a az olvasott fájlok hozzáfordulási idejét úgy rögzíti, hogy az olvasás a fájl dátumából ne legyen megállapítható
- A létező archívumot bővíti fájlokkal (az -O és az -F kapcsolóval használható)
- b a kibontás során felcsereli a szavakat és a félszavakat
- B a bemeneti/kimeneti puffer méretét 512-ről 5120 bájtra növeli
- c a fájlok fejrézében a régi (hordozható) ASCII formátumot használja
- C n a bemeneti/kimeneti puffer méretét *n* bájttra állítja be
- d a kibontás során automatikusan létrehozza a szükséges alkönyvtárat
- E fájl azokat a fájlokat bontja ki, amelyek neve szerepel a fájlból
- f csak azokat a fájlokat másolja, amelyek neve nem felel meg a megadott keresési feltételek
- F fájl a szabványos kimeneti eszköz helyett a megadott fájlt használja archívumként. A fájl egy gazdaszámítógép nevét is tartalmazhatja, fgy az archív fájl például egy távoli gép mágnesszalagos egységére is írható (-F zeus : /dev/tape).
- H formátum a fejrész adatait a megadott formátumban olvassa/írja
  - bin régi bináris formátum
  - ode régi hordozható formátum (POSIX.1)
  - newc új hordozható formátum (SVR4)
  - cre új SVR4 formátum CRC ellenőrző összeggel
  - tar régi tar-kompatibilis formátum
  - ustar POSIX.1-kompatibilis tar-formátum
  - hpbin régi HP UNIX bináris formátum
  - hpode hordozható HP UNIX formátum
  - i a cpio parancs kibontás üzemmódját állítja be

- I fájl a szabványos bemeneti eszköz helyett a megadott fájlt használja. Például egy távoli gép mágnesszalagos egységén lévő archívum eléréséhez gazdanév is megadható (zeus : /dev/tape).
- L megkülbözteti a szimbolikus csatolásokat, azaz nem a csatolásokat, hanem az általuk hivatkozott fájlokat másolja
- m új fájl létrehozásakor megtarja az eredeti módosítási dátumát
- M üzenet több adathordozós archíválást tesz lehetővé. Ha egy tárolóeszköz betelik, az üzenet jelenik meg a képernyón. Az üzeneten belül a %d változó használható az adathordozó aktuális számának megjelenítésére.
- n a könyvtártartalom megjelenítésekor az UID és a GID numerikus értékként jelenik meg
- o a cpio parancs tömörítés üzemmódját állítja be
- O fájl a szabványos kimeneti eszköz helyett a megadott fájlt használja. Például egy távoli gép mágnesszalagos egységén lévő archívum eléréséhez gazdanév is megadható (zeus : /dev/tape).
- p a cpio parancs könyvtárak helyi másolása üzemmódját állítja be
- r [felhasználó][.][csoport] a tömörítés és a könyvtárak helyi másolása üzemmódban módosítja a fájl tulajdonosát (csak a rendszer adminisztrátora használhatja)
- s a kibontás üzemmódban bájtokat kezel
- S a kibontás üzemmódban félszavakat kezel
- t lista egy archívum tartalmáról listát jelenít meg
- u az azonos nevű és korábbi dátumú fájlok felülírását engedélyezi
- v fájnevek listáját jeleníti meg. A részletes lista a -t kapcsolóval együtt érhető el.
- V a feldolgozott fájlokra egy pontot (,,") jeleníti meg

Példák:

```
find. -name „*.txt” -print | cpio -ocv > /dev/tape
      a mágnesszalagos egységre elkészíti az összes „txt” kiterjesztésű fájl biztonsági másolatát
cpio -icdv < /dev/tape
      a mágnesszalagos egységen lévő összes fájlt kibontja a merevlemezre
find. -print | cpio -pdv /tmp
      az aktuális könyvtárban lévő összes fájlt a /tmp könyvtárba másolja
```

```
crontab [-u felhasználó] file
crontab [-u felhasználó] műveletek
```

Egy felhasználói crontab fájlon hajt végre műveleteket: cseréli, szerkeszti, listázza vagy törli. Az adminisztrátor az -u kapcsolóval tetszőleges felhasználó crontab fájlját kezelheti.

A rendelkezésre álló kapcsolók:

- e a crontab fájlt az alapértelmezett szerkesztőprogrammal (EDITOR környezeti változó) szerkeszti
- l egy felhasználó crontab fájlt listáz
- r törli a crontab fájlt

```
csh [kapcsolók] [argumentumok]
```

A C nyelvi szintaxison alapuló parancsértelmező,

**csplit [kapcsolók] fájl [kifejezés]**

A megadott fájlt több kisebb fájlra bontja, és megjeleníti az így létrehozott fájlok méretét. Ha fájlnévként „-” a megadott érték, akkor a parancs a szabványos bemeneti eszközt olvassa. A felosztás helye a kifejezés segítségével adható meg a következő formában:

szám a sorok számát adja meg, amely után új kimeneti fájlt kell létrehozni  
 /regkif[eltolás] reguláris kifejezés, amely a felosztási helyeket definiálja, emellett pozitív (+)

vagy negatív (-) eltolási érték is megadható

%regkif%[eltolás] hasonló a fenti kifejezéshez, de ebben az esetben a megadott szakasz nem kerül a felosztott fájlba, hanem kimarad

{ismétlődés} annak a kifejezésnek az ismételt alkalmazását jelenti, amelyhez hozzáfűzik.

Ha szám helyett a csillag karakter (\*) írjuk be, akkor a kifejezés a bemeneti fájl végének eléréséig ismétlődik.

*A rendelkezésre álló kapcsolók:*

- f kiterjesztési létrehozott kimeneti fájlok kiterjesztését adja meg
- b kiterjesztés a létrehozott kimeneti fájlok kiterjesztését módosítja. A kiterjesztés formátumát a formázó parancsok határozzák meg: a printf. %d a kimeneti fájl számát decimális formátumra állítja be, míg a %x hexadecimális számábrázolást eredményez.
- k a már létrehozott fájlok a parancs megszakítása esetében is megmaradnak
- n n a kimeneti fájlok nevében lévő növekvő számok hossza (az alapérték 2)
- q tiltja a képernyőn való megjelenítést
- s lásd -q
- z tiltja a 0 hosszúságú fájlok létrehozását

*Példák:*

```
csplit -k linux.txt '%cut%' {30}
       a linux.txt fájlt a „cut” helyen legfeljebb 30 kimeneti fájlra bontja
csplit -k list.txt 10 {100}
       a list.txt fájlt legfeljebb 100, egyenként 10 soros fájlra bontja
```

**ctags [kapcsolók] fájlok**

Beolvassa a megadott C, Fortran, Pascal, LaTeX vagy Lisp forrásfájlokat, és előállítja a benne megadott függvények és makrók listáját. A lista a vi vagy az emacs szerkesztővel dolgozható fel. Az aktuális könyvtárban tags néven létrehozza a kulcsszólistát.

*A rendelkezésre álló kapcsolók:*

- a a talált neveket létező listához fűzi hozzá
- B keresési mintát hoz létre a vi szerkesztőben való visszafelé kereséshez
- C a C++ üzemmódot állítja be, amelyben a .c és .h fájlok kezelése C++ üzemmódban történik
- d előfeldolgozó definíciók bejegyzéseit is előállítja
- f fájl a neveket a megadott fájlba írja. Ha a -f nincs megadva, akkor a tags fájlt használja.
- F keresési mintát hoz létre a vi szerkesztőben való előre kereséshez (alapértelmezett)
- H súgószöveget jelenít meg
- i fájl a megadott fájlból folytatja a keresést
- o fájl a kimeneti fájl nevét módosítja
- S a behúzásokat figyelmen kívül hagyja

- t a típusdefiníciókat is figyelembe veszi
- T a típusdefiníciók, struktúrák, számlálások és C++ tagfüggvények listáját is elkészíti
- u a lista frissítése
- v indexfájlt készít vgrind formátumban, és azt a szabványos kimeneti eszközre küldi
- V a verziószámot jeleníti meg
- w ismétlődő bejegyzésekkel nem ad figyelmeztetést
- x kereszthivatkozási listát készít cxref formátumban, és azt a szabványos kimeneti eszközre küldi

### **cut kapcsolók [fájlok]**

-Mezők vagy oszlopok sorozatát veszi ki a bemeneti fájl egy sorából. A -b, a -c vagy az -f kapcsolóval valamelyike használható. Ezek visszövel elválasztott számokat vagy kötőjellel definiált mezőket tartalmazó listát feltételeznek.

#### A rendelkezésre álló kapcsolók:

- b lista a listában definiált pozícionál lévő karaktert jelöli ki
- c lista a listában megadott oszlopokat jelöli ki
- d c a -f kapcsolóval együtt a mezőhatároló karakter (c) megadására szolgál
- f lista a tabulátorral vagy más határoló karakterrel elválasztott mezőket jelöli ki a listából
- s az credményt a mezőhatárolókat tartalmazó sorokra korlátozza

Példa:

```
cut -d: -f1,3 /etc/passwd
az összes felhasználó bejelentkezési nevét és azonosítóját adja meg
```

```
date [kapcsolók] [+formátum]
date [kapcsolók] [karaktersorozat]
```

Az első alakjában az aktuális dátumot és időpontot adja vissza a megadható formátumban. A második megadással a rendszer adminisztrátora állíthatja be a rendszeridőt.

#### A megadható formátumok:

% %	százalékjel
%n	új sor
%t	tabulátor
%H	óra (00 ... 23)
%I	óra (01 ... 12)
%k	óra (0 ... 23)
%l	óra (1 ... 12)
%M	perc (00 ... 59)
%p	AM vagy PM (napszakjelző)
%r	az időpont 12 órás formátumban (00:pp:mm[AM PM])
%s	az 1970. január 1. 0:00 óta eltelt másodpercek
%S	másodper (00 ... 59)
%T	az időpont 24 órás formátumban
%X	az időpont helyi formátumban
%Z	az időzóna, ha megadott, egyébként üres
%a	a nap nevének helyi rövidítése
%A	a hétfajának helyi rövidítése

%b	a hónap nevének helyi rövidítése (Jan ... Dec)
%B	a hónap helyi neve (Január ... December)
%c	a helyi dátum az időponnal és az időzónával
%d	a hónap napja (01 ... 31)
%D	dátum (hh/mm/éé)
%h	megegyezik a %b formátummal
%j	az évben folyamatosan számozott napok (001 ... 366)
%m	a hónap számmal megadva (01 ... 12)
%U	a hétköznap számmal megadva (00 ... 53), a hétköznap első napja vasárnap
%w	a hétköznap számmal (0 ... 6)
%W	a hétköznap számmal megadva (00 ... 53), a hétköznap első napja hétfő
%x	a dátum helyi formátumban (mm/dd/éé)
%y	az évszám utolsó két jegye (00 ... 99)
%Y	évszám (1995 ... )

A karakterkódok formátuma az időpont beállításához:

DD	a hónap napja
hh	óra
mm	perc
CC	az évszám első két jegye (évszázad)
YY	az évszám utolsó két jegye
ss	másodperc

A rendelkezésre álló kapcsolók:

- d dátum a megadott dátumot küldi ki (amely tartalmazhatja a hónap nevét, az időzónát stb.)
- s dátum a dátum formátumát állítja be (amely tartalmazhatja a hónap nevét, az időzónát stb.)
- u az időzóna helyett az UTC (Universal Coordinated Time) megadást használja

---

**dd** [kapcsolók=érték ...]

A szabványos bemeneti eszközről vagy egy megadott fájlból a szabványos kimeneti eszközre vagy megadott fájlba másol. A leggyakrabban használt kapcsoló az **i f**, amely a bemeneti fájlt adja meg, illetve az **o f**, amely a kimeneti fájlt definíálja. A dd parancs használható például egy rendszermag fájlképének közvetlen lemezre másolásához, vagy indítólemez készítéséhez.

A rendelkezésre álló kapcsolók:

**bs=n** a bemeneti vagy a kimeneti blokkméretet n bajtra állítja be. Lehetőség van az n mértékegységének a megadására is, például 8k, ami 8 kilobájtot jelent

**cbs=n** konvertáláskor a mezőméretet határozza meg

**conv=jelölök** a bemenet konvertálását a következő argumentumok szerint hajtja végre:

ascii konvertálás EBCDIC kódóból ASCII kódba

ebedic konvertálás ASCII kódóból EBCDIC kódba

ibm konvertálás ASCII kódóból IBM EBCDIC kódba

block változó hosszúságú mezőket cbs hosszúságú mezőkbe konvertál, a hiányzó helyeket üres karakterekkel tölti fel

unblock rögzített hosszúságú mezőket (cbs) változó hosszúságú mezőkbe konvertál

lease nagybetűket kisbetűkre konvertál

ucase kisbetűket nagybetűkre konvertál

swap a bemeneti fájl minden két bajtját felcseréli

noerror az olvasási hibákat figyelmen kívül hagyja

**notrunc** nem csonkolja a kimeneti fájlt  
**syncac** ibr méretű bemeneti blokkok üres helyeit zérusokkal tölti fel  
**count=n** csak n blokkot másol  
**if=fájl** a bemeneti fájlt adja meg  
**of=fájl** a kimeneti fájlt adja meg  
**ibr=n** a bemeni puffer méretét adja meg  
**obs=n** a kimeni puffer méretét adja meg  
**skip=n** kihagy n bemeneti blokkot

**df [kapcsolók] [útnévek]**

A fájlrendszer foglalt és szabad blokkjainak számát adja meg. Ha nincs megadva útnév, akkor az aktuális fájlrendszer listáját kapjuk. Ha van megadott útnév, a kapcsolódó fájlrendszerek áttekintését adja. Olyan eszközök (/dev/hda1) közvetlen elérési útneve is megadható, amelyen a fájlrendszer található. Szokásos esetben csak a zérusnál nagyobb adathordozóval rendelkező fájlrendszerek jelennek meg az eredményben.

*A rendelkezésre álló kapcsolók*

- a az összes aktuális fájlrendszt megjeleníti, a zérus méretűket is
- i a blokkinformáció helyett az i-csomópont statisztikát jeleníti meg
- k egy kilobájtos blokkméretet használ (ez az alapértelmezés)
- P POSIX kimeneti formátumot használ
- t típus a kimenetet a megadott típusú fájlrendszerre korlátozza
- x típus a megadott típusú fájlrendszeret figyelmen kívül hagyja

**diff [kapcsolók] fáj1 fáj2**

Két fájlt vagy két könyvtárban lévő fájlokat hasonlíthat össze. Ha a két útnév valamelyikének megadása „-”, akkor ennél a fájlokat a szabványos bemeneti eszkökről veszi. A diff parancs az összes olyan sort kijelzi, amelyik csak az egyik fájlból fordul elő, vagy amelyek eltérőek. A kimenet a patch parancssal fájlok módosítására használható. Fájlok összehasonlítására és összefeszítésére az ediff Emacs Lisp program is használható.

*A rendelkezésre álló kapcsolók:*

- a az összes bemeneti fájlt szövegfájlnak tekinti, és az összehasonlítást soronként hajtja végre
- b az üres karakterek eltérő számát nem tekinti eltérésnek (a sorok végén sem)
- B az üres karaktereket figyelmen kívül hagyja
- c az eredményben az eltérő sorok környezetét is megjeleníti (három sor)
- C n azonos a -c kapcsolóval, de az eltérő sorok környezetében n számú sort jelenít meg
- d a fájlok összehasonlítására jobb, de lassabb algoritmust használ
- D név a két fájlt egyesíti, és megfelelő előfeldolgozó utasítások beszúrásával (#ifdef név) megkülönöztethetővé teszi a változatokat. Ha a név megadása a fordítás alatt történik, akkor a változat a fáj1-be kerül, egyébként a fáj12-be.
- e utasításokat ad ki az ed szerkesztő számára, hogy a fáj12-ből létrehozható legyen a fáj11
- f azonos az -e kapcsolóval, de fordított sorrendben, bár ez nem használható
- h nem használható
- H heurisztikus módszer alkalmazásával gyorsítja a végrehajtást
- i a kisbetű-nagybetű különbséget figyelmen kívül hagyja

- l (csak teljes könyvtárak összehasonlításakor) a kimenet a pr parancsral feldolgozható, így minden fájl új oldalon kezdődhet
- n a kimenetet RCS formátumban állítja elő
- N két könyvtár összehasonlításakor a hiányzó fájlokat létezőnek, de üresnek tekinti
- q a fájlok eltérésének csak a tényét közli
- r (csak teljes könyvtárak összehasonlításakor) az alkönyvtarakat rekurzív módon kezeli, és az összes fájlt összehasonlíti
- s két fájl azonosságát is kijelzi
- S** fájl könyvtárak összehasonlítását adott fájllal kezdi
- t a tabulátorokat szóközökkel helyettesíti
- T az összes kimeneti sor elején szóköz helyett tabulátorot ad ki
- u az eredményt GNU „egyészített” formátumban adja meg
- v a verziószámot jeleníti meg
- w sorok összehasonlításánál a szóközöket és a tabulátorokat figyelmen kívül hagyja
- x minta teljes könyvtárak összehasonlításánál a megadott mintának megfelelő fájlokat és alkönyvtarakat figyelmen kívül hagyja
- y az eredményt könnyebben olvasható kétoszlopos formátumban adja meg

**diff3** [kapcsolók] fáj11 fáj12 fáj13

Három fájlt hasonlít össze soronként.

