Most akkor nyílt vagy szabad a Linux?

Nézzük meg közelebbről ezt a linuxos körben oly gyakorta parázs vitát kiváltó kérdéskört!

Bizonyára mindannyian hallottatok az idei konferenciáról, sőt, remélem, a legtöbben meg is látogattátok. Mi is kint voltunk és sokakkal találkoztunk. Először arra gondoltam, hogy itt majd a nap történéseiről, az előadásokról írok, hogy milyen sokat fejlődött a szakma és a többi, és a többi, de utánagondolva mégis lényegesen fontosabbnak tartom, hogy néhány alapvető dolgot tisztázzunk.

Számos esetben tapasztalom, hogy a linuxosok nem igazán képesek „egy zászló alatt” dolgozni. Itt van például a szerzői jogok kérdésköre. A legtöbben nem értik, mit is takar a szabad program fogalma, és akadnak, akik ugyan értik, de nagy lufinak hiszik. Néhányan pedig az egész mozgalmat egyenesen kártékony vírusnak tartják. Az elveket ismerők viszont ahelyett, hogy mindent megtennének az irányvonalak lényegének megértetése irányában, felesleges veszekedésekbe fektetnek nagy energiákat.

Pontosítok: aki érti, hogy *Richard Stallmann* miről beszél, az megérzi benne annak az iránynak a képviselőjét, ami a számítástechnikai és a programozó közösség számára is nagy lépést jelenthet előre. Mégis azt tapasztalom, hogy akik elindultak ezen az úton, marják egymást, mint a veszett kutyák. Hogyan is van ez? Miért alakul így? Hogy ezen elgondolkodhassunk, engedjétek meg, hogy röviden ismertessem (helyszűke miatt egy-egy témát gyakran leegyszerűsítve) a főbb alapelveket.

A forrással együtt terjesztett programok haszna

Elsőként tisztázzuk, miért jó, ha egy program forrását is megkapjuk. Egyrészt a segítségével a programozók kényelmesebben tudják az adott rendszert felhasználni (ha például egy vezérlőprogram forrása nyílt, az adott modult használó programozó könnyebben megérti a rendszer működését, megtalálja a gyengéit), másrészt a hozzáértő programozók számára nagyobb biztonságot jelent (ha veszik a fáradságot és végigböngészik a kódot, megnyugtathatják magukat, hogy a rendszer jó és nincs benne hátsó kapu stb.). Ez nem biztos módszer, hiszen a programozók van, hogy restek a gondos áttanulmányozásra, valamint az is megesett már, hogy a programozó úgy rejtett el hátsó ajtót, hogy a nyílt forrás ellenére sem lelték meg. A forrás további, jelen esetben számunkra mellékes előnyöket is tartalmaz (gépfüggő fordítás stb.).

A forráskód terjesztése azonban további kérdéseket vet fel. Miképpen tudom biztosítani, hogy amit én írtam, az enyém is marad, például ki véd meg attól, hogy egy másik programozó eléírja a nevét, és a sajátjaként forgalmazza, végezetül pedig hogyan lesz nekem ebből pénzem? Ezen kérdések megválaszolására más és más irányvonalak alakultak ki. Mivel számos létezik közülük, engedjétek meg, hogy három „lépcsőbe” soroljam őket (a besorolás önkényes):

Három új terjesztési modell

A kereskedelmi területhez legközelebbi változat szerint a kód elérhető és olvasható ugyan, de nem használható fel és nem másolható. Ennek főképpen oktatási haszna lehet, és csak azok a cégek szokták használni, amelyek biztosak benne, hogy rendszerüket a vásárló mindenképpen megveszi, illetve „úgysem tud belenyúlni” a kódba. Ebben az esetben igazából csak a vevő megnyugtatásáról van szó.

A második „lépcső”, a Nyíl Forráskód arra támaszkodik, hogy egyfelől fel tudja használni a már rendelkezésre álló kódrészleteket, programokat, másfelől biztosítani kívánja, hogy a programozó legalább egyszeri pénzt kapjon a munkájáért. Ha tehát valakit felkérnek, hogy készítsen egy robotgépvezérlő modult, azért munkabér gyanánt kap valamennyi pénzt. Amikor pedig a munka elkészül, a forrást nyílttá, mások által szabadon felhasználhatóvá teszi, így ha bárkinek egy ilyen gépet kell vezérelnie, elég, ha a kész modult „testreszabja”, és máris dolgavégezetten távozhat. Ebben az esetben a programozó lemond a jog birtoklásáról, és nem tud „extra profitra” szert tenni a program többszöri eladásából (bár senki sem tiltja neki, hogy bedobozolja és árusítsa).

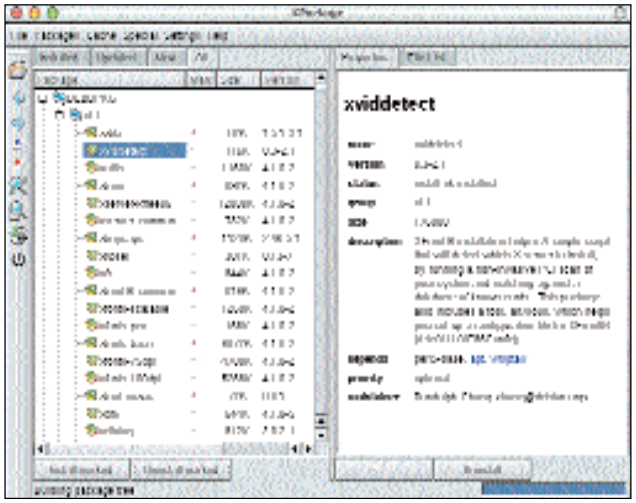
A szabad program (Free Software) irányvonal, Richard Stallman „áramlata” pedig a végletekig elmegy az ingyenességgel, a program ugyanis díjtalan, sőt már törvényes szigor mellett, kötelezően ingyenesnek is kell maradnia. Ha tehát írok egy modult, majd ezt a vonalat követve terjesztem, és valaki továbbfejleszti, ő is csak szigorúan ingyenesen adhatja tovább. Na jó, leegyszerűsítettem a fogalmakat, de első megközelítésnek megteszi.

Nem arra számítok, hogy mindenki azonnal megérti ezeket az új elveket, és mostantól ingyenesen dolgozik, hanem arra, hogy látni fogjuk, mennyire egy irányba haladunk ahhoz képest, amit az ellenpólus képvisel. Mivel én magam is úgy érzem, hogy ez az egyoldalnyi iromány édeskevés az elméletek és irányvonalak tényleges ismertetésére, a Linuxvilág következő számaiban többet foglalkozunk a hasonló kérdésekkel. Ha bárkinek véleménye, hozzászólása van, levelét szeretettel várom!