*A rendelkezésre álló kapcsolók:*

- a soronkénti összehasonlítás, az összes bemeneti fájlt szövegfájlnak tekinti
- A** a fáj12 és fáj13 közötti eltéréseket a fáj11-be tölti, és jelöli ezeket
- e utasításokat ad ki az ed szerkesztő számára, így a fáj12 és fáj13 közötti eltérések a fáj11-ben integrálhatók
- E azonos az -e kapcsolóval, de a kimenet kevésbé részletes
- i a létrehozott ed szkript végén a w és a q parancsot helyezí el
- m a szerkesztőszkriptet a fáj11-re alkalmazza, és megjeleníti azt
- T az összes kimeneti sor elején szóköz helyett tabulátorot ad ki
- v a parancs verziószámat jeleníti meg
- x azonos az -e kapcsolóval, de csak az egymást átfedő eltéréseket jelzi ki
- X** azonos az -E kapcsolóval, de csak az egymást átfedő eltéréseket jelzi ki
- 3 azonos az -e kapcsolóval, de csak az egymást át nem fedő eltéréseket jelzi ki

**dirname** útnév

Teljes elérési útvonalat állít össze (a basename ellentettjét hajta végre). Ha az útvonal végén nincsen fájl, akkor a „.” lesz az eredmény.

**du** [kapcsolók] [fájlok | könyvtárak]

A megadott fájlok vagy könyvtárak méretét adja eredményül.

*A rendelkezésre álló kapcsolók:*

- a** az összes fájl méretét is megadja, nemcsak a könyvtárakét
- b** a fájlméreteket bájiban adja meg
- k** a fájlméreteket kilobájiban adja meg
- l** a csatolt fájlok méretét adja meg, még akkor is, ha ez ismétlődéseket jelent

**-s** az összes fájl és alkönyvtár összesített méretét adja meg  
**-x** az eltérő fájlrendszerök könyvtárait figyelmen kívül hagyja

**echo [-n] [szöveg]**

Ezt a parancsot általában beépítik a héjprogramba. A szöveget a szabványos kimenetre küldi. A **-n** az új sor karaktert nem adja ki.

**ed [kapcsolók] [fájl]**

Nagyon régi szabványos szerkesztő, amelynek a **diff** parancssal való együttműködésen kívül már nincs jelentősége.

**egrep**

Lásd **grep**.

**env [kapcsolók] [változó=érték] [parancs]**

Ha a parancsot paraméterek nélkül használjuk, akkor az összes környezeti változó listáját adja. Emellett a parancs lehetővé teszi parancsok indítását módosított környezetben. A **parancs-sorban** új változók definíálhatók vagy meglévők törlhetők.

A rendelkezésre álló kapcsolók:

- i** az örkölt környezetet figyelmen kívül hagyja
- u név** eltávolítja a megadott környezeti változót

**expr arg1 operátor arg2 [operátor arg3 ... ]**

Kifejezést értékel ki, majd az eredményt a szabványos kimeneti eszközre küldi. A kifejezés lehet numerikus, logikai vagy relációs. Ezt a parancsot általában a héjszriptekben használják.

Aritmetikai operátorok:

**+, -, \*, /, %** (moduláris maradék)

Relációs operátorok:

**=, !=, <, >,**

Logikai operátorok:

**! (vagy), & (és)**

: (Arg2 keresése szokásos kifejezésként az Arg1-ben)

Példák:

**expr7+7/2**

a kiértékelés eredménye 7 (egész aritmetika, balról jobbra)  
**expr\$=,,hello”**

a kiértékelés eredménye 1, ha sa „hello” karaktersorozatot tartalmazza, egyébként 0

**false**

Ez a parancs nem hajt végre műveletet, egyszerűen a *false* (hamis) eredményt adja (nem a 0 eredményt). Lásd még *true*.

**fdformat [-n] eszköz**

Mágneslemez alacsony szintű formázását hajtja végre. A szükséges paraméter a megfelelő eszköz elérési útvonala. Az első lemezegység címzése `/dev/fd0XXX`, a másodiké `/dev/fd1XXX`. A -n kapcsoló tiltja a lemez ellenőrzését.

Eszköz	Szektor	Sáv	Méret	Tárkapacitás (KB)
<code>/dev/fd01200</code>	15	80	$5\frac{1}{4}$	1200
<code>/dev/fd0D720</code>	9	80	$3\frac{1}{2}$	720
<code>/dev/fd0H1440</code>	18	80	$3\frac{1}{2}$	1440
<code>/dev/fd0E2880</code>	36	80	$3\frac{1}{2}$	2880

**fgrep**

Lásd *grep*.

**file [kapcsolók] fájlok**

A megadott fájlok típusát adja eredményül. A fájltípus a bővíthető szabályfájl alapján kerül megállapításra (`/etc/magic`).

A rendelkezésre álló kapcsolók:

- c a szabályfájl ellenőrzése
- f fájl a fájlból felsorolt fájlokat vizsgálja
- m fájl az `/etc/magic` helyett a megadott fájlt használja szabályfájlként
- L a szimbolikus csatolásokat is vizsgálja
- z tömörített fájlok vizsgálatát is engedélyezi

**find útnév feltételek**

A parancs rekurzív módon megkeresi azokat a fájlokat, amelyek az összes megadott feltételt kielégítik. A feltételek listájának kiértékelése balról jobbra történik. A feltétel ellenetétje az elője tett felkiáltójellel (!) állítható be. Két feltétel között VAGY kapcsolat az -o megadással lehetséges. A *find* különösen hasznos egyéb parancsokkal (például *cpio*) együtt használva.

Numerikus feltételek az alábbi három módon adhatók meg:

- +n az n-nél nagyobb érték
- n az n-nel egyenlő érték
- n az n-nél kisebb érték

Rendelkezésre álló beállítások (*mindig igazak*)

- depth egy könyvtárban lévő fájlok feldolgozása előbb történik, mint magáé a könyvtáré
- follow a szimbolikus csatolásokkal jelzett könyvtárakat is vizsgálja

#### *Lehetséges feltételek:*

- amin n az utóbbi n percben használt fájlok
- anewer fájl a megadott fájlnál későbben használt fájlok
- atime n az n nappal ezelőtt utoljára használt fájlok
- ctime n az n nappal ezelőtt utoljára módosított fájlok (akár maguk a fájlok, akár jogosultságai vagy tulajdonosuk)
- fstype típus a fájlrendszerben lévő bizonyos típusú fájlok (pl. ext 2, msdos, proc)
- group csoport adott csoportba tartozó fájlok (név vagy azonosító)
- inum n az n i-csomópontszámmal rendelkező fájlok
- links n az n csatolással rendelkező fájlok
- local a fizikailag a helyi rendszeren tárolt fájlok
- mtime n az n nappal ezelőtt utoljára módosított fájlok (csak maguk a fájlok)
- name minta azon fájlok nevei, amelyek megfelelnek a megadott mintának
- newer fájl azon fájlok, amely utolsó módosítása újabb, mint a megadott fájlé  
(lásd még az `mt -ime` feltételt is)
- nogroup azon fájlok, amelyek csoportja nem létezik az /etc/groups könyvtárban
- nouser azon fájlok, amelyek tulajdonosa nem létezik az /etc/passw könyvtárban
- perm nnn azon fájlok, amelyek jogosultsága megegyezik a megadott oktális értékkel
- size n [c, k] az n blokk, n bájt vagy n kilobájt méretű fájlok
- type ca c típusú fájlok, ahol a c a következő lista valamelyik eleme lehet
- b blokkspecifikus fájl
- c karakterspecifikus fájl
- d könyvtár
- p FIFO vagy elnevezett adatcsatorna
- l szimbolikus csatolás
- f normál fájl
- user felhasználó a megadott felhasználóhoz tartozó fájlok (név vagy azonosító)

#### *Lehetséges műveletek:*

- exec parancs {}; a parancsot minden fájra végrehajtja, és ellenőrzi, hogy a visszatérési kód 0-e.  
A végrehajtás során a {} helyére az aktuális fájl neve kerül.
- ok parancs {}; megegyezik az exec művelettel, de a felhasználónak a parancsokat jóvá kell hagynia („y”)
- print a megtalált fájlokat vagy könyvtárakat megjeleníti
- print formátum megegyezik a -print művelettel, de a kimenet a formátum karakterszorozattal formázható

#### *Példák:*

```
find . -typef-print
az aktuális könyvtárban és ennek alkönyvtáraiiban található normál fájlokat jeleníti meg
find ./usr/include-typef" -execgrep "read" {} \; -print
az /usr/include könyvtárban lévő normál fájlból a „read” karakterszorozatot keresi
```

**finger [kapcsolók] [felhasználó]**

Felhasználóról szolgáltat információt. A felhasználó név, név@gazdanév vagy @gazda-név formában szerepelhet. Az első két esetben a felhasználók neve, az időpontok, a legutóbbi bejelentkezés és egyéb adatokat kapunk. Ha a home könyvtárban létezik a .plan vagy a .project fájl, akkor ez is megjelenítésre kerül. Ha csak egy @gazdanév megadott, akkor az összes jelenleg bejelentkezett felhasználó listáját kapjuk.

*A rendelkezésre álló kapcsolók:*

- l részletes listát jelenít meg (a @gazdanév megadásnál)
- m a megadott felhasználónak pontosan meg kell egyeznie a felhasználónével. E kapcsoló megadása nélkül a megadott név az /etc/passw fájlból tárolt teljes felhasználói névvel is összehasonlításra kerül.
- p a megfelelő felhasználóhoz tartozó .plan és a .project fájl nem jelenik meg
- s rövidített kimeneti formátum

**ftp [kapcsolók] [gazdanév]**

Ez a parancs fájlok átvitelére szolgál az ftp protokollal. A gazdanév megadható névvel vagy IP címvel. Ha nincs megadott gazdanév, a program promptot jelenít, és ekkor ftp parancsok vihetők be. A help befrására a súgó jelenik meg.

*A rendelkezésre álló kapcsolók:*

- d hibakereső üzemmód
- g a fájlnévek megadásánál a helyettesítő karakterek használatát tiltja
- i kikapcsolja a lekérdezéseket (mget, mput)
- n az .netrc fájlból felsorolt automatikus bejelentkezést tiltja
- v az összes ftp kiszolgáló üzenetet jeleníti meg

**gcc [kapcsolók] [fájlok]**

A C, a C++ és az Objective C mellett a GNU fordítóprogrammal más nyelvek (például Ada és Pascal) is használhatók. Ezekről a szolgáltatásokról részletes ismertetés a GNU Info dokumentumokban található.

**grep [kapcsolók] regexp [fájlok]**

A szabványos bemeneti eszközről érkező fájlokban vagy adatokban soronként szokásos kifejezést (regexp) keres, és a megfelelő sorokat a szabványos kimeneti eszközre küldi.

*A rendelkezésre álló kapcsolók:*

- b a megtalált kifejezésnek a bájt pozícióját is megadja
- c csak a sorok számát adja meg, amelyekben a kifejezést megtalálta
- h a fájlnéveket nem adja meg
- i a méret – és a kisbetű/nagybetű – különbségeket figyelen kívül hagyja
- l csak azokat a fájlnéveket adja meg (a sorokat nem), amelyekben a kifejezés megtalálható
- n a megtalált sorok sorszámát adja meg
- s nem jelenít meg hibaüzenetet, ha egy fájl nem található vagy nem nyitható meg
- v azokat a sorokat keresi meg, amelyek nem tartalmazzák a szokásos kifejezést

**groff [kapcsolók] [fájlok]**

A groff az nroff és a troff GNU változata. A parancs a kézikönyvoldalak és egyéb, a megfelelő formátumban lévő dokumentumok formázására szolgál. Az egyéb előfeldolgozók, ilyen például az eqn vagy a tbl, a groff-ba integrált, és kapcsolókkal hívható. Az eredmények ASCII, DVI vagy PostScript formátumban tárolható. A dokumentumok formázásánál fontos megadni a használt makrócsomagot. A kézikönyv oldalánál például a -man kapcsoló adható meg. A formázott fájlt a szabványos kimeneti eszközre küldi a parancs.

*A rendelkezésre álló kapcsolók:*

- a Kimenet egyszerű ASCII formátumban
- e az esq előfeldolgozót aktivizálja
- E tiltja a hibatünetek megjelenítését
- h a súgó szövegét jeleníti meg
- m makró a formázásra különleges makrócsomagot használ
  - man makrók a Manual oldalakhoz
  - ms ms makrócsomag
- p a pic előfeldolgozót aktivizálja
- s a soelim előfeldolgozót aktivizálja
- t a tab előfeldolgozót aktivizálja
- T formátum a kimeneti formátumot (ascii, ps, dvi) adja meg
- v a verziószámot jeleníti meg

*Példa:*

**zeus:/home/uh1> groff -man -Tps ls.1 ls.ps**

az ls.1 Manual oldalt formázza a megfelelő makrócsomaggal, és az eredményt PostScript formátumban az ls.ps fájiba tölti.

**groups [felhasználó]**

Azokat a csoportokat adja meg, amelyekbe a megadott felhasználó tartozik. Paraméter nélküli megadásnál az aktuális felhasználó saját csoportjait listázza, míg paraméterrel a megadott felhasználó csoportjait. A parancs az /etc/passw és az /etc/groups fájlt értékeli ki.

**gzip [kapcsolók] [fájlok]**

A fájlokat tömöríti vagy kibontja az LZ77-es eljárás szerint, és a fájlnévhez a „.gz” kiterjesztést fűzi. Ha nincs megadott fájl (vagy az „.”), akkor a szabványos bemeneti eszközt olvassa, míg az eredményt a szabványos kimeneti eszközre küldi. A gzip a compress parancssal tömörített fájlokat (kiterjesztésük „.Z”) is ki tudja bontani.

*A rendelkezésre álló kapcsolók:*

- a az ASCII szövegfájlokban a sorvégeket megfelelően kezeli (CRLF vagy LF)
- c az eredményt, a bemeneti fájl felülírása nélkül, a szabványos kimeneti eszközre küldi
- d tömörített fájlokat bont ki
- f a már létező fájlokat felülírja
- l tömörített fájloknál megjeleníti a tömörített és a kibontott méretet, a tömörítési arányt és az eredeti fájl nevét

- q tiltja a figyelmeztető üzenetek megjelenítését
- r rekurzív módon az alkönyvtárakat is feldolgozza
- S kiterjesztés tömörített fájlok nál a fájlkiterjesztést módosítja
- v minden fájnál megjeleníti a nevet és a tömörítési arányt
- # a tömörítés minőségét állítja be az 1 (rossz) és a 9 (jó) érték között; az alapértelmezett érték 6

**head [kapcsolók] [fájlok]**

A megadott (szöveg)fájlok első 10 sorát jeleníti meg. Több fájl megadásakor az eredményben a fájlnév megelőzi a fájl tartalmát.

*A rendelkezésre álló kapcsolók*

- # a megjelenített sorok számát a megadott értékre módosítja
- c n[blk|m] az első n számú bájtot jeleníti meg, ahol a megadott érték lehet bájt (b), kilobájt (k) vagy megabájt (m)
- q a fájlnév nem szerepel az eredményben

**hostname [név]**

Paraméter nélküli hívásakor a gazdagép névét adja meg; egyébként beállítja a megadott nevet. A gazdanév megadása általában a rendszer indításakor történik és superuser jogosítványokat igényel.

**id [kapcsolók]**

Az aktuális felhasználó valós és az érvényben lévő felhasználói azonosítóját (UID) és összes csoportját (GID) adja meg.

*A rendelkezésre álló kapcsolók:*

- g csak a GID azonosítót jeleníti meg
- G csak a felhasználó további csoportjait jeleníti meg
- n a GID vagy az UID azonosítót névként jeleníti meg (csak a -g, -u és -G kapcsolóval együtt használható)
- r az érvényben lévő helyett a valós GID azonosítót jeleníti meg (csak a -g, -u és -G kapcsolóval együtt használható)
- u csak az UID azonosítót jeleníti meg

**join [kapcsolók] fáj1 fáj2**

Két alfabetikusan rendezett ASCII fájlt egyesít egy kulcs szerint. Az azonos kulccsal rendelkező sorok egyesítésre kerülnek, és ezeket a szabványos kimeneti eszközre küldi a parancs. A kulcsokat a tabulátor vagy a szóköz karakterrel kell elválasztani. Ha nincs más megadott kapcsoló, akkor kulcsként az első oszlopot használja.

*A rendelkezésre álló kapcsolók*

- a [n] üres sort helyez el az eredményben, ha a fájl n (1 vagy 2) egy sora nem tartalmaz a másik fájlból lévő egyező kulcsot
- e karaktersorozat az üres kimeneti mezőket a megadott karaktersorozattal helyettesíti
- j n m az n fájl (1 vagy 2) m oszlopát használja kulcsként
- o n.m csak a fájln m oszlopát jeleníti meg
- t z mezőelválasztóként (bemeneti/kimeneti) a z karaktert használja

**kill [kapcsolók] processzek**

Ezt a parancsot általában beépítik a héjprogramba: jelzést küld egy vagy több processznek. További kapcsolók nélkül a küldött jelzés a TERM, amely leállítja a processzi. Csak a rendszer adminisztrátora küldhet nem saját processzeknek jelzést. A processzek azonosítása a processzszámmal (PID) történik. A jelzések numerikus vagy szimbolikus formában adhatók meg.

**A rendelkezésre álló kapcsolók:**

- l az összes jelzésnevet jeleníti meg
- jelzés adott jelzést küld a megadott processzeknek. Hasznos jelzések a következők:

Szám	Név	Leírás
1	SIGHUP	Egy terminálkapcsolat megszakítását állítja elő Számos démon esetében a konfigurációs fájlok újböli olvasására szolgál
2	SIGINT	Egyenértékű a <Ctrl-c> kiadásával
3	SIGQUIT	Leállítja a processzt, és rendszerkritikus indít el
9	SIGKILL	Leállítja a processzt. Ez a jelzés nem állítható le
15	SIGTERM	Leállítja a processzt (alapértelmezés)
10	SIGUSR1	Felhasználótól függő jelzés, amelynek jelentése alkalmazásonként eltérő
12	SIGUSR2	Lásd a SIGUSR1 jelzést

**ksh [kapcsolók] [argumentumok]**

Lásd a bash parancsot.

**last [kapcsolók] [attribútumok]**

A bejelentkezési statisztikáról (/etc/wtmp) szolgáltat információt. Ha nincs további megadott argumentum, akkor az összes bejelentkezési, kijelentkezési, lezárási és újraindítási műveletet adja meg. Ez a lista a felhasználó vagy az esemény nevét, a bejelentkezési terminált, a bejelentkezési gazdagépet és az időpontot tartalmazza. A választék adott bejegyzésekre korlátozható keresési argumentumok megadásával (név, bejelentkezési terminál).

**A rendelkezésre álló kapcsolók**

- # a kimenetet adott számú sorra korlátozza
- f fájl adatbázisként az /etc/wtmp helyett a megadott fájlt használja
- t terminál csak a megadott terminalról végrehajtott bejelentkezéseket listázza
- h számítógép csak a megadott számítógépről végrehajtott bejelentkezéseket listázza

Példa:

```
hermes:/root# last uhl
uh1      ttys4      mobby      Sun Jan 29 17:24      still logged in
uh1      ttys2      tonnie     Sun Jan 29 16:32      16:47 (00:15)
wtmp begins Sun Jan 29 15:18
hermes:/root#
```

**ld [kapcsolók] tárgy\_fájlok**

A szerkesztő-összekapcsoló program egyedi tárgy\_fájlokat végrehajtható programmá szerkeszt. Ritkán hívják közvetlenül, általában a C szerkesztőprogram vagy a make parancs automatikusan hívja a szerkesztő-összekapcsoló programot.

**ldd [kapcsolók] [programok]**

A programhoz szükséges dinamikus könyvtárakat listázza.

*A rendelkezésre álló kapcsolók:*

- d áthelyezést hajt végre, és a hiányzó függvényeket listázza (csak ELF formátum)
- r az adatok és a programkód áthelyezését hajtja végre, és a hiányzó objektumokat listázza (csak ELF formátum)
- v a parancs verziószámát jeleníti meg
- V a dinamikus szerkesztő-összekapcsoló program (ld.so) verziószámát jeleníti meg

**lex [kapcsolók] [fájlok]**

Az átolvasó generátor a nyelvtani átolvasóból mint bemeneti fájlból létrehozza a lex.yy.c kimeneti fájlt.

**ln [kapcsolók] útnév cél\_útnév**

Csatolást hoz létre. Kapcsolók nélküli hívásakor szoros csatolást hoz létre egy fájlhoz. Az -s kapcsolóval szimbolikus csatolás jön létre, amely egy könyvtári is mutatható. Ha a cél útnév már létezik és fájl, hibaüzenetet kapunk. Csak az -f kapcsolóval kerül felülírásra ez a fájl. Ha a cél\_útnév könyvtár, akkor a csatolás ebben a könyvtárban jön létre.

*A rendelkezésre álló kapcsolók:*

- f a létező fájlokat figyelmezhetés nélkül felülírja
- s szimbolikus csatolások létrehozása

**lpc [parancs [argumentum]]**

A nyomtató munkakezelőjének vezérlésére szolgál. Lehetővérteszi egyedi nyomtatók és ezek nyomtatási várólistáinak aktív vagy inaktív állapotba hozatalát, nyomtatási munkák áthelyezését a várólistákon és állapotadatok megjelenítését. Az lpc argumentum nélküli hívásával interaktív parancsüzemmódba léphetünk. A parancsok az lpc hívásakor is átadhatók.

*A rendelkezésre álló parancsok:*

- help a rendelkezésre álló parancsok listáját jeleníti meg
- abort {all | nyomtató} lezárja az aktív munkakezelőket, és tiltja a megadott nyomtatót(ka)t
- clean {all | nyomtató} a megadott nyomtató várólistájáról eltávolítja az összes befejezetlen fájlt
- disable {all | nyomtató} tiltja a megadott nyomtatót(ka)t
- down {all | nyomtató} üzenet kikapcsolja a megadott várólistákat, tiltja a megadott nyomtatót(ka)t, és a megadott üzenetet írja a nyomtató állapotfájljába. Az üzenet az lpc hívásakor jelenik meg.

**enable [all | nyomtató]** engedélyezi a megadott *nyomtató(k)* várólistáját, és lehetővé teszi új munkák felvételét  
**exit, quit** kilépés az *lpc* programból  
**restart [all | nyomtató]** megkíséri újraindítani a *nyomtató-démon(oka)*  
**start [all | nyomtató]** aktív állapotba hozza a *nyomtató(ka)t*, és a megadott *nyomtató(k)ra* elindítja a *nyomtató-démon*  
**status [all | nyomtató]** az aktív *nyomtató-démon(ok)* és várólisták állapotadatait adja ki  
**stop [all | nyomtató]** leállítja a *nyomtató-démon* aktuális munkáját, és tiltja a megadott *nyomtatót*  
**topq nyomtató [munka#][felhasználó]** a megadott munkát a várólista elején helyezi el  
**up [all | nyomtató]** aktivizálja a várólistákat, és elindítja a *nyomtató-démont*

---

### **lpq [kapcsolók] [munka#] [felhasználó]**

Nyomtatási várólisták aktuális állapotáról szolgáltat információt.

A rendelkezésre álló kapcsolók:

- I részletes jelentést készít minden egyes munkáról
  - P név nyomtatási várólistát jelöli ki
- 

### **lpr [kapcsolók] [fájlok]**

A fájlokat a nyomtatási várólistába küldi. Az adatok a szabványos bemeneti eszközön keresztül is kinyomtathatók. Kapsolók nélküli hívásnál az *lp* várólistába küldi a fájlokat.

A rendelkezésre álló kapcsolók:

- # n a megadott dokumentumkból n számú példányt készít
  - C szöveg a munka adatait nyomtatja a címmel
  - h a fejléc adatok nyomtatását tiltja
  - J munka a címmel kinyomtatja a *munka* nevét
  - m a felhasználónak a nyomtatási munka befejezésekor levelet küld
  - P név a megadott nyomtatási várólistát jelöli ki
  - r kinyomtatása után törli a fájlt (az -s kapsolóval együtt használható)
  - s a fájl nem a munkakezelőbe, hanem csatolásra kerül, így a nyomtatófájl nem törölhető a nyomtatás alatt
  - U felhasználó a címmel kinyomtatja a *felhasználó* nevét
- 

### **lprm [kapcsolók] [munka#] [felhasználó]**

Bejegyzéseket töröl a nyomtatási várólistából. A megadásnál *munkaszámok* vagy *felhasználói* nevek használhatók. Ha nincs megadott argumentum, akkor az aktív munkát törli.

A rendelkezésre álló kapcsolók:

- az összes tételet törli a várólistából
  - P név a megadott nyomtatási várólistát jelöli ki
- 

### **ls [kapcsolók] [fájlok]**

Könyvtárak tartalmát jelenti meg, vagy megadott fájlokat listáz. Ha nincsenek megadott fájlok, akkor az aktuális könyvtár tartalmát listázza ki. Ha a fájlok argumentum megadott (esetleg helyettesítő karakterekkel), akkor csak az ezzel egyező fájlokat listázza.

*A rendelkezésre álló kapcsolók:*

- a az összes fájl t listázza, beleértve a ponttal kezdődőket is
- A megegyezik az -a kapcsolóval, de a „.” és a „..” tételeket kihagyja
- B a tilde (-) karakterre végződő biztonsági másolati fájlokat kihagyja
- b a nem nyomtatható karaktereket oktális számkként jeleníti meg
- c a fájlokat állapotuk legutóbbi módosítási ideje szerint rendezzi
- C csak a fájlnéveket jeleníti meg, de több oszlopban (ez az alapértelmezés)
- d könyvtárnév megadásakor csak magát a könyvtárat listázza, nem a tartalmát
- f nem rendezett eredmény
- F minden fájlnévhez egy, a fájl típusát jelző karaktert fűz (normál fájl, könyvtár, programfájl, csatolás stb.)
- G tiltja a hosszú formátumú csoportok megjelenítését
- i minden fájlhoz a kapcsolódó csomópontot is megjeleníti
- k a fájlméretet kilobájtból jeleníti meg
- l minden fájl megjelenítése hosszú formátumban történik: a sor tartalmazza a jogosultságokat, a tulajdonost, a csoportot, a méretet stb.
- L szimbolikus csatolásoknál nem a csatolást, hanem azt a fájlt vagy könyvtárat jeleníti meg, amelyre a csatolás mutat
- m a fájlnéveket soronként, vesszőkkel elválasztva jeleníti meg
- n az UID és a GID azonosítót számmal adja meg
- r a fájlokat ellentétes sorrendben jeleníti meg
- R rekurzív módon az alkönyvtákat és tartalmukat is listázza
- s a fájlméretet a fájlnév előtt kilobájtból jeleníti meg
- S a listát a fájlméret szerint listázza
- t a listát a legutóbbi módosítás ideje szerint rendezzi: az újabb fájlok állnak előbb
- u a listát a legutóbbi hozzáférés ideje szerint rendezzi
- x a fájlokat vízsínesen rendezett oszlopokban jeleníti meg
- X a fájlokat kiterjesztésük szerint rendezzi

**m4 [kapcsolók] [fájlok]**

Ezt a makróprocesszort számos program használja, így a GNU Autoconf rendszerben és az fvwm konfigurációs fájlokhöz. A nyelv leírása a GNU Info rendszerben található.

**mail [kapcsolók] [címek]**

Ez a program e-mail levelek olvasására és frására szolgál. A felhasználók általában a pine programot vagy egy grafikus levélolvasót használnak. A mail kiemelkedő viszont szöveges fájlok egy-szerű elküldésére, mivel a tartalom a szabványos bemeneti eszközön keresztül továbbítható.

*Példa:*

```
mail linux@fh-heilbronn.de < critique.txt
```

**make [kapcsolók] [cél]**

Egy make-fájlt olvas be, és egy vagy több célt frissít. A make elsősorban forrásfájlok fordítására használatos. Részletes leírását lásd a GNU Info rendszerben.

A rendelkezésre álló kapcsolók:

- C könyvtár egy make-fájl beolvasása előtt módosítja a megadott alkönyvtárat
  - d további hibakereső információt szolgáltat
  - e a környezeti változók felülről a make-fájlból található megfelelő változókat
  - f make-fájl a megadott make-fájlt használja
  - I könyvtár az importált make-fájlokat a megadott könyvtárban keresi
  - k hibánál csak az aktuális célt szakítja meg, nem a teljes make processzt
  - n a parancsokat csak kiadja, nem hajtja őket végre
  - p belső makródefiníciókat szolgáltat
  - r nem használja az alapértelmezett szabályokat
  - s tiltja a megjelenítést a képernyőn
  - t aktuális dátumú feldolgozandó fájlokat ad ki, a megfelelő műveletet nem hajtja végre
  - w egy művelet végrehajtása előtt és után megjeleníti az aktuális könyvtárat
- 

#### **man [kapcsolók] [[szakasz] név]**

Kézikönyvoldalakat jelenít meg a képernyőn. Ezek az oldalak az /usr/man alatti alkönyvtárban találhatók vagy a MANPATH környezeti változóban megadott könyvtárakban.

A rendelkezésre álló kapcsolók:

- a a megadott névnek megfelelő összes kézikönyvoldalt megjeleníti
  - f egyenértékű a whatis parancsal
  - h súgó oldalt jelenít meg
  - k egyenértékű az apropos parancsal
  - M útnév további könyvtárak listáját adja, amelyekben kézikönyvoldalak kereshetők
  - w nem kézikönyvoldalt jelenít meg, hanem annak elérési útvonalát
- 

#### **mesg [y | n]**

Azt határozza meg, hogy a felhasználók küldhetnek-e üzeneteket terminálra a write parancssal. Ha a mesg parancsot kapcsolók nélkül hívjuk, az aktuális állapotot jeleníti meg.

#### **mkdir [kapcsolók] könyvtárak**

Könyvtárat hoz létre.

A rendelkezésre álló kapcsolók:

- m jogok új könyvtárat hoz létre a megadott jogosultságokkal
  - p ha olyan könyvtár elérési útvonala a megadott, amelyben nincsenek alkönyvtárak, akkor ezeket létrehozza a parancs
- 

#### **more [kapcsolók] [fájlok]**

A fájlokat jeleníti meg képernyő-oldalanként. Az <Enter> egy sorral lefelé gördi a listát, míg a <Szóköz> billentyű a következő képernyőoldalra lépet. A <h> a súgót jeleníti meg, amely az összes parancs ismertetését tartalmazza, míg a <q> lezárája a more működését. Ha nincs megadott fájl, akkor a more a szabványos bemeneti eszközről olvas.

*A rendelkezésre álló kapcsolók:*

- +#** a megadott sorszámnál kezdi a megjelenést
- d** minden képernyőoldal alján megjeleníti a „Press space to continue, 'q' to quit” (Szóköz: következő oldal, 'q': kilépés) üzenetet
- f** az oldaltörést nem a képernyősorok, hanem a logikai sorok szerint számítja, és a tört sorokat csak egyszer számítja
- l** a lapdobás vezérlő karaktert (^L) figyelmen kívül hagyja
- s** az ismétlődő szomszédos üres karaktereket figyelmen kívül hagyja
- u** az aláhúzást figyelmen kívül hagyja

**mtools**

Olyan parancsok csoportja, amelyekkel MS-DOS fájlrendszerek egyszerű elérése valósítható meg. Általában ezek hajlékonylemezek kezelésére szolgálnak. Merevlemez DOS partíciójának elérése egyszerűbb, ha ezt csatoljuk (lásd a mount parancsot). Az egyes parancsok nagyon hasonlítanak a megfelelő DOS parancsokhoz. Ez azt jelenti, hogy a hajlékonylemezek a DOS szokásos betűjelzéseihez (A:, B:) érhetők el, ha a lemezegységeket az /etc/mtools fájlban megfelelően konfigurálták.