Szy György
a Linuxvilág főszerkesztője, a Kiskapu Kiadó vezetője. Mindenki véleményét és levelét örömmel várja az alábbi levélcímen:
Szy.Gyorgy@linuxvilag.hu

Programok telepítése Linux alá

Számos levelet kapok Olvasóinktól a CD-mellékleteken lévő programok telepítését illetően. Alapvetően két nagyon népszerű csomagformátum létezik, az egyik a RedHat által készített RPM, a másik pedig a Debian által életre keltett DEB. Mindkettő kifinomult függőségkezelést tartalmaz, így elég nehéz tönkretenni rendszerünket, de természetesen megfelelő kapcsolók segítségével erre is lehetőségünk van. Az RPM-alapú rendszereknél tehát parancssorból



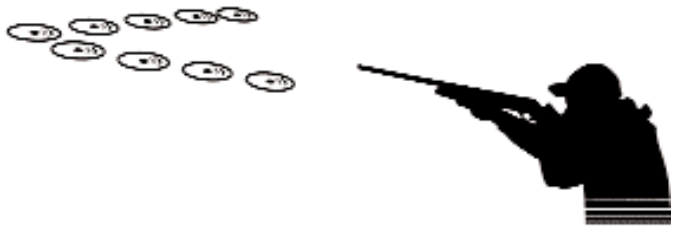
A KPackage

A csomag típusának kiválasztása

bármilyen csomagot az `rpm -i csomagneve.rpm` paranccsal telepíthetünk, DEB-alapú rendszernél pedig a `dpkg -i csomagnév.deb` paranccsal cselekedhetjük ugyanezt. Ezeknek a parancsoknak (`rpm`, `dpkg`) nagyon sokféle kapcsolójuk létezik, melyek segítségével csomagokat frissíthetünk, figyelmen kívül hagyhatjuk a függőségeket, lekérdezhetjük a telepített csomagok listáját, illetve tovább kereshetünk az adatbázisukban. Természetesen más módszer is létezik a programok telepítésére, kezdő felhasználók számára valószínűleg a grafikus csomagkezelő programok fogják a könnyed és fájdalommentes telepítést biztosítani, illetve az eltávolítás folyamatát kényelmessé tenni. Ilyen programokat minden RPM-alapú rendszer tartalmaz, a DEB-alapú rendszerek ugyancsak, a Debian Potato alapkiadás azonban nem. A Debian-rendszerek viszont egy nagyon hatékony csomagkezelő programmal bírnak, az `apt`-tal, ami képes az Internetről, CD-ről és bármilyen más forrásból csomagokat telepíteni.
A KPackage program valószínűleg a legtöbb rendszerben használható. Ez – mint neve is mutatja – szintén egy KDE-alkalmazás, képes kezelni például az `rpm`-, a `tgz`- és a `deb`-csomagokat. Könnyen használható grafikus felülete bizonyára a kezdő Debian-felhasználók számára sem nem fog akkora megrázkódtatást okozni, mint amekkorát a `dselect` szokott.

A kérdésekre legtöbbször a Linux-rendszerek sokfélesége miatt (gondolok itt a programkönyvtárak változataira, a rendszerhez tartozó programokra, a telepített csomagokra stb.) nem lehet mindenki számára megfelelő választ és egyben megoldást adni. A szerkesztőségben minden CD-re kerülő programot kipróbálunk, a gyakorlatban ez azt jelenti, hogy Debian Woody/SID és Mandrake 7.2 rendszerre telepítjük őket. Ezek elég friss programkönyvtárakat használnak, így valószínű, hogy az újabb Linux-változatokon a programok már kifogástalanul működnek. A játékoknál szinte előfeltétel a 3D-támogatás megléte és pontos beállítása, és mivel ez meglehetősen nagy feladat, nem is lehet telefonban vagy levélben elmagyarázni. A CD-hez tartozó két oldalon mindig megpróbálok kisebb segítséget nyújtani a programokhoz és telepítésükhöz, ezek azonban inkább csak az első lépés megtételéhez szükséges parancsok, a teljes telepítés során számtalan kérdés és előre nem látható esemény következhet be.
Mielőtt nekiláthatnánk a telepítésnek, számos fájlt előbb ki kell csomagolni. Ezek a csomagok általában `csomagnév.tar.gz` vagy `csomagnév.tgz` formában találhatók meg lemezmellékletünkön; némelyik közülük forráskódot is tartalmaz, amit le kell fordítani, némelyik azonban már a telepíthető és a telepítés után egyből futtatható programot foglalja magában. Kicsomagolásuk történhet a `tar zxvf csomagnév.tar.gz` paranccsal, ha viszont csak a gzip-tömörítést akarjuk kibontani, akkor a `gunzip csomagnév.tar.gz` paranccsal egy .TAR-állományt kapunk, amelyet a `tar xvf` paranccsal csomagolhatunk ki teljesen. Az egy adott programhoz tartozó programfordítás „művészete” is megtalálható néhány cikkünkben (ha forráskódból történő telepítés is szerepel benne).
A felhasználók gyakorlottságától függően a fentiek gondot is jelenthetnek, de az örök érvényű szabály itt is él: minél többet gyakorol az ember, annál könnyebben és nagyobb biztonsággal tud a Linux-rendszerben eligazodni.
Jó gyakorlatozást!

Programvadászat



Előző lapszámunk 15. számú CD-mellékletén a könyvtárszerkezet jogosultság-beállításába egy kis hiba csúszott, amit a felhasználó megnyithatatlan könyvtárakként észlel, rendszergazdaként azonban természetesen mind a könyvtárakat, mind a fájlokat meg lehet nyitni. Gyógyír lehet – azok számára, akik egyszerű felhasználóként szeretnének a fájlokhoz hozzáférni – rendszergazdaként futtatva az alábbi parancs:

```
mount /dev/xxx /yyy -o norock
```

ahol az xxx az eszköz neve, például hdc; az yyy pedig a könyvtár neve, ahová be szeretnénk fűzni a lemezt. Az okozott kellemetlenségekért Olvasóink szíves elnézését kérjük.

Irodára fel!

Jelenlegi CD-mellékletünk első lemezén Linuxra készített irodai programokból csemegézünk. Megtalálható ezek között, az OpenOffice sorozat legújabb tagja a KOffice 1.1-es változata, valamint sok-sok hibajavítással az AbiWord 0.9.2.

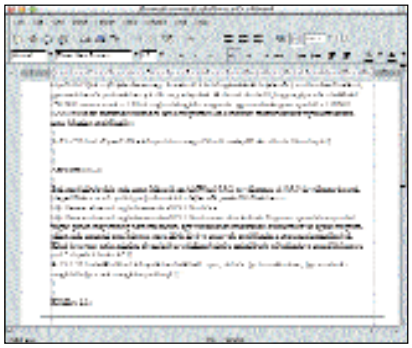
OpenOffice 638

Jó hír az OpenOffice-hívők számára, hogy az általuk kedvelt és használt irodai csomag sokat fejlődött. Az örömbe azonban egy kis üröm is vegyült, mert az ablakok méretének megjegyzésével sajnos még mindig gondjai akadnak. Ez azt jelenti, hogy az ablakot mindig újra kell méreteznünk ahhoz, hogy a tel-

jes képernyőfelületet használja. A régebbi változatokkal szemben az Excel-táblázatok megnyitásakor hibákat nem tapasztaltam, ami természetesen nem azt jelenti, hogy teljesen tökéletes, de az általam kipróbált fájlokat jól jelenítette meg. A weboldalak böngészésénél is jelentős javulást sikerült elérni, gyorsabban és pontosabban jeleníti meg a lapokat. Kicsit rémisztő viszont, hogy a gépemben található 256 MB memóriának a 33 százalékát rögtön lefoglalta, így ha ezt a programot sokat szeretnénk használni, mindenképpen 128 MB RAM az ajánlott (ez szerencsére a jelenlegi memóriaárakat figyelembe véve nem jelenthet nagy gondot). A 17. CD Iroda/OpenOffice könyvtárban a telepítő- és a forrásállomány is megtalálható.