*Parancsok:*

- mattrib** a fájlattributumokat módosítja
- mcd** az aktuális könyvtárat állítja be
- mcopy** fájlokat másol
- mdel** fájlokat töröl
- mdir** könyvtár tartalmát listázza
- mformat** a DOS fájlrendszer szerint hajlékonylemez formáz
- mlabel** az adathordozó címkejét módosítja
- mmd** alkönyvtárat hoz létre
- mrd** alkönyvtárat töröl
- mren** fájlt átnevez
- mtype** fájl tartalmát jeleníti meg

**mount [kapcsolók] [eszköz] [csatolási\_hely]**

Új fájlrendszeret csatol egy könyvtárstruktúrába. Egy fájlrendszer a megadott csatolási helyen csatlakoztatott a UNIX fájlstruktúrába. A meg nem adott paramétereit a parancs az /etc/fstab fájl megfelelő bejegyzéseiből veszi.

*A rendelkezésre álló kapcsolók:*

- a** automatikusan csatolja az /etc/fstab fájlban megadott összes fájlrendszeret
- f** figyelmen kívül hagyja az aktuális mount rendszerhívást (a -v kapcsolóval együtt hasznos)
- n** az /etc/fstab fájl bejegyzéseit figyelmen kívül hagyja
- o kapcsolók** további beállítások, amelyek az adott fájlrendszerrel függenek

*Általános kapcsolók:*

- async** az összes bemeneti és kimeneti művelet aszinkron
- auto** a fájlrendszer csatolható az -a kapcsolóval
- defaults** szabványos kapcsolók: *rw, suid, dev, exec, auto, nouser, async*
- dev** lehetővé teszi a karakter és blokkorientált eszközök használatát
- exec** parancsok végrehajtását teszi lehetővé

**noauto** csak közvetlenül csatolható, az - a kapcsolával nem  
**nodev** karakter- és blokkorientált eszközök nem használhatók  
**noexec** parancsok végrehajtása nem lehetséges  
**nosuid** a SUID és az SGID bit hatásának  
**nouser** az egyszerű felhasználó számára nem engedélyező fájlrendszer csatolását  
**remount** lehetővé teszi egy fájlrendszer újból csatolását, azaz a csatolási beállítások módosítását  
**ro** a fájlrendszer csak olvashatóan csatolja; ezt a kapcsolót kell használni CD-ROM  
 fájlrendszer csatolásákor  
**rw** a fájlrendszer írható-olvasható beállítással csatolja  
**suid** lehetővé teszi SUID és SGID parancsok végrehajtását  
**sync** az összes bemeneti és kimeneti művelet szinkron  
**user** az egyszerű felhasználó számára is engedélyező fájlrendszer csatolását

#### A fájlrendszertől függő kapcsolók:

**case=[lower | asis]** a kisbetű/nagybetű különbségeit állítja be (hpfs)  
**check=érték** fájlrendszer csatolása előtt lehetővé teszi konziszenciavezérlést  
 választását (ext2)  
**none** nincs konziszenciavezérlés  
**normal** az i-csomópontot és a blokk-bittérképet ellenőrzi (ez az alapértelmezés)  
**strict** a szabad blokok konziszenciáját is vizsgálja  
**check=érték** a fájlnévek megadásának módját határozza meg (msdos)  
**relaxed** kisbetű/nagybetű különbözik, a hosszú fájlnéveket csonkolja  
**normal** a különleges karakterek (\*, ?, < stb.) nem megengedettek (ez az alapértelmezés)  
**strict** hosszú fájlnévek és különleges karakterek nem megengedettek  
**conv=érték** azt határozza meg, hogy a sor vége (EOL) karakter a fájlrendszerhez való  
 hozzáéréskor konvertálásra kerül-e (msdos, hpfs, iso9660)  
**binary** nincs EOL-konverzió  
**text** CRLF/LF konverzió az összes fájlnál  
**auto** nincs konverzió a következő kiterjesztésű fájloknál: exe, com, bin, app,  
 sys, drv, ovl, ovr, obj, lib, dll, pif, arc, zip, lha,  
 zoo, tar, z, arj, tz, taz, tzp, tpz, gif, bmp, tif,  
 gl, jpg, pcx, tfm, vf, gf, pk, pxl, dvi  
**block=érték** az iso9660 fájlrendszerben a blokkméretet adja meg  
**crust** a crust jelzőbitet állítja be hibák figyelmen kívül hagyására bizonyos CD-ROM  
 programokban (iso9660)  
**debug** hibakeresési üzeneteket hoz létre (ext2, msdos)  
**errors=érték** a hibakezelést határozza meg (ext2)  
**continue** nincs egyedi hibakezelés (ez az alapértelmezés)  
**remount ro** a fájlrendszer újraszatolása csak olvashatóként  
**panic** hibánál rendszermag-hibakezelést indít el  
**fat=érték** felülről az automatikusan érzékel FAT típust (az érték 12 vagy 16 lehet) (msdos)  
**gid=érték** a fájlrendszerben minden egyes fájlhoz megállapítja a GID értékeit (msdos, hpfs)  
**grpid** az új fájlok ugyanazt a GID értéket kapják, mint az a könyvtár, amelyben létrehozták (ext2)  
**nocheck** egyenértékű a check=none kapcsolóval (ext2)  
**nogrid** az új fájlok a létrehozó eljárás GID értékét kapják (mint a System V rendszerben)  
 (ext2, ez az alapértelmezés)  
**norock** kikapcsolja a Rockridge kiterjesztések, a kisbetű/nagybetű különbséget  
 quiet és a hosszú fájlnévek használatát (iso9660)  
 quiet a chmod és a chown parancs végrehajtására kísérletekor tiltja a megfelelő hibaüzeneteket  
 (msdos)

**sb=érték** a megadott blokkpozícióban (általában az 1, 8193, 16385, ... pozícióban) alternatív szuperblokkot használ (ext2)

**sysvgroups** lásd nogrpid

**uid=érték** a fájlrendszerben minden egyes fájlhoz megállapítja az UID értéket (msdos, hpfs)

**umask=érték** a fájlokhoz az umask értéket határozza meg (msdos, hpfs)

**-r** a fájlrendszeret csak olvashatóan csatolja

**-t típus** adott típusú fájlrendszer csatol (alapértelmezés: minix; a lehetséges értékek: minix, ext, ext2, xiafs, msdos, hpfs, proc, nfs, iso9660, sysv, xenix, coherent)

**-v** részletes üzeneteket jelent meg

---

### **mv [kapcsolók] útnév cél**

Fájlokat vagy könyvtárakat áthelyez vagy átnevez. Ha a cél már létezik és fájl, akkor felülírja; ha könyvtár, akkor a megadott fájlokat és könyvtárakat áthelyezi a létező könyvtárba. Ha a cél nem létezik, akkor csak egy fájl vagy egy könyvtár adható meg forrásként, és azt átnevezi a megadott célnévre.

#### A rendelkezésre álló kapcsolók:

- b** a fájlról, felülírása előtt, biztonsági másolatot készít
- f** fájlok felülírása előtt nem kér jóváhagyást
- i** fájlok felülírása előtt jóváhagyást kér
- u** fájlt csak akkor helyez át, ha az újabb az azonos nevű célfájlnál
- S** lásd a cp parancsot
- V** lásd a cp parancsot

---

### **nice [-n érték | -érték] parancsok [argumentumok]**

Magasabb nice-szintű (azaz kisebb prioritású) parancsokat hajt végre. A nice parancsot általában a héjprogramba integrálják. A maximális nice-szint 19. A rendszer adminisztrátora negatív értékeket (-20-ig) is megadhat. Az alapértelmezett nice-szint érték 10.

---

### **nm [kapcsolók] fájlok**

A tárgy-fájlok vagy könyvtárak szimbólumtáblázatát adja meg.

---

### **nohup parancs [argumentumok]**

A parancsot általában a héjprogramba integrálják. A héjprogram leállását akadályozza meg, amikor a megadott parancs befejezi működését.

---

### **nroff [kapcsolók] fájlok**

Képernyőre vagy nyomtatóra megfelelő formázó utasításokat tartalmazó fájlokat formáz (lásd még a groff parancsot).

---

### **openwin**

Az X11 környezet indítására szolgáló szkript.

**passwd [felhasználó]**

A felhasználó saját jelszóját módosítja. A rendszer adminisztrátora más felhasználók jelszóját is módosíthatja.

**pr [kapcsolók] [fájlok]**

Szöveges fájlokat készít elő nyomtatásra. A fájl tartalmát oldalanként állítja elő, a cím-sorban a dátum, a fájlnév és az oldalszám helyezhető el.

*A rendelkezésre álló kapcsolók:*

- oldal a nyomtatást a megadott oldalon kezdi
- oszlop többoszlopos nyomtatási képet ad
- a az oszlopokat nem egymás alá, hanem egymás mellé nyomtatja
- c a nem nyomtatható karaktereket a „^” karakterrel jelöli
- d dupla sortávolságú nyomtatás
- e[karakterek[szélesség]] tetszőleges karaktert adott számú üres karakterrel helyettesít; a tabulátor karakternél az alapértelmezés 8 szóköz karakter
- f az oldal végén az üres sorok helyett lapdobás karaktert küld ki
- h szöveg a címsorban a fájlnévet a megadott szöveggel helyettesíti
- i[karakterek[szélesség]] az -e kapcsoló ellenértéket hajtja végre
- l hossz az oldal hosszát állítja be (az alapértelmezés 66 sor)
- n [karakterek[szélesség]] a sorok elő sorszámot illeszi; a szám és a szöveg elválasztására szolgáló karakter és a szám mérete (szélesség) is megadható
- o szélesség megadott szélességű bal oldali marginát állít be
- r a nem megnyitható fájloknál tiltja a hibaüzenetek megjelenítését
- t tiltja az fejléc és a láblec megjelenítését
- v a nem nyomtatható karaktereket formázva jeleníti meg
- w a sor szélességét adja meg (az alapértelmezés 72 karakter)
- x a terminálhoz nem hozzárendelt processzeket jeleníti meg

**ps [kapcsolók]**

Az aktuálisan aktív processzek listáját adja.

*A rendelkezésre álló kapcsolók:*

- a az összes felhasználó processzeit megjeleníti
- h a fejlesort nem jeleníti meg
- j a processz csoportazonosítóját és futásazonosítóját jeleníti meg
- l részletes kimeneti formátum
- m a tárolási helyfoglalás áttekintését adja
- r csak az éppen futó processzeket listázza
- s a jelzés állapotról szolgáltat információt
- u a processz tulajdonosának nevét és az indítási időt adja meg
- w széles kimeneti formátumnál tiltja a parancsorok csontolását

**pwd**

Az aktuális könyvtár teljes elérési útvonalát adja meg.

---

**rcp** [kapcsolók] források cél

Fájlokat másol számítógépek között. A források és a cél formátuma felhasználó@gazda:útnév, ahol a felhasználó elhagyható, és ekkor aktuális felhasználó nevét használja a parancs. Helyi fájlok nál csak az útnevet kell megadni.

A rendelkezésre álló kapcsolók:

- r rekurzív módon alkönyvtárakat és tartalmukat másolja
  - p a másolás során megtartja a fájl attribútumait (dátum, jogosultságok)
- 

**rlogin** [kapcsolók] gazdanév

A telnet parancshoz hasonlóan kapcsolatot létesít a megadott gazdagéppel, és bejelentkezik azon. Ha a távoli gazdagépen az aktuális felhasználót bevitték az .rhosts vagy /etc/hosts.equiv fájlba, a jelszó megadása nem szükséges.

A rendelkezésre álló kapcsoló:

- l név a távoli gazdagépen a nép paramétert használja felhasználónévként
- 

**rm** [kapcsolók] fájlok

Egy vagy több fájlt távolít el. Fájl eltávolításához írási jogosultság szükséges az azt tartalmazó könyvtárhoz. Ha a fájl írásvédelettel, jóváhagyás szükséges. Könyvtárak az rmdir parancssal távolíthatók el.

A rendelkezésre álló kapcsolók:

- f a fájlokat jóváhagyás nélkül akkor is eltávolítja, ha azok írásvédettek
  - i minden fájlnál jóváhagyást kér
  - r rekurzív módon eltávolítja az alkönyvtárakat és tartalmukat
  - v az eltávolításkor egyenként megjeleníti a fájlneveket
- 

**rmdir** [kapcsolók] könyvtárak

Könyvtákat távolít el. Ehhez a könyvtárnak üresnek kell lennie. Alkönyvtárak és tartalmuk eltávolítására az rm parancs használható a -r kapcsolóval.

---

**rsh** [kapcsolók] gazdanév [parancsok]

Parancsok hajt végre a távoli gazdagépen. A hozzáférési jogosultság az /etc/hosts.equiv vagy az ~/.rhosts fájl megfelelő bejegyzésével adható meg.

A rendelkezésre álló kapcsolók:

- l felhasználó a megadott parancsot másik felhasználónéven kíséri meg végrehajtani
- n a szabványos bemeneti eszközöt a /dev/null eszközre irányítja át (a csh problémáknál használható)

Példa:

rsh -luhl zeus.demo.de ls  
az ls parancsot hajtja végre uhl felhasználónével a zeus.demo.de gazdagépen

**sdiff [kapcsolók] fájl1 fájl2**

Két fájlt hasonlít össze, és az eltéréseket két oszloban adja meg (lásd a diff parancsot is). Ez az eredményt áttekinthetőbb, mint a **diff** parancsé. A valamelyik fájlból hiányzó sorok jelölése a „<” vagy a „>”, míg a különböző sorpároké a „|” karakterrel történik.

**sed [kapcsolók] [fájlok]**

A felhasználó közreműködése nélkül módosításokat hajt végre. A parancsot általában szkriptekben használják szöveg cseréje, törlése vagy beszúrása végrehajtásához. Ha nincs megadott **fájl1**, akkor a **sed** a szabványos bemeneti eszközöt használja.

*A rendelkezésre álló kapcsolók:*

- e 'utasítások' a megadott fájlokon szerkesztő **utasításokat** hajt végre
- f szkriptfájl a szerkesztő **utasításokat** a **szkriptfájlból** olvassa be
- n a képernyőn nem jeleníti meg a bemeneti sorokat

**shutdown [kapcsolók] időpont [üzenet]**

A rendszer futási szintjét módosítja vagy lezárja a rendszert. Az **időpont** és a figyelmeztető **üzenet** argumentumként átadható. Azonnali lezáráshoz az időpont argumentum értéke **now**.

*A rendelkezésre álló kapcsolók:*

- c folyamatban lévő lezárást szakít meg
- h az összes processz leállításával és a fájlrendszer lecsatolásával lezárja a rendszert
- k nem hajt végre lezárást, csak a figyelmeztető üzenetet jeleníti meg
- r újraindítsa a rendszert
- t mp a figyelmeztető **üzenet** és a **kill** jelzés küldése közötti késleltetést (mp-ben) állítja be

**sleep idő**

Várakozás a megadott **ideig** (mp-ben). A parancsot általában szkriptekben használják.

**sort [kapcsolók] [fájlok]**

A megadott fájlban a sorokat rendezzi. Ha nincsenek megadott **fájlok**, akkor a szabványos bemeneti eszközöt használja a parancs.

*A rendelkezésre álló kapcsolók:*

- +n-m az n és m közötti rendezési kulcsot állítja be
- b a vezető üres karaktereket figyelmen kívül hagyja
- c ellenőrzi, hogy a megadott fájlok rendezettek-e: ha igen, akkor a program futása hibájának befejeződik
- d a rendezésnél figyelmen kívül hagyja az írásjeleket
- f a rendezésnél különbséget tesz a kis- és a nagybetűk között
- i a nem nyomtatható ASCII karaktereket figyelmen kívül hagyja
- m felcseréli a két bemeneti fájlt
- M az első három karaktert hónapnévként (JAN, FEB, ..., DEC) értelmezi, és hónapok szerint hajtja végre a rendezést

- n numerikusan rendez
- o fájl a szabványos kimeneti eszközt a megadott fájlba irányítja át
- r megfordítja a rendezési sorrendet
- t kar az oszlopok közötti határoló karaktert adja meg (az alapértelmezés szóköz vagy tabulátor)
- u az ismétlődő sorokat eltávolítja

Példa:

```
sort +2n -t:/etc/passwd
a jelszófájl rendezi numerikusan a harmadik oszlop szerint
```

### **strings [kapcsolók] fájlok**

Bináris, tárgy- vagy programfájlokban karakterszorozatokat keres. Karakterszorozat minden négy vagy több nyomtatható karakterből álló és a null karakterrel lezárt sorozat.

A rendelkezésre álló kapcsolók:

- a általában a tárgyfájloknál, csak a kód- és az adatszegmensben hajtja végre a keresést;
- e a kapcsoló biztosítja, hogy a teljes fájl feldolgozásra kerüljön
- f minden karakterszorozat előtt a megfelelő fájlnév áll
- n a karakterszorozat minimális hosszát határozza meg (az alapértelmezés 4)
- o a karakterszorozat pozícióját bajtban adja meg

### **strip [kapcsolók] fájlok**

Tárgy- és programfájlok ból kitörli a szimbólum, hibakereső, sorszám és egyéb információkat, így csökken a méretük.

### **stty [kapcsolók] [üzemmódok]**

A terminál IO üzemmódokat állítja be. Ez magában foglalja a terminál összes általános paraméterét, valamint a sebesség, a handshaking és a különleges karakterek funkcióinak beállítását. A --help kapcsolóval az összes rendelkezésre álló beállítás lista jeleníthető meg.

A rendelkezésre álló kapcsolók:

- a az összes aktuális beállítást jeleníti meg
- help a súgó tárnyítható

### **su [-] [felhasználó] [argumentum]**

A héjprogramot indítja másik felhasználóval. Ez a parancs olyan terminálon való bejelentkezésre használható, amelyet másik felhasználó használ. Ha a felhasználó nem megadott, akkor egy root héjprogram indítására kerül sor. Az új héjprogram az exit vagy a <Ctrl-d> beírásával zárható le. A „-“ megadásakor a teljes bejelentkezési eljárársa sor kerül, míg a -c megadásakor a parancsok másik felhasználói azonosítóval hajthatók végre.

### **tail [kapcsolók] [fájlok]**

A megadott fájlok utolsó tíz sorát adja meg.

A rendelkezésre álló kapcsolók:

- c [blkfm] az utolsó n bájtot adja meg blokk (b), kilobájt (k) vagy megabájt (m) egységeben
- f az utolsó sor kiadása után nem fejezi be a működést, hanem megvárja a fájlhoz kiírását.
- n Amint új sorok kerülnek hozzáírásra a fájlhoz, azt is kiírja. Ebben az üzemmódban a program a break (<Ctrl-c>) segítségével állítható le. Ez az üzemmód elsősorban naplósájlok nyomón követésére alkalmas.
- v az utolsó n sort adja ki
- v címsorként a fájl nevét jeleníti meg

### **talk felhasználó [@gazdanév] [tty]**

Talk kapcsolatot létesít a megadott felhasználóval. Ha ez a felhasználó több terminálon jelentkezik be, akkor a használó terminál beírható a parancssorban. A talk kapcsolat két részre osztja a képernyőt: a helyi beviterek a képernyő felső felén, míg a távoli gépen bevitt adatok a képernyő alsó felén jelennek meg. A kapcsolat a <Ctrl-c> segítségével szüntethető meg. Sajnos a talk két, egymással nem kompatibilis verziója létezik, így a kapcsolat különböző platformok között nem minden sikeres.

### **tar [kapcsolók] [archív] [fájlok]**

Tar archívumokat kezel (eredetileg mágnesszalagon). Ez a parancs fájlokat ír az archívba, vagy fájlokat olvas be onnan. A következő műveletek közül legalább egyet paraméterként át kell adni.

A rendelkezésre álló műveletek:

- c archív létrehozása
- r fájlokat fűz az archívhoz (nem mágnesszalagon)
- t egy archív tartalmát adja ki
- u fájlokat fűz az archívhoz, ha azokat még nem tartalmazza, vagy ha azok módosultak (nem mágnesszalagon)
- x fájlokat bont ki archívóból

A rendelkezésre álló kapcsolók:

- b n a blokkiányezést az n értékre állítja be
- f archív az archívot adja meg, amely lehet normál fájl, vagy eszköz fájl egy mágnesszalagos egységen, például /dev/rmt0, illetve hajlékonylemez egységen, például /dev/fd0
- h a szimbolikus csatolások helyett a hivatkozott fájlokat archiválja
- k a létező fájlokat nem fríja felül
- L a szimbolikus csatolásokat követi
- m a kibontott fájlok módosítási idejére az aktuális időpontot állítja be
- M a kibontást több adathordozón elhelyezkedő archívóból hajtja végre (például több hajlékonylemezről)
- N dátum csak azokat a fájlokat archiválja, amelyek újabbak, mint a megadott dátum
- o a kibontott fájlok tulajdonosaként az aktuális felhasználót állítja be
- O a fájlokat a szabványos kimeneti eszközre bontja ki
- v az archiválásnál vagy a kibontásnál megjeleníti a fájlnévét
- z az archív létrehozásakor tömörítést, a kibontáskor kifejtést alkalmaz

Példák:

```
tar -cvf archive.tar *
az aktuális könyvtában lévő összes fájlt és alkönyvtárat az archive.tar archívba menti
tar -cvf /dev/fd0 *.txt
az aktuális könyvtában lévő összes .txt kiterjesztésű fájlt az első hajlókonyolemezre menti
tar -xvf20 /dev/rmt0
az első mágnesszalagos egységről kibontja az összes fájlt (a blokkméret 20)
tar -tvf zarchive.tar.z
egy tömörített tar archív tartalmát listázza
```

---

#### **tee [kapcsolók] [fájlok]**

Ez a program szűrőként használatos. A szabványos bemeneti eszközök között másolja a szabványos kimeneti eszközre és a megadott fájlokba.

A rendelkezésre álló kapcsolók:

- a a szabványos bemeneti eszközről kapott adatokat a felülírás helyett a fájl végéhez fűzi
  - i a megszakítási jeleket figyelmen kívül hagyja
- 

#### **telnet [gazdanév [port]]**

A megadott gazdagéphez létesít kapcsolatot a telnet protokoll használatával. A portszám is megadható. Ezt a programot gyakran használják az egyes portokon rendelkezésre álló szolgáltatók ellenőrzésére. Ha a gazdanév nem megadott, a telnet parancsüzemmódba kerül, amelyben telnet parancsok vihetők be. A help parancs a legfontosabb parancsokat listázza.

#### **test feltétel**

A megadott feltételt értékeli ki: zérus értéket ad vissza, ha az eredmény igaz, egyébként a zérustól eltérő értéket. A feltétel szögeletes zárójelek között is megadható (ezt a megadást elsősorban szkriptekben használják).

Fájlok:

- b fájl a fájl blokkeszköz
- c fájl a fájl karaktereszköz
- d fájl a fájl könyvtár
- f fájl a fájl normál fájl
- g fájl a fájl csoportazonosító bitjét (SGID) állítja be
- G fájl a GID megegyezik a tulajdonos csoportjával
- k fájl a fájl sticky bitje beállított
- O fájl az UID megegyezik a fájl tulajdonosával
- p fájl a fájl elnevezett adatcsatorna
- r fájl a fájl létezik és olvasható
- s fájl a fájl mérete nagyobb 0 bajtnál
- S fájl a fájl foglalat
- t n az n fájlleíró terminálnak felel meg
- u fájl a fájl felhasználó-azonosító bitje (SUID) beállított
- w fájl a fájl létezik és írható
- x fájl a fájl létezik és végrehajtható
- d1 -ef d2 a d1 és a d2 fájl csatolt

**d1 -nt d2** a *d1* fájl újabb, mint a *d2*  
**d1 -ot d2** a *d1* fájl korábbi, mint a *d2*

*Karaktersorozatok:*

**-n z1** a *z1* karaktersorozat hossza nagyobb zérusnál  
**-z z1** a *z1* karaktersorozat hossza zérus  
**z1 a z1** karaktersorozat nem null  
**z1=z2** *z1* egyenlő *z2*-vel  
**z1 != z2** *z1* nem egyenlő *z2*-vel  
**z1<z2** *z1* az ábécésorrend szerint kisebb *z2*-nél  
**z1>z2** *z1* az ábécésorrend szerint nagyobb *z2*-nél

*Numerikus feltételek:*

**n1 -eq n2** *n1* egyenlő *n2*-vel  
**n1 -ge n2** *n1* nagyobb vagy egyenlő *n2*-vel  
**n1 -gt n2** *n1* nagyobb *n2*-nél  
**n1 -le n2** *n1* kisebb vagy egyenlő *n2*-vel  
**n1 -lt n2** *n1* kisebb *n2*-nél  
**n1 -ne n2** *n1* nem egyenlő *n2*-vel

*Kombinatorikai feltételek:*

**!a1** igaz, ha az *a1* kifejezés hamis  
**a1 -a a2** igaz, ha *a1* és *a2* igaz  
**a1 -o a2** igaz, ha *a1* vagy *a2* igaz

*Példák:*

```
if [ -f /etc/shadow ]
    ellenőrzi, hogy létezik-e az /etc/shadow fájl
if [ „$res” != „j” ]
    ellenőrzi, hogy a res változó tartalma „j”
while [ -z „$res” ]
    ellenőrzi, hogy a res változó tartalma türes karaktereket-e
```

**time** parancs [argumentum]

Végrehajtja a megadott parancsot, és kijelzi a végrehajtási időt.

---

**touch** [kapcsolók] fájlok

A fájlok utolsó elérésének dátumát/idejét és az utolsó módosítás dátumát/idejét módosítja. Ha a megadott fájl nem létezik, üres fájlként létrehozza.

*A rendelkezésre álló kapcsolók:*

- a** csak az utolsó hozzáférés időpontját módosítja
- c** nem létező fájloknál nem hoz létre üres fájlokat
- m** csak az utolsó módosítás időpontját módosítja
- r** fájl az időpontot a megadott fájlból veszi
- t érték** a fájl dátumát és a rendszeridőt a megadott értékre állítja; a formátum HHNNÓópp  
 (hónap, nap, óra, perc)

---

**tr [kapcsolók] [karaktersor1 [karaktersor2]]**

A szabványos bemeneti eszközt másolja a szabványos kimeneti eszközre, és a folyamat közben karaktereket helyettesít vagy töröl. Ha a **karaktersor1** megtalálható a szabványos bemeneti eszközön, akkor a **karaktersor2** megfelelő karakterével helyettesíti.

*A rendelkezésre álló kapcsolók:*

- c a **karaktersor1** karaktereinek komplementesét adja ki
  - d törli a **karaktersor1**-ben megtalálható karaktereket
  - s a kimenetben tiltja az ismétlődéseket
- 

**troff [kapcsolók] [fájlok]**

Nyomtatóra vagy levilágítóra küldött fájlokat formáz (lásd még az **nroff** és a **groff** parancsot is).

---

**true**

Ez a parancs a 0 („sikeres”) értéket adja visszatérési értékként. Elsősorban szkriptekben használják.

---

**umask [érték]**

A fájlgenerációs maszk aktuális értékét adja eredményül oktális számként, illetve beállítja ezt az értéket. Ez a maszk az újonnan létrehozott fájl maximálisan elfogadható jogosultságait határozza meg. Itt az **umask** érték kivonásra kerül a létrehozandó fájl jogosultságaiból.

---

**uname [kapcsolók]**

Az aktuális rendszer nevét és verziószámát adja eredményül.

*A rendelkezésre álló kapcsolók:*

- a az összes rendelkezésre álló adatot megadja
  - m a hardver (processzor) típusát adja meg
  - n a gazdanevet adja meg
  - r az operációs rendszer verziószámát adja meg
  - s az operációs rendszer nevét adja meg
  - v a rendszermag fordításának dátumát és időpontját adja meg
- 

**uncompress [kapcsolók] [fájlok]**

A **compress** parancssal tömörített eredeti fájlt állítja vissza.

*A rendelkezésre álló kapcsoló:*

- c a fájl tartalmát a szabványos kimeneti eszközre küldi. Itt az **uncompress** úgy működik, mint a **zcat** parancs.

**uniq [kapcsolók] [fáj1][fáj2]**

A soronként rendezett fáj1 fájlban törli az egymás utáni azonos sorokat, és ezeket a fáj1-be (vagy a szabványos kimeneti eszközre) tölti.

*A rendelkezésre álló kapcsolók:*

- c az ismétlődék számát adja meg
- d csak az ismétlődő sorokat adja ki
- u csak az egyedi sorokat adja ki
- n két sor összehasonlítása előtt n számú (szöközzel vagy tabulátor karakterrel elválasztott) mezőt kihagy
- +n az összehasonlítás elkezdése előtt n számú karaktert kihagy
- w az összehasonlítandó karakterek számát határozza meg

**uptime**

Az aktuális időt, az utolsó újraindíthatás óta eltelt időt, a bejelentkezett felhasználók számát és az éppen betöltött rendszert adja meg.

**uudecode [fájlok]**

Az uuencode parancsal kódolt fájlt dekódolja az eredeti nevet, tulajdonost és jogosultságokat használva.

**uuencode [fáj1] név**

Bináris fájlt kódol ASCII fájlformátumra, így az e-mail rendszeren elküldhető. A kódolt fájl mintegy 35%-kal nagyobb az eredetinél. Az eredményt a szabványos kimeneti eszközre küldi. A megadott név a fájl neve lesz, amikor a címzett kibontja azt.

**vi [kapcsolók] [fájlok]**

ASCII fájlok feldolgozására szolgáló teljes képernyős szerkesztő. Nagy mértékben az ex szerkesztőn alapul, és általában az összes terminálon működik.

**w [kapcsolók] [felhasználók]**

Az összes jelenleg bejelentkezett felhasználót és műveleteket listázza. Ha nincs megadott felhasználó, akkor az összes felhasználót megadjá.

*A rendelkezésre álló kapcsolók:*

- h a címsort nem jeleníti meg
- f azt határozza meg, hogy a bejelentkezési terminál kimenet legyen-e összefoglaló kimeneti formátum

**wc [kapcsolók] [fájlok]**

Szöveges fájlban a karakterek, a szavak és a sorok számát adja meg.

A rendelkezésre álló kapcsolók:

- c csak a karakterek számát adja meg
- l csak a sorok számát adja meg
- w csak a szavak számát adja meg

#### **whatis [parancsok]**

A kézikönyvoldalak alapján a megadott parancsok rövid ismertetését jeleníti meg.

#### **which [parancsok]**

A megadott parancsok (általában belső parancsok) fájlelérési útvonalát adja meg.

#### **who [kapcsolók] [fájl]**

Az aktuálisan bejelentkezett felhasználók listáját, a hozzájuk tartozó terminált, a bejelentkezés idejét és annak a gazdagépének a nevét adja meg, amelyen bejelentkeztek. Ha fájlnév is megadott, akkor az /etc/utmp fájl helyett ezt használja a kiértékeléskor.

A rendelkezésre álló kapcsolók:

- am i a felhasználó saját adatait adja meg
- i azt adja meg, hogy a felhasználó mennyi ideig volt inaktív
- H oszlopfejéléket ad meg
- q csak a bejelentkezési nevet és felhasználók számát adja meg
- w az adja meg, hogy a felhasználó elfogadta (+) vagy nem (-) a write parancssal előállított üzeneteket

#### **write felhasználó [terminál]**

A felhasználó adott termináljára küld üzenetet. Az üzenetet a szabványos bemeneti eszköz az EOF (<Ctrl-d>) karakterig olvassa.

#### **xargs [kapcsolók] [parancsok]**

Parancsot hajt végre a szabványos bemeneti eszkösről beolvasható argumentumokkal. Ez lehetővé teszi tetszőleges hosszúságú argumentumok átadását a parancsoknak.

A rendelkezésre álló kapcsolók:

- 0 a fájnevek a null karakterrel végződnek
- e karaktersorozat a feldolgozás befejeződik, amint a megadott karaktersorozat szerepel a fájnevek listájában (az alapértelmezés a „\_” karakter)
- l n a parancsot n számú argumentummal hajtja végre
- n n a parancsot legfeljebb n argumentummal hajtja végre
- p interaktív feldolgozás, ahol a felhasználónak az „y” választ kell adnia a parancs végrehajtása előtt
- s n az egyes argumentumok legfeljebb n számú karaktert tartalmazhatnak
- t a parancsot végrehajtása előtt megjeleníti

#### **zcat [fájlok]**

A megadott fájlokat bontja ki, és tartalmukat a szabványos kimeneti eszközre küldi. A tömörített fájlok változatlanok maradnak.

theoretical framework, and the way it has been applied to the study of the history of science. In this section I will focus on the first two topics, leaving the third one to the next section. I will also limit myself to the history of science, as the history of philosophy is a separate field of inquiry.