AbiWord 0.9.2

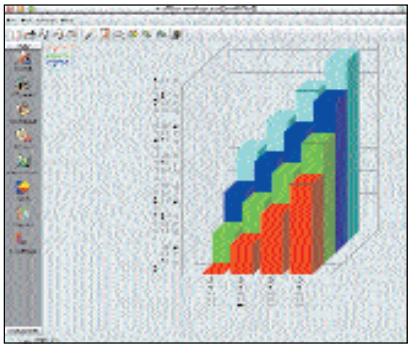
Sok apró hibajavítás után most felkerült az AbiWord 0.9.2-es változata. A 0.9.0-s változat óta a fejlesztők számos zavaró apróságot kijavítottak, amelyekről bővebben a ➲ http://www.abiword.org/release-notes/0.9.1.html és a ➲ http://www.abiword.org/release-notes/0.9.2.html címen olvashattok. A próbát végző gépen a nyomtatás azonban



még mindig nem működött, egy hibaüzenet kíséretében az egész program összeomlott, ekkor már menteni sem lehetett. Ezt a hibát leszámítva nem akadt gondom a program használatával. Kicsi és gyors, ezért minden olyan helyre ajánlható, ahol a számítógép teljesítménye nem felel meg a mai „alapelvárásoknak”.
A 17. CD Iroda/AbiWord könyvtárban található .RPM, .DEB és .TGZ formátumban, így mindenki megtalálhatja a neki megfelelő példányt.

KOffice 1.1

A KOffice a KDE-rendszerhez fejlesztett irodai programcsomag: minden olyan programot tartalmaz, amely a napi irodai munka elvégzéséhez szükséges. A rendszer része a KWord szövegszerkesztő, amely a .KWD (KWord), a .ABW (AbiWord), a .HTML, a .TXT és a .KPR (KPresenter) fájlokat egyaránt kezeli. Sajnálattal vettem észre, hogy a számomra legfontosabb formátumot, az rtf-et nem kezeli: sem olvasni, sem pedig írni nem tudja. Ennek ellenére kellemes felületű, könnyen használható szövegszerkesztő és az AbiWordhöz hasonlóan csekély memória- és erőforrásigényének köszönhetően kisebb teljesítményű gépeken is meglehetősen gyors. Ide tartozik még a Illustrator, egy olyan vektorgrafikus program, amely engem leginkább a korai CorelDraw-kra emlékeztetett. Egyszerű grafikák készítéséhez megfelelő, viszont a program elég gyakran összeomlott, így marad a jó



öreg módszer (biztos mindenki számára ismerős már valahonnan): egy lépés – egy mentés!
A Krayon a KOffice képfeldolgozó programja, nagyon emlékeztet a Gimpre és a Photoshopra is, de csak a Krayon-fájlokat tudja kezelni. A KChart segítségével diagrammokat készíthetünk, a KSpread pedig a csomag táblázatkezelője. A KFormulával matematikai képleteket szerkeszthetünk, a KPresenterrel viszont bemutatónkat készíthetjük el.
A programok között még egy irányítópult is található, ami a KOffice Workspace nevet kapta, segítségével egy helyről tudjuk indítani az alkalmazásokat.
A Koffice 1.1. a 17. CD Iroda/KOffice könyvtárában található, több népszerű Linux-változathoz.

Rendszermag

Mint mindig, most is felkerült a Linux lelkének legújabb, legfrissebb változata, ami jelenleg a 2.4.9-es változatot jelenti.
A változások teljes listáját a ➲ http://www.kernel.org/pub/linux/kernel/v2.4/ChangeLog-2.4.9 oldalon olvashatják.
A 17. CD Rendszermag könyvtárában található.

Opera 5.05

Augusztus 14-én vált elérhetővé az Opera böngésző legújabb változata, ami egyelőre még csak a próbaváltozat (technology preview 1). Már kezeli a legkülönfélébb Netscape-bővítményeket, így a Flasht, a Pluggert és a Java

1.2-t is. Ezzel ez a böngésző is elérkezett oda, hogy bizonyos oldalak és szolgáltatások igénybevételéhez már nem kell másik programot használnunk, hiszen mindent elintézhetünk belőle is. Ne feledkezzünk meg azonban arról, hogy még csak a próbaváltozat!
A 17. CD Opera könyvtárában található Debian-, RedHat-, és a mindenki által használható .TGZ formátumban.

Magazin

A 17-es CD Magazin könyvtára a cikkekhez tartozó programkódokat, forráskódokat és futtatható állományokat tartalmazza.

Linux teletanácskozás

A kodrészlet1.pl és kodreszlet2.pl fájlokat rejti.

Naplófájl színezése

A szinezes1.txt-t és a szinezes2.txt-t tartalmazza.

Pov-Ray ismeretek (3. rész)

A cikkhez tarozó képek forrásai és maguk a képek is megtalálhatók ebben a könyvtárban, néhány segédprogram kíséretében.

Bemutatkozik az Enhydra

A cikkben szereplő négy lista lelőhelye.

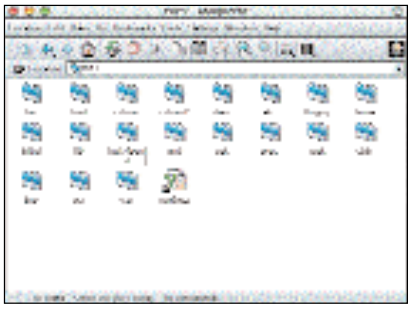
Postfix-csemegék (3. rész)

A Postfix-cikk két listája.

KDE 2.2

Ebben a hónapban második CD-mellékletünkön olvasóink a KDE 2.2 népszerű grafikus ablakkezelő rendszert találhatják. A rendszer méltán közkedvelt, mivel könnyen használ-

ható, és nagymértékben elősegíti a kezdők átállását a Microsoft operációs rendszereiről. Harmincnégy különböző nyelven érhető el (ezek között természetesen a magyar is megtalálható), így ezt a felületet a csak anyanyelvüket beszélők is sikerrel használhatják. Működik benne a „húzd és dobd” (Drag and Drop) módszer, amit akár nyomtatáshoz is használhatunk. Nagyon sok kiegészítőprogram megtalálható már a KDE-rendszerhez, és szinte minden megszokott programnak létezik már a KDE-s megfelelője. A felület



beállítását egy kellemes, egyszerűen használható beállítóprogrammal (Control-Panel) mindenki könnyedén megteheti. A rendszer részeit az alábbiakban ismertetjük.