The first topic concerns the relationship between philosophy and science. This is a question that has been debated for centuries, and it is still a topic of active research. On the one hand, some philosophers believe that science is a branch of philosophy, and that the methods of science are based on philosophical principles. On the other hand, many scientists believe that science is an independent discipline, and that its methods are based on empirical observation and experimentation. In this section, I will argue that the relationship between philosophy and science is more complex than either of these views suggest. I will also discuss the role of philosophy in the development of science, and the ways in which philosophy can contribute to our understanding of scientific concepts and theories.

The second topic concerns the nature of scientific theory. This is a question that has been debated for centuries, and it is still a topic of active research. On the one hand, some philosophers believe that scientific theory is based on a priori principles, and that it is not subject to empirical observation. On the other hand, many scientists believe that scientific theory is based on empirical observation and experimentation. In this section, I will argue that the nature of scientific theory is more complex than either of these views suggest. I will also discuss the role of philosophy in the development of scientific theory, and the ways in which philosophy can contribute to our understanding of scientific concepts and theories.

The third topic concerns the role of philosophy in the development of science. This is a question that has been debated for centuries, and it is still a topic of active research. On the one hand, some philosophers believe that philosophy is irrelevant to the development of science. On the other hand, many scientists believe that philosophy is essential to the development of science. In this section, I will argue that the role of philosophy in the development of science is more complex than either of these views suggest. I will also discuss the ways in which philosophy can contribute to the development of science, and the ways in which science can contribute to the development of philosophy.

# FÜGGELÉK

## AZ /ETC FÁJLOK ÁTTEKINTÉSE

A legtöbb konfigurációs fájl az /etc könyvtárban található. Ez a szakasz a legfontosabb konfigurációs fájlokat tekinti át. Az itt következő lista elsősorban a Slackware változat alapján készült, amely nem teljesen kompatibilis az új Linux File System Standard (FSTND) szabvánnyal.

- **bootptab** - a bootpd démonhoz tartozó konfigurációs fájl
- **sh.cshrc** - a (t)csh globális definíciói
- **csh.login** - a (t)csh globális bejelentkezési definíciói
- **diphosts** - SLIP kapcsolat létrehozására alkalmas számítógépek felsorolása
- **DIR\_COLORS** - az ls parancs színbeállításait tartalmazó konfigurációs fájl
- **disktab** - a nem szabványos merevlemez paramétereit megállapítására szolgáló fájl a LILO számára
- **exports** - az NFS rendszerrel exportáláンド könyvtákat adja meg
- **fdprm** - hajlékonylemezes egységek paraméterei
- **fstab** - a betöltésnél csatolandó vagy aktivizálandó fájlrendszeret és cserépartíciókat tartalmazza
- **ftp.access** - konfigurációs fájl az FTP démon számára; lehetővé teszi a wu-ftp vagy a diku-ftp számára hozzáférési megkötések és üzenetek definiálását
- **ftpusers** - konfigurációs fájl az FTP démon számára; az ebben a fájlból felsorolt felhasználók nem jelentkezhetnek be FTP-vel
- **gateways** - elosztók listáját tartalmazó fájl
- **gettydefs** - konfigurációs fájl a getty számára
- **group** - ez a fájl az összes csoportot és az ezekhez tartozó felhasználókat sorolja fel. Ha egy felhasználó több csoportba tartozik, akkor felhasználó neve több csoportnál előfordul. A passwd fájlból csoportnévként csak a felhasználó elsődleges csoportjának neve szerepel.
- **gshadow** - háttérfájl a group fájlihoz
- **host.conf** - egy IP cím keresési szekvenciáját adja meg
- **hosts** - ez a fájl más számítógépek nevét és IP címét tartalmazza. A host.conf fájl beállításaitól függően ebben a fájlból egy gazdagép címének keresése a névkiszolgálóban való keresés előtt vagy után történik.
- **hosts.allow** - a helyi hálózati szolgáltatásokat igénybe vehető számítógépek listáját tartalmazza
- **hosts.deny** - azon számítógépek listáját tartalmazza, amelyek nem vehetik igénybe a helyi hálózati szolgáltatásokat
- **hosts.equiv** - a Berkeley r-segédpárogramok szolgáltatásait igénybe vehető megbízható felhasználókat/gazdagépeket definiálja
- **hosts.lpd** - a helyi nyomtató használatára engedélytel rendelkező számítógépeket sorolja fel
- **inetd.conf** - ez a konfigurációs fájl az inetd démon számára egy kapcsolat és egy adott port közötti megfeleltetést és az indítandó démont határozza meg
- **inittab** - ez a konfigurációs fájl az init processz számára a héjszkriptek hozzárendelését tartalmazza a futtatási szintekhez

- **issue** - ez a szöveges fájl a bejelentkezési prompt előtt jelenik meg; általában a gazdagép nevét és üdvözölő üzenetet tartalmaz
- **ld.so.cache** - a dinamikus szerkesztő-összekapcsoló program konfigurációs adatait tartalmazó fájl
- **ld.so.conf** - a közös könyvtárak elérési útvonalait tartalmazó fájl
- **lilo.conf** - konfigurációs fájl a Linux Loader (Linux betöltő, LILO) számára
- **magic** - ez a fájl a fájlok elején található bájtminta és a megfelelő fájltípus közötti összefüggést definiálja; így a fájltípus a **magic** fájlból a kezdő bájtminta keresésével meghatározható
- **motd** - ez a szöveges fájl (a nap üzenete) általában minden bejelentkezés után automatikusan megjelenik
- **mtab** - a mount belső táblázata, amely az összes jelenleg csatolt fájlrendszeret tartalmazza
- **mtools** - konfigurációs fájl az MTools számára
- **named.boot** - indítási fájl a névkiszolgáló számára
- **networks** - az ismert hálózatok IP címét és nevét tartalmazza
- **passwd** - az ebben a fájlból definált felhasználók azonosítása a felhasználói azonosítóval, a felhasználói névvel, a bejelentkezési adatokkal és az elsődleges csoporttal történik. A tényleges jelszó tárolása a **shadow** fájlból történik, amely a **passwd** fájllal ellentétben, csak **root** jogosultsággal olvasható.
- **printcap** - konfigurációs fájl a nyomtatási várolisták számára
- **profile** - globális profilfájl a Bourne héjprogramok (**bash**) számára
- **protocols** - a TCP/IP csomagtípusokat definiálja
- **resolv.conf** - konfigurációs fájl a TCP/IP rendszer számára, amely a helyi körzet nevét és a névkiszolgáló címét tartalmazza
- **rpc** - konfigurációs fájl az RPC kiszolgáló számára
- **securetty** - azt határozza meg, hogy a superuser mely TTY-kre jelentkezhet be
- **services** - a TCP/IP portok hozzárendelését tartalmazza a megfelelő szolgáltatásnevökhez (programokhoz)
- **shadow** - kódolt felhasználói jelszókat tartalmazó fájl
- **shells** - ez a fájl az érvényes rendszerhéjprogramokat tartalmazza. Amikor a felhasználó anonymous vagy **ftp** néven jelentkezik be, az FTP démon ellenőrzi, hogy a felhasználó bejelentkező héj szerepel-e ebben a listában: ha nem, akkor megtagadja a bejelentkezést.
- **syslog.conf** - konfigurációs fájl a **syslog** démon számára
- **termcap** - konfigurációs fájl a **termcap** könyvtár számára, amely a különböző típusú terminálokhöz tartozó vezérlőkaraktereket definiálja
- **utmp**, **wtmp** - az összes felhasználó bejelentkezését és kijelentkezését tartalmazó naplófájl
- **xmmounttab** - konfigurációs fájl az **xmount** számára
- **xvmounttab** - konfigurációs fájl az **xvmount** számára

## AZ /ETC KÖNYVTÁRAK ÁTTEKINTÉSE

- **/etc/default** - a következő paraméterek alapértelmezett értékeit tárolja:
  - **useradd** - alapértelmezett értékek az **useradd** parancs számára
  - **getty.XXX** - a **getty** program konfigurációja az XXX portra
  - **uugetty.XXX** - az **uugetty** program konfigurációja az XXX portra
- **lilo** - a LILO telepítő programot tartalmazza
- **skel** - olyan fájlokat tartalmaz, amelyeket az **useradd** végrehajtásakor automatikusan egy új felhasználó home könyvtárába másol a rendszer
- **rc.d** - olyan szkripteket tartalmaz, amelyeket a megfelelő futási szintre való áttéréskor egy processz hív

- **rc.0** - szkript a 0-ás futási szint számára (rendszer lezárása)
- **rc.6** - szkript a 6-os futási szint számára (bejelentkezés az `xdm`-en keresztül)
- **rc.K** - szkript az egyfelhasználós üzemmód számára
- **rc.M** - szkript a többfelhasználós üzemmód számára (1...6-os futási szint)
- **rc.S** - betöltésnél a fájlrendszerek cseréjét és a konziszenciát ellenőrzését végrehajtó szkript
- **rc.keymap** - a nemzeti billentyűzetkiosztást betöltő szkript
- **rc.local** - helyi démonokat indító szkript
- **rc.inet1** - hálózati interféseket inicializáló szkript
- **rc.inet2** - hálózati démonokat indító szkript
- **rc.serial** - soros interfésekkel konfiguráló szkript

## A RENDSZERMAG KONFIGURÁLÁSA

(Az eredeti kötetben itt egy hosszú példa található a rendszermag konfigurálásáról. Ezt a példát kihagytam mert az újabb rendszermagok – a könyv fordításának ideje alatt elkészült legutolsó stabil kernel, a 2.0.21 is – konfigurálására a hagyományos parancssoros lehetőségen túl színes, karakteres és grafikus lehetőség is van. A `make menuconfig` utasítással színes, karakteres, menüs, a `make xconfig` utasítással X Window környezetben konfigurálhatjuk a rendszermagot. Ez a két lehetőség kényelmesebb, mint a korábbi, az eligazodást stúgó oldalak segítik – a lektor.)

## AJÁNLOTT IRODALOM

Andeleigh, Prabhat K.

*UNIX System Architecture*. Prentice-Hall (1993)

Carl-Mitchell, Smoot, and Quarterman, John S.

*Practical Internetworking with TCP/IP and UNIX*. Addison-Wesley (1993)

Comer, Douglas E.

*Internetworking with TCP/IP*, Vol. 1–3. Prentice-Hall (1991)

Flanagan, David

*X Toolkit Intrinsics Reference Manual*, Vol. 5. O'Reilly (1993)

Gilly, Daniel

*UNIX in a Nutshell*. O'Reilly (1992)

Gulbins, Jürgen

*UNIX*. Springer (1988)

Heller, Dan

*XView Programming Manual*, Vol. 7. O'Reilly (1991)

*Motif Programming Manual*, Vol. 6. O'Reilly (1992)

Hewlett-Packard

*Ultimate Guide to the Vi and Ex Text Editors*. Addison Wesley (1990)

Kehoe, Brendan P.  
*Zen and the Art of the Internet*. Prentice-Hall (1993)

Krol, Ed  
*The Whole Internet: User's Guide and Catalog*. O'Reilly (1993)

Lippmann, Stanley B.  
*C++ Primer*. Addison Wesley (1991)

Mui, Linda and Pearce, Eric  
*X Window System Administrator's Guide*, Vol. 8. O'Reilly (1993)

Nemeth E., Snyder G., Seebass S., Trent, R. H.  
*Unix System Administration Handbook*, 2nd Ed. Prentice-Hall (1995)

Nye, Adrian  
*Xlib Programming Manual*, Vol. 1. O'Reilly (1992)  
*Xlib Reference Manual*, Vol. 2. O'Reilly (1992)

Nye, Adrian, and O'Reilly, Tim  
*X Toolkit Intrinsics Programming Manual*, Vol. 4. O'Reilly (1990)

Open Software Foundation  
*OSF/Motif Programmer's Guide*. Prentice-Hall (1993)  
*OSF/Motif Programmer's Reference*. Prentice-Hall (1993)  
*OSF/Motif Users' Guide*. Prentice-Hall (1993)

Quercia, Valerie, and O'Reilly, Tim  
*X Window System User's Guide*, Vol. 3. O'Reilly (1991)

Schoonover, Michael A.  
*GNU Emacs-UNIX Text Editing and Programming*. Addison-Wesley (1992)

Stevens, W. Richard  
*Advanced Programming in the UNIX Environment*. Addison Wesley (1992)  
*UNIX Network Programming*. Prentice-Hall (1990)

Tanenbaum, Andrew S.  
*Distributed Open Systems*. Prentice-Hall (1995)

Young, Douglas A.  
*X Window System, Programming and Applications with Xt*. Prentice-Hall (1990)

# TÁRGY MUTATÓ

## A

Ablakkezelő konfigurálása 152  
Ablakkezelők 121  
Ada fájlok 289  
Ada9X 294  
Adatbázisok 257  
Adatcseré 33  
Adatsomagok 164  
Adatvesztés 43  
Adminisztrátori feladatok 99  
Adobe betűkészlet 251  
Aktív partíció visszaállítása 84  
Aktuális kernel (rendszermag) 83  
Alapértelmezett érték 108  
Alapértelmezett útvonal 167  
Algebrai rendszer 259  
Alias 17, 22  
Alkalmazási szint 164  
Alkalmazások 246  
Alkönyvtárak 15  
Alternatív boot manager 84  
Alternatív shelllek 36  
    bash shell 36  
    Bourne Again shell 37  
    tcsh shell 37  
Andrew multimédiás környezet 256  
Anonim hozzáférés 203  
Anónimusz FTP 109  
ANSI 295  
ANSI szabvány 258  
API konverter 56  
APL 293  
Áramkör-szimuláció 260  
Archie 204  
Archiválás 313  
Arena tallózó 237, 238  
Árnyalási technika 251  
Árnyék jelszó 87  
ARP 165  
    kérés 166  
    üzenet 165  
    válasz 165  
ARPA/NET 162  
ASCII 32, 248, 252, 253  
Asedit 248

AT és T 123  
Athena 118, 123, 247  
    elemgyűjtemény 123  
    projey 118  
    szövegelemek 247  
Átirányítás 19  
Átjáró 167  
Átvihető alkalmazások 124  
AUCTEX csomag 285  
AUIS (Andrew User Interface System) 256  
Automatikus hibajavítás 111  
Automatikus kapcsolatok 186  
Automatikus útvonal 36  
AWK 303  
axe 247  
Azonosítás 176

## B

Basic 293  
Befejezés 55  
Bejelentkezés/i/ (login) 11, 87  
    jog több számítógépre 205  
    könyvtár (home dir, login dir) 14  
    soros interfészen keresztül 96  
Belső elemek 120  
Berkeley foglalatok 178  
Berkeley r-szegédprogramok 205  
Betöltési konfiguráció módosítása 55  
Betöltési művelet kezelése 81  
Betölthető modulok 35  
Betöltő aktíválása 84  
Betöltő szektor 117  
Betöltőlemezek 69, 114  
    készítése 79  
Betűtipusok és színek 286  
Bézier görbék 250  
Billentyűhozzárendelés 264  
Billentyűtérkép 264  
Billentyűzet konfigurálása 87, 138  
BIOS 40, 41  
Bittérkép 119  
Biztonsági intézkedések 176  
Biztonsági másolat 207, 110, 263  
Boot manager 81  
Boot szektor telepítése 82

Borland 307  
 Bővítmények betöltése és aktivizálása 288  
 BSD (Berkeley Software Distribution) 8  
 BSD UNIX 205  
 Busz-rendszer (ISA/EISA/PCI) 67

**C**

C nyelv 293  
 fordítóprogram 64, 293, 295  
 news 234  
 program 308  
 shell 37  
 üzemmód 263  
 C++ 258  
 C++ fordítóprogram 293, 295  
 C++ interfész 124  
 C-szerű szintaxis 17  
 Cache 33  
 memória 103  
 CAD szoftverek 7  
 CAP (Columbia Appletalk Packet) 197  
 Card, Remy 32  
 CD disztribúciók 63  
 CD-előfizetések 63  
 CD-ROM 98  
 CD-ROM-meghajtók 66, 67  
 telepítése 67  
 Célcím 213, 214  
 Ciklusok 26  
 Címek 164  
 COBRA (IDL) 124  
 COFF 7, 57  
 Common Lisp 284, 295  
 Cons cellák 280  
 Cox, Alan 238  
 CPU 12  
 Crossfire 262