Konqueror böngésző

A Konqueror a KDE 2 böngészője, fájlkezelője és dokumentumnézegetője. Összetevő-alapú felépítésének köszönhetően egyesíti magában az Internet Explorer/Netscape Communicator és a Windows Explorer tulajdonságait. Hamarosan az Interneten fellelhető összes szabványt támogatni fogja, például a JavaScript, a Java, a HTML 4.0, a CSS-1 és 2 (stíluslapok), az SSL és a Netscape Communicator bővítményeit (Flash, RealAudio, RealVideo lejátszásához).

KMail

A KDE levelezőprogramja a POP3-as fiókokat is kezeli. Azok pedig, akik egyszerre több postafiókkal is rendelkeznek, összes teendőjüket ebből a levelezőprogramból intézhetik, mivel egyszerre több fiókot is képes kezelni.

KDE játékok

Ez a csomag a játékos kedvű emberek szívét dobogtathatja meg. Ne számítsunk lövöldözős, véres, esetleg autóverseny-programokra, annál inkább aranyos logikai játékokra.
A játékok között találhatunk Mahjongg, kígyó-, póker- és aknakereső programokat.

KDE multimedia

Ebben a csomagban olyan eszközök rejlenek, amelyek segítségével különböző multimédiás fájlokat játszhatunk le, ez lehet például audio-CD vagy MPEG-film is.
A csomag részei többek között: Aktion filmlejátszó, kscd CD-lejátszó, KMix keverőpult, KMidi midilejátszó stb.

KDE graphics

A csomagban található programok használatával egyszerűbb ábrákat, ikonokat, fraktálokat, valamint képernyőmentéseket készíthetünk.

KDevelop

A felülethez tartozó fejlesztőkörnyezet, melyben könnyedén alkothatunk QT-eszközkészletre épülő grafikus felületeket.
Az Interneten nagyon sok hasznos forrást találhatunk a KDE rendszerhez, érdemes körülnézni a következő oldalakon:

- a KDE rendszerhez készült legújabb programokat és fejlesztéseket találhatjuk meg a ➲ http://apps.kde.org-on,
- a KDE rendszerhez készült irodai csomag honlapja a ➲ http://www.koffice.org, fájlkezelő honlapja a ➲ http://www.konqueror.org böngésző,
- a KDE fejlesztői felületének honlapja a ➲ http://www.kdevelop.org,
- a hibajelentések helye a ➲ http://bugs.kde.org,
- a KDE nemzetközivé tételével foglalkozó fő honlap (a KDE már 34 nyelven érhető el, természetesen magyarul is) a ➲ http://i18n.kde.org,
- témák a KDE-felülethez, ablakkezelőnket az itt található témákkal csinosíthatjuk: ➲ http://kde.themes.org,
- a KDE-multimédia oldala a ➲ http://multimedia.kde.org,
- a KDE-játékok oldala pedig a ➲ http://games.kde.org.

Csontos Gyula (Csontos.Gyula@linuxvilag.hu) a Linuxvilág szakmai, hír- és CD-szerkesztője. Szabadidejében szívesen mászik hegyet és kerékpározik.

Csúszik a FreeBSD 5.0

Jordan Hubbard augusztus 30-i bejelentése alapján a FreeBSD 5.0 tervezett megjelenési ideje 2002. november 1-je lett. Az eredeti időpont 2001 novembere volt, így a „nagy mű” még karácsony előtt elkészült volna, de sokan túl korainak találták a kibocsátást. A rendszerbe így be tudják építeni a *Julian Elischer* által készített KSE-foltokat is, melyek a következők:

- SMPng, következő nemzedékbeli többprocesszoros feldolgozás,
- KSE rendszermag-ütemező,
- teljes támogatás több új felületre, például a PowerPC-re, a SPARC64-re vagy az IA64-re (Itanium).

Ez a tizennégy hónap nyereség valószínűleg kiforrott, nagy teljesítményű és megbízható rendszert fog eredményezni.

A 4-es sorozat továbbra is változatlan ütemezéssel (igaz, kisebb-nagyobb csúszásokkal) jelenik meg. 2001. szeptember nyolcadikán látott napvilágot a 4.4-es változat.

➲ http://www.freebsd.org

Lapzárta után

Az IBM nyerte a Kormányzati Portál megvalósítására kiírt pályázatot

Az IBM nyerte meg a Kormányzati Portál első szakaszának megvalósítására kiírt pályázatot. Az IBM Magyarország Kft. olyan magyar tulajdonú cégekkel – Webigen Rt., IQSoft Rt., MÁV Informatika Kft. – együtt pályázott, amelyek szintén jelentős államigazgatási referenciákkal rendelkeznek, és az általuk vállalt feladatok területén már korábban elismerést vívtak ki. Egyes feladatok elvégzésére olyan sajátos területeken tapasztalatokat szerzett cégeket is bevontak, mint az Interface Kft.

Debianról BeOS-használóknak

Az alábbi címen a szerző a BeOS-használókat szeretné bevezetni a Debian/GNU Linux használatába, ezzel is segítve számukra az átállást. A telepítésen túl a ReiserFS, a biztonság, a frissítés, az X11, a KDE, a KOffice és más fontos, hasznos programok használatával és karbantartásával ismertet meg bennünket. Az az olvasó, aki szorgalmasan végigrágja magát ezen az olvasmányon, közelebbi kapcsolatba kerül a Debian különleges filozófiájával, és megszerezheti a kellő gyakorlatot a rendszer könnyed használatához. Minden linuxosnak a figyelmébe ajánlom (csak angol nyelven érhető el).

➲ http://tinyplanet.ca/pubs/debian/html/book1.html

VMWare 3.0

A VMWare 3.0-s próbaváltozata már elérhető és letölthető. A VMWare olyan virtuális számítógépet nyújtó program, melynek gazda operációs rendszere lehet Linux, Windows NT, 2000 és XP. Újdonságai a következők:

- 2.4.x rendszermag-támogatás egészen a 2.4.7-esig,
- USB-eszközök,
- DVD-ROM kezelése,
- CD-írók támogatása,
- SCSI-támogatás, lapolvasók, valamint szalagos egységek kezelése,
- 128 GB-os merevlemezrészeket támogat az IDE-felületen, SCSI-csatolón pedig 256 gigásat.

BedOS

Oroszországból érkezik az olcsó Windows?

A BedOS mindenképpen figyelemre méltó kezdeményezés. „A legjobb részeket foglalja magában a Windows 95, Memphis, Windows 98, Windows ME operációs rendszerekből. Ennek hatására olyan gyors, mint a Windows 95 és megbízhatóbb, mint a Windows ME. Teljesen kicserélték a multimédia-rendszert, melynek a Jet Audio az alapja. Telepítéséhez a gépünkön egy már meglévő Microsoft-rendszer szükséges, ami lehet például Windows 95 OSR2, Windows 98 vagy Windows ME. Mivel külön rendszerkönyvtárai vannak, a Windowszal nem zavarják egymást. Tartalmaz néhány Microsoft-összetevőt is. A telepítés után a Windows eltávolításakor is teljes értékű operációs rendszer áll s rendelkezésünkre. A BedOS letölthető shareware-ként, korlátozás nélkül használható, ötven dollárért pedig eltüntethetjük az indításkor megjelenő shareware-figyelmeztetést” – olvashatjuk a BedOS honlapján.

➲ http://winbeos98.km.ru

➲ ftp://ftp.km.ru/pub/v01/Soft/System/BedOS72214-1.exe

➲ ftp://sunsite.cnlab-switch.ch/mirror/winsite/win95/misc/BedOS72214-1.exe

➲ ftp://ftp.univie.ac.at/mirror/WinSite/win95/misc/BedOS72214-1.exe

Delphi 6

A Delphi 6 a korábbi Delphi-változatok előnyeinek és az eddigi irányvonalnak a megtartása mellett a legújabb fejlesztéseket használja fel. A Delphi 6 windowsos RAD-eszköz, amely legújabb jellemzőivel kiváló alapot nyújt e-Business-alkalmazások létrehozására. A Delphi 6 nyitott felépítéséhez új eszközök, új összetevők kapcsolódnak, és új operációs rendszerek használatára is lehetőség nyílik, melyek között természetesen a Linux is szerepel. A Delphi 6 az új webszolgáltatásokat és elosztott alkalmazásokat egyaránt támogatja, akárcsak az összes létező ipari szabványt (mint XML, SOAP, WSDL, XLS), valamint egyre nagyobb figyelmet szentel a webszolgáltatásokra épülő, más cégek által kifejlesztett felületek támogatására is, úgymint Microsoft .Net és BizTalk, továbbá Sun Microsystems ONE.

➲ http://www.borland.hu

Vásárlás előtti próba

Biztosan mindenki gondolt már arra, milyen jó lenne kipróbálni a kiszemelt programot, mielőtt a sokszor nem is kevés pénzt kiadnánk érte. Ehhez nyújt segítséget a

Runaware weboldala. Egy gyors feliratkozás után azonnal nekiláthatunk a kiválasztott program kipróbálásának. Találhatunk itt linuxos és windowsos programokat – az utóbbiak azonban sajnos sokkal nagyobb számban vannak jelen. A linuxos programok közül kipróbálhatjuk a Hancom Office, az Applixware 5.0, a Corel WordPerfect Office 2000 Linux irodai csomagokat, a Corel Photo-Paint 9 Linux és a CorelDRAW 9 for Linux grafikai programokat, az iceSculptor kiadványszerkesztőt és a Mentor Pro profi súgórendszer-készítőt (linuxos programokhoz). A szolgáltatás a 4-es sorozatú Netscape, illetve Explorer böngészőkkel használható.
Kalandra fel!

➲ http://www.runaware.com

StarOffice 6

A Sun Microsystems 2001. október elején tervezi kiadni a StarOffice 6 irodai csomag próbaváltozatát, 2002 első negyedében pedig a véglegeset. Alapját természetesen az OpenOffice képezi.
Aki az eseményről minél hamarabb értesülni szeretne, az látogasson el a következő címre

➲ http://www.sun.com/staroffice/beta6.html,

és iratkozzon fel a listára.
Az 5.2-eshez képest az OpenOffice-ban a már megszokott látvány vár majd bennünket: nem lesz beépített asztal (desktop), sem indítópult, továbbá az alkalmazások nem egy nagyméretű keretprogramból fognak futni, hanem mindegyik külön egységként lesz kezelhető, ezért erőforrásigényük is mérséklődik. Mivel az összes ismert dokumentumformátumot – ezen belül széles körben az ázsiai nyelveket is támogatja, a Sun cég ettől a szolgáltatástól várja termékének átfogó elterjedését. Linux, Solaris és Windows operációs rendszerekhez lesz elérhető.

➲ http://www.sun.com
➲ http://www.openoffice.org

A Novell és a Yahoo!

A Novell kibővítette kapcsolatait a Yahoo!-val annak érdekében, hogy a gyorsan bővülő vállalati portálpiacon a felhasználók számára újabb megoldásokat tudjon kínálni. A megállapodás részeként a Yahoo! a Corporate Yahoo! vállalati portálban az iPlanet Directory Server helyett a Novell eDirectory-t fogja címtárkiszolgálóként használni. Emellett a Novell maga is bővíti hálózati szolgáltatásait a Corporate Yahoo!-val, és a jövőben a két cég eljárásaira és újításaira épülő megoldásokat is szállít a Novell-vásárlók és -üzlettársak számára.

A Corporate Yahoo! vállalati portálmegoldásokkal a Yahoo! a munkahelyeken szerzett vezető szerepét aknázza ki, valamint azt a tapasztalatot, amelyet a személyre szabott, vonzó, a végfelhasználók számára nagy fontosságú adatokat összegyűjtő portálok terén gyűjtöttek. A Yahoo! a gyakorlatban is igazolta, hogy egyedi vállalati portálmegoldásai gyorsan telepíthetők és igen könnyen felügyelhetők. Mindez a szervezetek számára lehetővé teszi a termelékenység és az üzleti kapcsolattartás gyors javítását. A felhasználó személyazonosságának felügyeleti szolgáltatásait kínáló eDirectory segít a még pontosabb testreszabhatóságban, valamint leegyszerűsíti a különféle személyazonosságok, kapcsolatok és portálmegoldások felügyeletének kihívásait.

➲ http://corporate.yahoo.com
➲ http://my.yahoo.com
➲ http://www.novell.hu
➲ http://www.ctp.com
➲ http://www.novell.com/news/press/pressroom
➲ http://www.novell.com/competetive

Újra Corel Linux

A Corel nagy reményekkel vágott bele a linuxos fejlesztésekbe, lerázva magáról a kötelékeket, amelyek a termékeiket egy felülethez kötötték. Gőzerővel segítették a Wine fejlesztését, hogy az általuk készített programok és csomagok egy kis támogatással bármilyen Linux-változaton fussanak. Megjelent a Corel Linux OS első változata, majd szinte teljesen a feledésbe merült ez az ígéretesnek induló, asztali operációs rendszerként fejlesztett Linux. A pénzügyi gondokkal küszködő cég már-már feladni látszott ez irányú szárnypróbálgatásait, azonban szerencsénkre feltűnt egy másik kanadai cég, a Xandros, amely folytatná a Corel Linux fejlesztését. A 2001. augusztus 29-én San Franciscóban a Corel és a Xandros között létrejött szerződés értelmében a Xandros cég a következő 18 hónapban minden jogot birtokol a Corel-változat fejlesztéséhez.
A Xandros felhasználóbarát, könnyen telepíthető, egyszerűen használható Linux-változatot hoz majd létre, amely megfelelő lehet a Windows-rendszerek kiváltásához. Ám nemcsak az asztali rendszereket célozzák meg, hanem komolyabb alkalmazási területeken is (például ISP-k, vállalati rendszerek stb.) jelen szeretnének lenni. A Xandros a v3.0-s Corel OS-re építené asztali rendszerét. Az új rendszer tervezett kiadásának időpontja 2002 első negyedéve lesz.
Addig is együtt szorítunk nekik!