**CS**

Csatolás 15  
 számláló 15  
 körön 15  
 szimbolikus 15  
 Csatlánk 234  
 Csatlfájlok (swap files) 13, 86  
 Csomag mechanizmus (swap) 6  
 Csomagidelek helye 86  
 Csatelpartíció 65  
 létrehozása 76  
 CSLIP 160, 162  
 Csoporthoz 109  
 Csö-karakter 21

**D**

DECnet 119  
 deliver 209  
 Démon 12, 28, 29, 91, 180  
 cron 29  
 névkiszolgáló 180  
 nyomtató (lpd) 29, 92  
 program 12  
 protokoll 29  
 syslog 29  
 Device szakasz 132  
 Dinamikus hozzárendelés 175, 278  
 directors fájl 213  
 dirent 270  
 DNS névkiszolgáló 179  
 Dokumentáció 311  
 DOS, boot lemez 41  
 emulátor 7, 8, 40, 41, 47, 56, 197  
 partiók 33, 42  
 DPMI 40  
 DPMI 49  
 DVI fájl formátuma 253

**E**

E-mail 29, 163, 198  
 ediff 270  
 Egérrel való áthúzás 246  
 Egyedi, azonosító (user ID) 11  
 csomagok küldése 178  
 Egyéni függvények 298  
 Egyfelhasználós üzemmód 66  
 Egység és prioritás 92  
 Elágazások 24  
 Elektronikus dokumentáció 239  
 posta 91  
 Elemek attribútumai 121  
 Elengyőjtemény 120  
 Elérési útvonal 209  
 Elf 8, 57  
 elm 208  
 Elfre definiált függvények 279  
 Eltávolítható adathordozók 14  
 Eltérf azonosító 206  
 Emacs 19, 241, 265, 266, 269  
 dokumentáció 266  
 fajlnév 269  
 fájlkiterjesztés 269  
 indítás 265  
 keresés funkciója 266  
 kézikönyv 268  
 Lisp 263, 273  
 oktató 266  
 súgó 266

- szerkesztők 242, 304, 305, 308  
 telepítése 266  
 üzemmódok 268  
 változatok 248  
 EMS 40, 50  
 Emulátorok 7, 8, 40, 56, 57  
     dos 7, 8  
     iBCS2 55  
     windows 8  
 Erőforrásfájlok 121  
 Értelmezők 296  
 Eszközkezelő 13, 17  
     fájlok 98  
 Eszközök 16  
 /ETC fájlok 353  
 /ETC könyvtárak 354  
 Eszmecsere 233  
 Ethernet 162, 165, 167  
     cím 165  
     csomag 167  
     kártya 162
- F**
- Faces 286  
 Fájlnevek tartományának kiterjesztése 21  
 Fájlok cseréje 235  
 Fájlrendszerek 14, 85  
     kezelése 111  
     létrehozása 76  
 FAQ (Frequently Asked Questions) 238, 243  
     ismertetők 243  
     források 243  
 FAT 14  
 FAX 96  
 Fejlepcfájl 112  
 Feladó címe 208  
 Felhasználói, azonosító 108  
     hozzáférés 99  
     mód 13  
 Felhasználók és csoportok 108  
     tájékoztatása 110  
 Felületfejlesztők 301  
 File System Standard 104  
 Files szakasz 127  
 finder (kereső) 271  
 F6- és aleszközsám 16  
 Fogadás 96  
 Foglalatok (socket) 178  
 Font 251  
 Font-lock üzemmód 269  
 FOR, ciklus 26  
     parancs 26  
 Fordítás 88, 211
- Fordított aposztróf 21  
 Forgalomirányító 166  
 Források közé és célpártíció kiválasztása 77, 78  
 Forráskód 6, 88, 114  
 Források 64  
 Forródrót 239  
 Forth 293  
 Fortran 293  
 Fortran 295  
 F6- és alverziós szám 113  
 Free software 9  
 Fresco 124  
 Frissítések 112  
 FSF 9, 263  
     programcsomag 312  
 FTP (File Transfer Protocol) 203, 206  
     fájlok 164  
     kiszolgálók (szerverek) 5, 9, 41, 56, 62, 64, 65,  
         198, 203, 293  
     protokoll 203  
 Func-menu 272  
 Funkcionális nyelv 283  
 Futtatási szint 100  
 Függőleges szinkronizálás 136  
 Függvényhívás 275  
 Függvényjelzők 272
- G**
- GAWK 303  
 Gazdaság 179  
 GCC 112  
 Gépnevek 179  
 Ghostscript/Ghostview 251  
 Globális, beállítások 211, 284  
     változók 284, 277  
 GNAT 294  
 GNU 5, 36, 37, 241  
     AWK segédprogram 242  
     C fordító 6, 7, 242, 295  
     Emacs 9, 243, 247, 263, 286, 295, 304  
     hibakereső (GDB) 305, 310  
     kernel 9  
     sakkprogram 202, 261  
     tar parancs 110  
 GoodStuff ikontár 122  
 Gopher szolgáltatás 235  
 GPL 9  
 Grafikus, előfeldolgozó 204  
     kezelőfelület 11  
     programok 249  
 groff 252  
 GUI (grafical user interface) 118

**GY**

Gyors adatátvitel 207  
 Gyorsítókártyák 125  
 Gyökérkönyvtár 104, 112  
 Gyökérlemezek 69

**H**

Hajlékonylemez-meghajtók definíálása 53  
 Hálózatba kapcsolás alapjai 162  
 Hálózati, adminisztrátor 77  
   alkalmazások 198  
   átjátszóság 119  
   cím 166  
   démonok 198  
   forgalom 119  
   hardver 162  
   maszk 165, 166  
   osztályok 164  
   paraméter megadása 77  
 Hálózatkezelés 238  
 Hálózatmenedzsment-struktúra automatikus felismerése 238  
 Hang 35  
 Hangkártyák 67  
 Hardver 65  
 Héjprogram 311  
 Helyettesítő karakterek 21, 150  
 Helyi váltózok 278  
 Hibakeres 183  
 Hibakereső 44, 259  
 Hibáüzenet 91, 312  
 Hírcsoport 232, 242, 262  
 Hírolvasó 233  
 Hivatalos 179  
 HMA 40  
 HOWTO 238  
   ismertetők 243  
 Hozzáférési jogok 34  
 HP48 (X48) emulátor 58  
 HPGL 249  
 HTML (HyperText Markup Language) 236  
   3-as szabvány 238  
 HTTP (HyperText Transfer Protocol) 236  
   kiszolgáló 238  
 Hypercard 303

**I**

I-csomópont (I-node) 14, 32  
 I-node táblázat 103  
 I/O cím 70, 88  
 I/O portok 51  
 IBCS2 7, 8, 257

IBCS2 emulátor 55  
 IBM 3270 emulátor 61  
 ICCC 121  
 ICS (Internet Chess Server) 202  
 IDE, interfész 66  
   kontroller 66  
 Idézőjelek 24  
 Időzítő 44  
 idraw 250  
 IF 24  
 Image fájl 41, 53, 68, 104  
   létrehozása 41  
 Imake 307  
   fájl 312  
 IMAP2 protokoll 208, 209  
 Implicit szabály 306  
 Indítás 55  
 Indítóüzenet 44  
 INETD 199  
 Info, formátum 241  
   üzemmód 267, 268, 286  
 Ingres és Postgres 258  
 Ingyenes szoftver 204  
 INMOS 8  
 Input prompt 24  
 Inset 257  
 Integrált felületszerkesztő 124  
 Interaktív, függvények 282  
   használat 18  
   tervezés 301  
 Interfésznek nevei 167  
 Internet 6, 162, 163, 239  
   sakk-kiszolgáló 202  
   szolgáltatások 203  
   szuperverszerver (démon) 199  
 InterViews 124, 250  
 IP 164  
   címek 202, 216  
   cím dinamikus hozzárendelésének lehetősége 173  
 IPX interfész 55  
 Irányítási táblázat 164, 168  
 IRC (Internet Relay Chat) 234  
   kiszolgáló(szerver) 234  
   résznevő 234  
 ISDN kártyák 8, 67  
 ISODE 197

**J**

Játékprogramok 261  
 Jelölés 264  
 Jelszó (password) 11, 108  
   nélküli 205  
 joe 247  
 Just Logic SQL 258

**K**

Kapcsok (hook) 288  
 Kapcsolatok ellenőrzése 168  
 Karakter és blokkeszközök 16  
 Karaktereket kezelő függvények 281  
 Karakteroszatok kezelése 21  
 Képernyőfelbontás 100  
 Képfrissítési frekvencia 137  
 Keresési minták 27  
 Kereskedelmi disztribúciók 65  
 Keret 264  
 Kernel (rendszermag) 88, 114  
     forráskódja 88  
     image fájlok 83  
     kód 35  
     mód 13  
     monolitikus 35  
 Készrenáció jel 19  
 Kettős hozzáférés 43  
 Keyboard szakasz 128  
 Kézi beállítások 312  
 Kibocsátási megjegyzések 245  
 Kibontás 311  
 Kibővített parancsok 38, 39  
     gzip 39  
     GNU tar 38  
 Kihelyezett állomás 176  
 Kisztolgáló (szerver), telepítése 190  
     architektúrák 119  
     konfigurálása 190  
 Kódkarakter 282  
 Konfigurációs fájl 43, 189  
 Konfigurálás 43, 81, 85, 167, 174, 185, 283, 312  
     általános 85  
     rdev programmal 89  
 Konverterek 295  
 Konzol 91  
 Könyvtárak és segédletek 123  
 Környezet 18  
 Körzettel 179  
 Központi szkript 211  
 Kurzormozgatás és szövegfeldolgozás 281  
 Küldés 97  
 Különleges formák 275  
     használatára példák 276  
 Különleges ügyfélprogram 234

**L**

Lamport, Leslie 253  
 LAN 162  
 Lan manager 189  
 Laposzás (paning) 12  
 LaTeX 249, 253

LDP (Linux Documentation Project) 244  
 Leképező fájl 83  
 Levelel egyszerű továbbítása 211  
 Levelezés 207  
     automatikus rendezése 209  
 Levelezési, átjáró 217  
     listák 244  
 Levélolvásók 207  
     TCP/IP 207  
 LILO 33  
     Boot manager 33  
     eltávolítása 84  
 Linus Torvalds 5, 6  
 Linux, cserepartíció 76  
     disztribúciók 62  
     fájlrendszer 31  
     jellemzői 10  
     kezelőprogramok 188  
     könyvtárfa 104  
     Loader(LILO) 81, 83  
     mint X terminál 125  
     verziók 7  
 Linux parancsok 315  
     apropos 315  
     ar 315  
     at 315  
     atq 316  
     atrm 316  
     awk 316  
     basename 316  
     bash 317  
     batch 317  
     bc 317  
     cal 317  
     cat 318  
     cc 318  
     cd 318  
     chgrp 318  
     chmod 319  
     chown 319  
     cksum 319  
     clear 319  
     cmp 320  
     comp 320  
     compress 320  
     cp 320  
     cpio 321  
     crontab 322  
     csh 322  
     csplit 323  
     ctags 323  
     cut 324  
     date 324  
     dd 325  
     df 326

diff 326  
 diff3 327  
 dirname 327  
 du 327  
 echo 328  
 echo 328  
 ed 328  
 egrep 328  
 env 328  
 expr 328  
 false 329  
 fdformat 329  
 fgrep 329  
 file 329  
 find 329  
 finger 331  
 ftp 331  
 gcc 331  
 grep 331  
 groff 332  
 groups 332  
 gzip 332  
 head 333  
 hostname 333  
 id 333  
 join 333  
 kill 334  
 ksh 334  
 last 334  
 ld 335  
 ldd 335  
 lex 335  
 ln 335  
 lpc 335  
 lpq 336  
 lpr 336  
 lprm 336  
 ls 336  
 m4 337  
 mail 337  
 make 337  
 man 338  
 mesg 338  
 mkdir 338  
 more 338  
 mtools 339  
 mount 339  
 mv 341  
 nice 341  
 nm 341  
 nohup 341  
 nroff 341  
 openwin 341  
 passwd 342  
 pr 342  
 ps 342  
 pwd 342  
 rcp 343  
 rlogin 343  
 rm 343  
 rmdir 343  
 rsh 343  
 sdiff 344  
 sed 344  
 shutdown 344  
 sleep 344  
 sort 344  
 strings 345  
 strip 345  
 stty 345  
 su 345  
 tail 345  
 talk 346  
 tar 346  
 tee 347  
 telnet 347  
 test 347  
 time 348  
 touch 348  
 tr 349  
 troff 349  
 true 349  
 umask 349  
 uname 349  
 uncompress 349  
 uniq 350  
 uptime 350  
 uudecode 350  
 uuencode 350  
 vi 350  
 w 350  
 wc 350  
 whatis 351  
 which 351  
 who 351  
 write 351  
 xargs 351  
 zcat 351  
 Lisp 293, 295  
 értelmező 269, 274  
 szimbólumok 273  
 üzemmód 263  
 lispdir 271  
 Listakezelő függvények 280  
 Lokális változók 23

**M**

M4 makrófeldolgozó 152  
 MacDraw 250

Macintosh emulátor 61  
 telepítése 62  
 Make segédprogram 306  
 Make-fájl 302, 306, 312  
 Makrók 275  
 Man 239  
 Manual (kézikönyv) oldalak 252  
 formátuma 239  
 Manuális módosítás 114  
 Maple V 259  
 Maradék 213  
 Másolás merevlemezre 77  
 Matematikai alkalmazások 259  
 MBR 82, 99  
 MCC Interim disztribúció 63  
 Megszakítási szám 88  
 Memórialap 12  
 Merevlemez(ek) 66  
 elérése 51  
 Mesterséges intelligencia 295  
 Metacard 303  
 Metakarakterek 27  
 MIME (Multipurpose Internet Mail Extension) 208  
 MINIX, fájlrendszer 5, 31  
 klón 5  
 operációs rendszer 5  
 Mintaegezettetés 27  
 MIT IIS  
 MOCKA 295  
 Modem 184  
 Moderált, csoportok 232  
 híressport 242  
 Moderator 232  
 Modula-2, Modula-3 rendszer 294, 295  
 Monitor szakasz 129  
 Monolitikus kernelek 35  
 mountpoint 85  
 MPEG videolejátszó 252  
 MSQL 257  
 MTools 33, 41  
 elérési mód 33  
 MUD (Multi-User Dungeons) 262  
 Multimedias levelezőprogram 257  
 MuMail 209  
 Munkaadókkal 11  
 MuPAD programcsomag 259  
 Működtetés 81  
 MX rekordok 214, 217

**N**

NAG (Network Administration Guide) 244  
 Naplófájl 91, 92, 186  
 Nem moderált csoportok 232  
 Nemzeti karakterkészletek 93

Netscape tallózó 237  
 Névkiszolgálók 179, 180  
 News 231, 233  
 felépítés és telepítés 231  
 kiszolgálók (szerverek) 231, 233  
 telepítése 231, 233  
 NFS (Network File System) 188  
 NFS 206  
 NFS-en keresztül végrehajtott telepítés 77  
 NIS (Network Informatin System) 187  
 NNTP 202  
 protokoll 231  
 NOVELL 197  
 szerverek (kiszolgálók) elérése 55  
 Null nyomtatókábel 177

**NY**

Nyomtató démon 92  
 Nyomtatóhoz hozzáférés 54

**O**

Oberon rendszer 294  
 ObjectBuilder 301  
 Objective-C 9, 293, 295  
 ObjectLibrary 301  
 OBST 258  
 ODBC port 258  
 Oktatórendszer 263  
 OpenLook 123  
 Órajel-frekvencia 133  
 OS/2 33  
 boot manager 84  
 OSF/Motif 7, 121, 123, 302  
 Osztálykereső 248

**Ö**

Összeomlás 111

**P**

Paraméterek átadása kernelnek 90  
 Parancsmód 247  
 Parancsnevek rövidítése 22  
 Parancssorok áttekintése 30  
 Párhuzamos, feldolgozás 12  
 összeköttetés 177  
 Partíciók logikai beillesztése 78  
 Particíciók 73  
 Pascal 295  
 PC, hangkártyák 35  
 hangszóróinak elérése 52  
 PC/NFS 196