➲ http://www.xandros.net/
➲ http://www.corel.com
➲ http://linux.corel.com
➲ http://linux.corel.com/products/pp9/download.htm

Algoritmusok Afrikában – 1. rész

Elgondolkodtak-e azon, hogy az a nagy sietség, amivel az internetelérés lehetőségét az egész világon terjesztjük, mégsem szolgálja egyformán mindenki érdekeit? Tizenegy évvel ezelőtt számítógépet telepítettem egy szakmai továbbképző és fejlesztési központban Tutume-ben, Botswanában. Tutume kicsi, mezőgazdaságából élő falu, a Kgalagadi sivatag északkeleti szélén található, mely Afrika déli részén terül el.

A legszélsőségesebb eset: az „új gazdaság" hívői az Internetet tartják a megoldásnak. Mindenre. Az oktatástól az egészségügyi ellátáson és a környezetszennyezés kiküszöbölésén, valamint a társadalmi egyenlőtlenségek felszámolásán át a világbéke megteremtéséig.

A számítógép feladata itt az lett volna, hogy segítse a Tutume Brigádok – a központban működő szervezetek – munkáját. A cégek között akadt téglakészítő üzem, ácsszerszámokkal kereskedő üzlet, autójavító műhely, cirokmalom, iskolaiegyenruha-gyártó részleg, traktorkölcsönző és veteményeskert. A központ munkájának felmérése folyamán minden kétséget kizáróan megállapítottam: a számítógép elengedhetetlen eszköz a Brigád kusza, rosszul nyilvántartott pénzügyeinek rendbetételéhez, amelyeket egyébként nyolc különálló főkönyvben vezettek, és az elmaradás elérte a kilenc hónapot. A következő néhány hónap azzal telt el, hogy számítógépesítettem a nyilvántartást, és a helyieket betanítottam a rendszer alapfokú használatára. Az első év végére a központ pénzügyi nyilvántartása naprakész és pontos lett.

Ha nem mesélném tovább a történetet, hanem Kgalagadi csodálatos naplementéjére térnék át, úgy tűnhetne: ez igazi sikertörténet volt. Annak ellenére, hogy még az Internet- (és a Linux-) korszak fénykora előtti históriáról van szó, ma ezt is diadalként tálalhatnám: sikerült „áthidalnom a digitális szakadékot" Afrikában.

Az igazság az, hogy nem vagyok büszke erre a történetre. Ahogy teltek a hónapok Tutume-ben, rájöttem, hogy súlyos baklövést követtem el, komoly veszélynek téve ki a Brigádot. Olyan alapvető kérdéseket kellett feltennem magamnak, mint: mi lesz a számítógéppel, ha én már nem leszek itt? Képesek lesznek-e az itteni alkalmazottak önállóan működtetni a rendszert? Rendesen elkészítik-e majd a biztonsági mentéseket? Elég képzettek lesznek-e majd ahhoz, hogy leküzdjék a használat során felmerülő nehézségeket és újratelepítsék a rendszert, ha szükséges? Ha a számítógép tönkremegy vagy ellopják, lesz-e pénzük másikat venni? És mihez kezd a központ akkor, ha az általam gondosan felkészített alkalmazottakat a jó kereseti lehetőségek a nagyvárosba csalják el?

Szerencsére a Tutume-i Brigádok elkerülték ezt a kudarcot. Miután felmértem a veszélyt, a következő néhány hónapot azzal töltöttem, hogy a könyvelést visszaállítottam a régi kézi könyvvitelre, a személyzetet bővítettem és megtanítottam őket a helyes könyvviteli eljárásokra. Lehetővé tettem, hogy a számítógépet elsődleges segédeszközként alkalmazzák, ne pedig központi pénzügyi adatbázisként.

De mi közük van a tutume-i és lobatse-i történeteknek a Linuxhoz és a feltörekvő piacokhoz? A cikk hátralevő része ezt tárgyalja.

A digitális szakadék

Kilenc év telt el azóta, hogy elhagytam Botswanát. Az ezt követő pár évben többször jártam Afrikában, 1995-ben Linux-felhasználó lettem, 1997-ben nonprofit szervezeteknek kezdtem Linuxot telepíteni, majd 2000 májusában a nyugat-afrikai Guineában kötöttem ki. Valamikor eközben „találták fel" a digitális szakadékot. Úgy tűnik, a kifejezést először az Egyesült Államok Kereskedelmi Minisztériuma használta, a Nemzetközi Távközlés- és Információfelügyelet kiadta „Akik átesnek a hálón" című sorozat első részében. A jelentés az Egyesült Államok lakosságának távközléshez való hozzáférését elemezte, földrajzi és demográfiai szempontok alapján. Ebben többek között azt a következtetést vonta le, hogy az „információban gazdag" és „információban szegény" rétegek között egyre nagyobb távolság bontakozik ki.

Bill Clinton elnökségének utolsó évei alatt azt figyelhettük meg, hogy a digitális szakadék átlépte az USA határait és világméreteket öltött. A kérdéssel a nagy nyilvánosság az 1999-es G8-as csúcstalálkozó óta foglalkozik, ahol is a világ leghatalmasabb nemzetei eldöntötték, hogy az egyre szélesedő információs technológiai szakadék a Harmadik Világ fejlődésében a legjelentősebb gondok egyike.

Most, amikor ezt írom, a digitális szakadék áthidalása a legújabb irányzat, már ami a külföldi segítségnyújtást illeti, és nem egy segítségnyújtó szervezet és céget képviselő emberbarát ilyen irányú próbálkozása kapott nyilvánosságot. Nagyon leegyszerűsítve úgy tűnik, hogy az információs szakadékot az internetelérés hiányával azonosították. Léteznek tehát programjaink, mint például a USAID által pénzelt Leland-kezdeményezés, amelynek célja az afrikai internetelérés biztosítása; a Peace Corps, ami az AOL-lal közös kezdeményezés keretében népszerűsíti a számítástechnikát; valamint a nemrég alakult Geekcorps, mely már a második önkéntesekből álló csoportot küldi ki Accrába, a nyugat-afrikai Ghana fővárosába, hogy ott három hónapon át weblapokat tervezzenek. Természetesen a széles körű nyilvánosság jó lehetőséget biztosított a nemzetközi segítségnyújtó szervezeteknek, hiszen vonzó kampányokba foghattak a digitálisan rászorulók megsegítésére.

Következőkben annak taglalását folytatjuk, hogy az Internet valóban mindentudó varázsszer-e.

Az új technikai testamentum

A hálózat hittérítőinek célja az Internet csodájának megosztása. Miközben azon törik magukat, hogy az Internetet eljuttassák a fejlődő világhoz, elképzeléseikben nem mindig veszik kellőképpen figyelembe a műszaki adottságokat, és nem gondolják át, hogy a terv életképes, helyénvaló, értelmes vagy hosszú távon fenntart-

ható-e. Hallhattunk olyan közép-amerikai nőegyesületről, amely kézművestermékeit a Weben reklámozza; a jövő híveinek hangoztatása szerint óriási lehetőségek rejlenek a „távorvoslásban": segítségével virtuális egészségügyi ellátás nyújtható a ritkán lakott vidékeken élők számára; de hasonló jelenséget említ a US State Department Global Technology Corps is: „Láttunk olyan mexikói farmereket, akik az Internetet használták időjárás-előrejelzések letöltéséhez és a termés piaci árának felméréséhez". Ahol a norvégok régen gyapjúpulóvereket láttak, ott a műszaki fejlődés látnokai most webböngészőket.

A legszélsőségesebb eset: az „új gazdaság" hívői az Internetet tartják *a megoldásnak*. Mindenre. Az oktatástól az egészségügyi ellátáson és a környezetszennyezés kiküszöbölésén, valamint a társadalmi egyenlőtlenségek felszámolásán át a világbéke megteremtéséig. Mintha mindenki, akinek hálózati elérése van, feltölthetné magát a Nirvánába, vagy mintha a mennybe vezető út sávszélességgel lenne kikövezve. A szent kereszt helyett a digitális hittérítők jelvénye a parabolaantenna lett.

Az új vallás egyik alapköve az a hit, hogy az információ önmagában is elegendő a gazdaság működtetéséhez, a gondok orvoslásához és az egyenlőtlenségek leküzdéséhez. Ennek egyenes következménye az az üzenet, amely az egyik oldalról a másik felé tart: Mi birtokoljuk az információt és nem ti, a Mi információnk jó, a tiétek mit sem ér. Ezt szónokolja a CNN nemzetközi hallgatóságának, amikor azt állítja: „Az ember információ nélkül semmi".

Mára már világossá válhatott volna, hogy a szakadékról szóló szónoklatok csak újabb megnyilvánulásai a régóta ismerős, előítéleteken alapuló besorolásnak, annak, amely hosszú időkre két részre osztotta fel a világot: hívőkre és pogányokra, bölcsekre és vademberekre. A történelmi példákból okulva beláthatnánk, hogy e címkék gyakrabban ártottak a támogatásra szorulóknak, mint segítették őket.

Még fontosabb továbbá belátnunk, hogy a tájékoztatás és a számítástechnika elégségességébe vetett hit egyszerűen nem helytálló. Az információ önmagában nem segít az embereken. Ha ez igaz lenne, akkor az orvosok orvosi szakkönyvekből, a vállalkozók pedig pénzügyi szakkönyvekből lépnének elő.

Valójában a fejlődő világ tele van használaton kívüli röntgengépekkel, lerobbant traktorokkal, üres iskolapadokkal. Ezeket jó szándékú és buta adakozók pénzéből szerezték be a térség fejlődésének elősegítéséhez, és most nem azért használaton kívüliek, mert nincs hozzájuk leírás vagy nincs internetelérésük, hanem azért, mert nincs képzett technikus, szerelő és tanár. Röviden összefoglalva: igazából a tudás ad hatalmat az emberek kezébe.

Ezt mostanában már az Államokban is kezdik belátni. A „Hogyan töltsük meg az üres poharat – avagy a Digitális Szakadék korszerű filozófiája" (Educause Review, 2000. nov./dec.) c. könyvben *Solveig Singleton* és *Lucas Mast* a következőket írja: „A felsőoktatás álláspontja a következő: annak a diáknak, aki úgy hagyja el a középiskolát, hogy nem látott elektronikus tanulási segédeszközöket (például Internetet), sokkal kisebb gondjai lesznek, mint annak a tanulónak, aki úgy hagyja ott iskoláját, hogy nem tud megfelelően olvasni vagy számolni".

A vezető szabadpiaci kapitalizmust képviselő lap, a The Economist nemrégiben megállapította: „A szegény emberek nem azért kerülik el az Internetet, mert nem tudják megfizetni a használatát, hanem azért, mert nem rendelkeznek a hálózat hatékony használatához szükséges tudással." Nehéz belátni, hogy mitől megy majd jobban a szegényeknek, ha a Hálóra kötjük őket. Több értelme lenne, ha világméretű kampányt indítanánk az írástudatlanság ellen, mint annak, hogy a világméretű Hálózatot erőltessük.

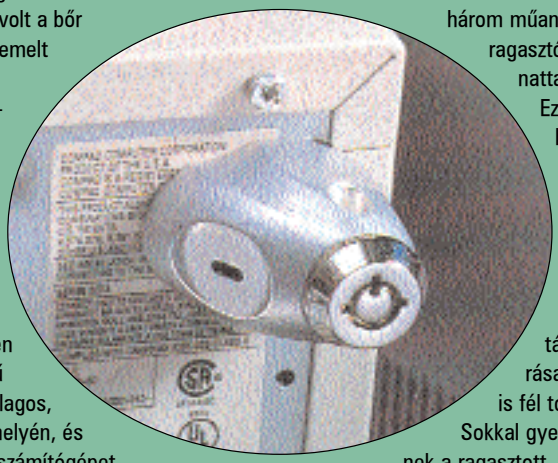
Elképzelhető, hogy a dot-com cégek bukásával és a tőzsdei mélyrepüléssel a digitális vallás tanai veszíteni fognak hitelességükből. Napjainkban, amikor a többmilliárdos internetszolgáltatók által alapított e-kereskedelmi csillagok – mint az Amazon és a Yahoo – Államok- és Európa-szerte nehézségekkel küzdenek, nehéz nem belegondolni abba, hogyan válhatnának valóra az e-kereskedelem ígéretei máshol, főleg olyan helyeken, ahol még rendes bank- és hitelrendszer sem működik. Számomra, aki az elmúlt évtizedben több évet éltem a fejlődő világban vidéki és városi környezetben egyaránt, ahol a lakosság nagy része még ma is tűzifával főz és vödörrel hordja a vizet, elmondhatom, hogy a külföldi segélynyújtó szervezeteknek a számítástechnika elterjesztésére tett kísérleteinek gyakorlati haszna elhanyagolható volt, ha nem egyenesen nevetséges. Mivel sürgősebb feladatok is akadnak a fejlődő világ nemzetei számára, így az egészségügyi ellátás biztosítása (AIDS, TBC, malária), az élelmezési gondok megoldása, a víz- és csatornarendszer kiépítése, az oktatás folyamatos biztosítása, a szegénység leküzdése, a környezetszennyezés megakadályozása, vagy a politikai korrupció leküzdése, a webböngészés lehetőségének biztosítása valószínűleg meglehetősen hátul foglal helyet az emberi szükségletek listáján. Lehetőségünk nyílik-e tehát arra, hogy tegyünk valamit, ami javíthat ezeknek az embereknek a mindennapjain? Játszhat-e a Linux és a nyílt forráskód meghatározó szerepet a születő piacokon? Van-e mód arra, hogy a technika segítségével emberi kérdéseket oldjunk meg olyan helyeken, mint Afrika, anélkül, hogy gyapjúpulóvereket próbálnánk eladni a sivatagban? Nem írnám ezt a cikket, ha nem lenne rá mód. Minderről bővebben következő lapszámunkban számolok be.

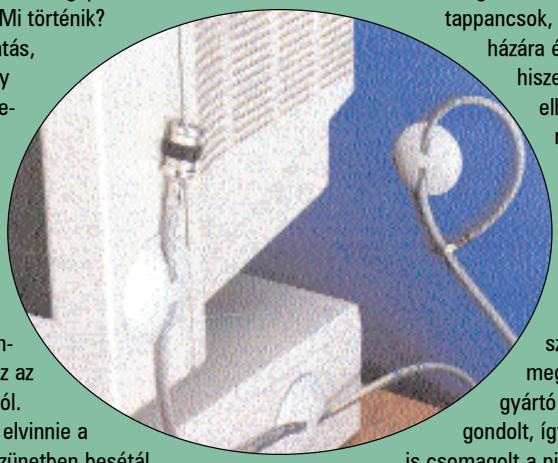
Wayne Marshall
(guinix@yahoo.com)
Unix-programozó és technikai tanácsadó. A nyugat-afrikai Guineában él.