- Perifériák 67  
 Perl 304  
 Pine 208  
 PLIP 162, 177  
 POET 258  
 Pointer szakasz 128  
 Pont 264  
 Portok 178  
 Portsám 202, 236  
 Posix 5, 8  
 Postafiók-hálózat 6  
 Postafiókfájl 227  
 Postaládák (Mailbox) 65  
 PostScript 249, 251  
     szűrő 94  
 PPP 162, 173, 176  
     démon 174  
     kapcsolat megszakítása 175  
     ügyfelek 174  
 Prefixkódos ábrázolás 275  
 Primitív adattípusok 273  
 Prioritási szintek (nice level) 12  
 Processzek 12  
     közötti szinkronizáció és adatcsere 12  
 Program- és fájlkezelők 246  
 Programozási, nyelvök 293  
     stílus 283  
 Prolog 293, 295  
 Protokollok 184, 208, 209
- R**
- r puffer 264  
 RAM 65  
 rcp 206  
 RCS (Revision Control System) 306, 307  
 Readme fájlok 245  
 Referenciakártya 266  
 Reguláris kifejezések 27, 298  
 Release4 8  
 Rendszer, betöltése 99, 111  
     adminisztrátor 11, 99  
     konfigurációs adatok 187  
     leállítás 103  
     mag (kernel) konfigurálása 355  
 Resolver 216  
 Részleges függvény 267  
 RFC 162  
 Ritchie, Dennis 8  
 rlogin 206  
 Routers fájl 213  
 RPC (Remote Procedure Call) 187  
 RSA kódolás 237  
 rsh 207
- S**
- S3-as processzorkészlet 66  
 Samba 189  
 Sávszélesség 129  
 SCO 57  
 Screen szakasz 134  
 SCCS (Source Code Control System) 307  
 SCSI 66  
     eszközök 98  
     host adapter 68  
 sed 247  
 ServerFlags szakasz 127  
 Shadow fájl 108  
 Shell (héj, burok) 13, 17, 23, 109, 311  
     bejelentkezési 109  
     C 17  
     kibővített 17  
     modell 13  
     parancs 311  
     szabványos 17  
     szkript 23  
     UNIX 17  
 Simula 295  
 Slackware 62, 63, 67, 68  
     betöltő 68  
     disztribúció 62, 67  
     programcsomag 63  
 Slingshot 124  
 SLIP 162, 170  
 SLIP/PPP kiszolgáló 170  
 SLS disztribúció 62  
     smail és deliver 230  
     smail és UUCP 231  
     smail konfigurálása 211  
 Smalltalk 293  
 SMTP 202  
 SMTP protokoll 207  
 SNMP elemek adminisztrálása 238  
 SNNS (neurális hálózatok szimulátora) 260  
 Soros, csatlakoztatás 170  
     interfész 170  
 Stallman, Richard 9, 263  
 STONE 258  
 Streamer (sztrímer) 66, 98  
 Súgó függvények 267  
 SUIT 124  
 Sun Microsystems 123  
 Superuser hozzáférési jogosultság 206  
 Syslog démon 91  
 System V 8

**SZ**

Szabályok és hálózati etikett 233  
 Szekció 34  
 Szerkesztők 247  
 Szimbolikus csatolások 31, 113, 115  
 Szimulátorok 260  
 Szinkronizálási frekvencia 135  
 Szinkronizáló impulzus 137  
 Szkript szabványos kimenete 210  
 Softverek átvitele 311  
 Szöveg blokk 247  
 Szövegfeldolgozás 252  
 Szövegkonverzió 35  
 Szövegszerkesztő program 247  
 Szűrőcsomag 95  
 Szűrőprogram 92

**T**

Tallázó 205  
 Tárgykönyvtárak 57, 112  
 Tartomány 264  
 Távoli héjprogram 207  
 Távoli hibakeresés 305  
 Taylor UUCP 185  
 TCL (Tool Command Language) 295, 296, 297, 299  
 alkalmazások és kiterjesztések 299  
 függvényhívás 297  
 lista 296  
 nyelv 295, 296  
 szkriptek 299  
 utasítás 296  
 Tcl/Motif 301  
 TCP és UDP 178  
 TCP/IP 8, 119, 162, 238  
 Telepítés 57  
 alkalmazások 58  
 Telepítési eljárás 68  
 Telepítőcsomag 62  
 Telepítőprogram 62  
 Telnet 164, 198, 202  
 Terméktámogatás és súgó 239  
 Tetris 261  
 TeX 253  
 TeX fájl 312  
 Thompson, Ken 8  
 Tipusellenőrző függvények 279  
 Titkos adatok definiálása 176  
 Több nyomtatási sor nyilvántartása 94  
 Tömörített kernel 89  
 Töréspont 305, 310  
 Transports fájl 213  
 Transzputer-kártyák 67

TT 8  
 Turbo C 307  
 Turbo Pascal 307

**U**

UIT 124  
 Új csatornák 235  
 Új híresport 233  
 Újraindítási időtartam 187  
 UMB 40  
 Unifix disztribúció 63  
 UNIX 16  
 fejlesztések és szabványok 8  
 implementáció 31  
 környezet 7  
 operációs rendszer 18  
 parancsok 18, 23  
 rendszer 5, 11, 12, 13, 14, 293, 294  
 shellszkriptek 247  
 System V 123  
 szabványok 311  
 változatok 5  
 URL (Universal Resource Locator) 236  
 Usenet 231  
 /usr alatti könyvtár 107  
 Utválasztás 165  
 Útvonal 214  
 Útvonalbejegyzés 167  
 UUCP (UNIX to UNIX Copy program) 184, 214  
 protokoll 207

**Ü**

Ügyfél (kliens) 119, 195, 236  
 architektúrák 119  
 konfigurálása 195  
 Ütemező (scheduler) 12, 13, 88  
 Üzemmód 288  
 C 288  
 Emacs Lisp 288  
 inicializálása 288  
 Üzenetek cseréje 234

**V**

Változók 19, 23, 296  
 hozzárendelése 283  
 Vektorgrafikus ábrák 249  
 Vezérőlemelek 297  
 attribútumai 150  
 vi 247  
 Vibrálásmentesség 135  
 Videoadapter 48  
 Videokártyák 66

Videóüzemmód 129  
 beállítása 135  
 Virtuális, fájlrendszer 16  
 interfész 165  
 kapcsolat létrehozása 178  
 konzolok 31  
 lemez 114  
 temind 11, 202  
 Visszacsatoló eszköz 165  
 Vízzintes szinkronizálási frekvencia 137  
 VXP 302

**W**

WABI 56  
 Wais (Wide Area Information System)  
 szolgáltatás 245  
 kiszolgáló 245  
 WAM (Warren Abstract Machine) 295  
 Waterloo Software 259  
 While-ciklus 27  
 WINE 56  
 WordPrefect 57, 58  
 WWW (World Wide Web), W3 236, 301  
 elérése Mosaic tallózóval 237  
 kiszolgáló 236, 243  
 tallózó 236  
 WYSIWYG 253

**X**

X alkalmazások konfigurálása 148  
 X consortium 118

X erőforrások 121  
 X protokoll 119  
 szint 120  
 X Toolkit 120  
 X Window rendszer (X Window System) 6, 7, 118  
 felépítése 120  
 X designer 58  
 X/Open 8  
 X11 7, 8, 41, 47, 56, 63, 118, 209, 248,  
 263, 265, 301  
 konfigurálása 126  
 szerver (kiszolgáló) 125  
 X11R6 8  
 xcoral 248  
 xedit 247  
 XF86Config fájl 127  
 xfig 249  
 XFree86 118, 125  
 xgopher 236  
 Xia, Frank 32  
 xman 239, 241  
 XMS 50  
 xpaint 250  
 XPCE 295  
 XT könyvtár 120  
 Xteddy 261  
 xv 249  
 XView 124  
 XWPE 307

**Y**

YARD-SQL 258

# TARTALOM

## 1. FEJEZET BEVEZETÉS 5

1.1 A Linux történelmi előzményei .....	5	1.4 Unix fejlesztések és szabványok .....	16
1.2 Verziók .....	7	1.5 A Free Software Foundation .....	9
1.3 Lehetőségek .....	7	1.6 A Linux jellemzőinek áttekintése .....	10

## 2. FEJEZET ALAPFOGALMAK 11

2.1 Több felhasználó egyidejű kiszolgálása .....	11	2.6 Eszközök .....	16
2.2 Több feladat egyidejű végrehajtása .....	12	2.7 Shell .....	17
2.3 Memóriakezelés .....	12	2.8 Keresési minták .....	27
2.4 A shell modell .....	13	2.9 Démonok .....	28
2.5 Fájlrendszerek .....	14	2.10 A parancsok áttekintése .....	29

## 3. FEJEZET A LINUX JELLEMZŐI 31

3.1 Virtuális konzolok .....	31	3.5 Hang .....	35
3.2 A Linux fájlrendszer .....	31	3.6 Alternatív shelllek .....	36
3.3 Adatcsere .....	33	3.7 Kibővíthető parancsok .....	38
3.4 Betöltíthető modulok .....	35		

## 4. FEJEZET EMULÁTOROK 40

4.1 DOS emulátor .....	40	4.4 HP48 emulátor (X48) .....	58
4.2 WINE .....	56	4.5 Az IBM 3270 emulátor .....	61
4.3 Az iBCS2 emulátor .....	57	4.6 A Macintosh emulátor .....	61

## 5. FEJEZET TELEPÍTÉS 62

5.1 Linux disztribúciók .....	62	5.5 Betöltőlemez készítése .....	79
5.2 A források .....	64	5.6 A betöltési művelet kezelése (boot manager) .....	81
5.3 Hardver .....	65		
5.4 A telepítés .....	67		

## 6. FEJEZET KONFIGURÁLÁS 85

6.1 Általános konfigurálás .....	85	6.4 Bejelentkezés soros interfészen keresztül .....	96
6.2 Kernel .....	88	6.5 Fax .....	96
6.3 Démonok .....	91	6.6 Színtímek és CD-ROM .....	98

## 7. FEJEZET ADMINISZTRÁTORI FELADATOK 99

7.1 Az adminisztrátor .....	99	7.7 A felhasználók tájékoztatása .....	110
7.2 A rendszer betöltése .....	99	7.8 Biztonsági másolatok .....	110
7.3 Rendszerleállítás .....	103	7.9 Fájlrendszerek kezelése .....	111
7.4 A Linux könyvtára .....	104	7.10 Frissítések .....	112
7.5 Felhasználók és csoportok .....	108	7.11 Betöltőlemezek .....	114
7.6 Shelllek .....	109		

## 8. FEJEZET AZ X WINDOW SYSTEM 118

8.1 Jellemzők .....	118	8.6 Az X11 kiszolgáló (szerver) .....	125
8.2 A felépítés .....	120	8.7 A Linux mint X terminál .....	125
8.3 Az X erőforrások .....	121	8.8 Az X11 konfigurálása .....	126
8.4 Ablakkezelők .....	121	8.9 Az X alkalmazások konfigurálása .....	148
8.5 Könyvtárak és segédletek .....	123		

## 9. FEJEZET HÁLÓZATBA KAPCSOLÁS 162

9.1 A hálózati hardver .....	162	9.10 RPC .....	187
9.2 TCP/IP .....	162	9.11 NIS .....	187
9.3 IP .....	164	9.12 NFS .....	188
9.4 Soros csatlakoztatás .....	170	9.13 LAN Manager .....	189
9.5 PPP .....	173	9.14 PC/NFS .....	196
9.6 Párhuzamos összekötetés .....	177	9.15 Columbia Appletalk (CAP) .....	197
9.7 A TCP és az UDP .....	178	9.16 ISO/IEC .....	197
9.8 Gépcsevek .....	179	9.17 Novell .....	197
9.9 UUCP .....	184		

## 10. FEJEZET HÁLÓZATI ALKALMAZÁSOK 198

10.1 Hálózati démonok .....	198	10.7 Levelcélzás .....	207
10.2 Az Internet szuperverszerver (INETD) .....	199	10.8 News .....	231
10.3 Telnet .....	202	10.9 IRC .....	234
10.4 FTP .....	203	10.10 Gopher .....	235
10.5 Archie .....	204	10.11 World Wide Web .....	236
10.6 A Berkeley R-segédprogramok .....	205	10.12 Hálózatkezelés .....	237

## 11. FEJEZET TERMÉKTÁMOGATÁS ÉS SÚGÓ 239

11.1 Man, xman .....	239	11.5 WWW .....	243
11.2 Az Info formátum .....	241	11.6 levelezési listák .....	244
11.3 Hírcsoportok .....	242	11.7 Egyéb dokumentumok .....	244
11.8 FAQ és HOWTO ismertetők .....	243	11.8 Egyéb források .....	245

## 12. FEJEZET ALKALMAZÁSOK 246

12.1 Program- és fájlkészítők .....	246	12.6 Adatbázisok .....	257
12.2 Szerkeszők .....	247	12.7 Matematikai alkalmazások .....	259
12.3 Grafikus programok .....	249	12.8 Szimulátorok .....	260
12.4 Szövegfeldolgozás .....	252	12.9 Játékprogramok .....	261
12.5 Az Andrew multimédiás környezet .....	256		

## 13. FEJEZET A GNU EMACS 263

13.1 Áttekintés .....	263	13.5 Üzemmódotok .....	268
13.2 Alapfogalmak .....	264	13.6 Egyéb programok és bővítmények .....	269
13.3 Működtetés .....	265	13.7 Az Emacs Lisp .....	273
13.4 Dokumentáció és súgó .....	266	13.8 Konfigurálás .....	283

## 14. FEJEZET PROGRAMOZÁSI NYELVEK ÉS ESZKÖZÖK 293

14.1 Programozási nyelvek .....	293	14.10 Szerkesztők .....	304
14.2 C fordítóprogramok .....	295	14.11 A GNU hibakereső (GDB) .....	305
14.3 Pascal, Fortran, Simula és Modula-2 .....	295	14.12 A makró segédprogram .....	306
14.4 Lisp és Prolog .....	295	14.13 Imake .....	307
14.5 A Tcl nyelv .....	295	14.14 RCS .....	307
14.6 Felületekfejlesztők .....	301	14.15 Xwpe .....	307
14.7 Metacard .....	303	14.16 Példák .....	308
14.8 Awk, Gawk .....	303	14.17 Szoftverek átvitele .....	311
14.9 Perl .....	304		

## LINUX PARANCSOK 315

## FÜGGELÉK 353

Az /etc fájlok áttekintése .....	353	A rendszermag konfigurálása .....	355
Az /etc könyvtárak áttekintése .....	354	Ajánlott irodalom .....	355

..... 187  
..... 187  
..... 188  
..... 189  
..... 196  
..... 197  
..... 197

..... 207  
..... 231  
..... 234  
..... 235  
..... 236  
..... 237

..... 243  
..... 244  
..... 244  
..... 245

..... 257  
..... 259  
..... 260  
..... 261

268  
269  
273  
283

304  
305  
306  
307  
307  
307  
308  
308  
311

5  
5

**32 bites, többfelhasználós és több feladat párhuzamos végrehajtására képes UNIX-rendszer.** A Linux lehetővé teszi, hogy **több felhasználó** különböző programokat egyidejűleg hajtson végre, kihasználva az Intel 80386-os processzor, illetve az ezt követő processzorok képességeit. Az eredményként keletkező **teljesítmény** a klasszikus RISC munkaállomással már mindenkorban összehethető.

**Igazodás a leggyakoribb UNIX szabványokhoz (POSIX).** Mivel a Linux igazodik a meglevő UNIX szabványokhoz, a rendelkezésre álló szoftverek általában minden nehézség nélkül átvihetők Linux alá.

**Hálózati támogatás (TCP/IP).** Az a számítógép, amelyen a Linux fut, a TCP/IP hálózatba könnyen integrálható. A Linux támogatást biztosít a TCP/IP használatához a legismertebb Ethernet kártyákon, modemén (SLIPP/PPP) és ISDN-en keresztül.

**Grafikus felhasználói kezelőfelület (X11).** A Linux rendszer tartalmazza az X Window rendszer legújabb verzióját (Release 6), valamint a UNIX rendszerek szabványos felhasználói kezelőfelületét (**OSF/Motif**), megvásárolható.

**GNU segéd- és alkalmazói programok.** A Linux parancsai és segédprogramjai közül sok olyan van, amely ugyan a GNU projektből származik, azonban sok funkcionális bővítést tartalmaz.

**Teljes UNIX fejlesztői környezet.** A Linux alatt olyan programok fejleszthetők, amelyek problémamentesen futtathatók más UNIX rendszerek alatt is és a programozó munkáját a GNU C / C++ / Objective C fordítóprogramokon és számos szövegszerkesztő programon kívül egyéb szoftverfejlesztő eszközök is segítik.

---

### MEGJELENT:

---

KRIS JAMSA

JAVA



9 789630 938969