Nehéz belátni, hogy mitől megy majd jobban a szegényeknek, ha a Hálóra kötjük őket. Több értelme lenne, ha világméretű kampányt indítanánk az írástudatlanság ellen, mint annak, hogy a világméretű Hálózatot erőltessük.

Kensington-próba

Felrémlik bennem az a történet, amit egy biztonsági előadásról olvastam néhány éve. A kalózok semmiféle különleges programot nem használtak, egy kis hibát sem találtak a megtámadott cég rendszerében – egyszerűen csak vastag volt a bőr a képükön. Felhívták a kiszemelt cég egyik alkalmazottját, közölték vele, hogy karbantartás miatt szükségük van az illető jelszavára, aki gyanútlanul megadta nekik – és máris szabadon, észrevétlenül mászkáltak a cég hálózatában. Gondoljunk csak bele: egy kellően elszánt, megfelelően öltözött, jó előadókészségű személy megjelenik egy átlagos, közepes méretű cég telephelyén, és közli, hogy el kell vinnie a számítógépet karbantartásra, bővítésre. Mi történik? Talán lesz némi telefonálgatás, de könnyen meglehet, hogy a kedves munkatárs jóhiszeműen és készségesen segít összecsomagolni a számítógépet, amivel főhősünk boldogan távozik. Otthon nincs is más dolga, mint működésre bírni a gépet, és máris fontos szerződéseket, számlákat, okiratokat kap kézhez az áldozatul esett vállalkozásról. Ha merészebb, nem is kell elvinnie a gépet, elegendő, ha ebédszünetben besétál az üresen hagyott irodába, és lemásolja vagy kiszereli, ami megtetszik neki. Adatokat lopni ugyanis nemcsak hálózaton keresztül, de az adathordozó eltulajdonításával is lehet. Ha nem is tárolunk létfontosságú vagy érzékeny, esetleg titkos adatokat, például egy eltűnt merevlemezen, a rajta található munkának, illetve magának az adathordozónak az értéke esetenként önmagában is komoly veszteséget jelenthet számunkra.





Ha a programok oldaláról nézve mindent megtettünk biztonságunk érdekében, fordítsuk figyelmünket a fizikai védelem felé! A Kensington több olyan terméket is kínál, amelyekkel megakadályozhatjuk a gép ellopását, illetve az adathordozókhoz való illetéktelen hozzáférést. A cég termékei közül ezúttal négy jutott el hozzám. Ha a teljes számítógépet megszeretnénk védeni, a Desktop Microsaver lehet a segítségünkre. A lapos dobozban rendkívül egyszerű felszerelést lelünk: három műanyag tappancsot, némi ragasztót és egy műanyag bevonattal ellátott drótkötelet. Ez utóbbi alkotja a csomag lelkét: az egyik végén fémkarikát, a másikon pedig egy kulcsos zárral rögzíthető fémhengert találunk. A drótkötél látványa önmagáért beszél, első pillantásra akár autóvontatásra is alkalmas, teherbírása hozzá nem értőként ítélve is fél tonna körül mozoghat. Sokkal gyengébb láncszemnek tűnhetnek a ragasztott – esetleg csavarozott – tappancsok, amelyeket a számítógép házára és a monitorra rögzíthetünk, hiszen a ragasztást könnyű elhibázni. A forgalmazó cég munkatársa ehhez csak annyit fűzött hozzá: még nem nagyon hallott olyanról, hogy valaki a ragasztott rögzítőelemeket leszedte volna… A ragasztási művelethez szükséges hozzávalók mind megtalálhatók a dobozban, a gyártó még a felület tisztítására is gondolt, így egy apró alkoholos párnát is csomagolt a pillanatragasztó mellé. Ha a pár perces ragasztással végeztünk, nincs más dolgunk, mint a drótkötelet a tappancsokon átfűzni, a gépet lehetőség szerint az asztalhoz, egy csőhöz vagy bármilyen más rögzített elemhez hozzáhurkolni, majd a túlsó végére illesztve a fémhengert csupán a kulcsot kell elfordítanunk, és már le is zártuk a gépet. Fontos megjegyzés, hogy a csomagban található két kulcs valódi és egyedi, nem pedig a régebbi PC-kről ismert billentyűzetlezáróhoz hasonló alkalmatosság.

Mennyire teremti meg a biztonságot a Desktop Microsaver? Természetesen a helyzettől függ. Ha éjjel behatolnak a cég telephelyére, kényelmesen elvághatják a drótkötelet, és elvihetik a számítógépet. Annak veszélye azonban elhárul, hogy a munkatársak szeme előtt,

esetleg közreműködésükkel történjen meg a rögzítőelemek leverése, elvágása és a gép elszállítása.
Egy számítógépet természetesen nem kell teljes egészében elszállítani, elég, ha a merevlemezt sikerül kiszerelni belőle. Ennek elkerülésére is akad megoldás, a MemoryLock. Segítségével nemcsak a merevlemezt, hanem a gép teljes belsejét is megvédhetjük az illetéktelen hozzáférőktől.
Az ötlet itt is rendkívül egyszerű. A védelmi eszköz egy alig félmaroknyi fémzár, amely két részből áll. Az alsó, nagyobbik felét a számítógép házának hátuljára kell felcsavarozni, felhasználva a ház oldalát rögzítő – vagy a csomagban kapott – csavarok egyikét. A felső rész maga a zár, mely elfedve a csavarhoz való hozzáférést engedő üreget gyakorlatilag azt akadályozza meg, hogy eltávolítsuk a számítógépház oldalát rögzítő csavart, és lehúzzuk a ház oldalát. Amíg nem szerezzük meg a MemoryLock kulcsát, addig csak a ház durva felfeszítésével tudjuk elérni annak belső alkatrészeit. Megfelelő körülmények között egy feltörőnek ugyan ez sem jelenthet gondot, de a lopásra hajlamos munkatársak elől minden bizonnyal eredményesen zárhatjuk el a számítógépek belső szerveit.
A MemoryLock kulcsa már jobban hasonlít az említett billentyűzetzár kulcsához, de egyedi kivitelű. Peremén többféle mélységű és hosszúságú vágat található, a megfelelő irányú beillesztést pedig külön fémpöcök segíti.
A Desktop Microsaverhez hasonló védelmet kínál a The Leash, melynek már a neve is beszédes („póráz”). Elsősorban hordozható számítógépekhez használhatjuk, hiszen ezeken gyakorlatilag szabványnak számít a „Kensington lock”-aljzat. A hordozható gépek leírásait böngészve gyakran találkozhatunk a nevével. Ha netán olyan készüléket szeretnénk vele rögzíteni, mely nem ilyen aljzattal rendelkezik, egy kisebb műanyag lapot kell ráragasztanunk.
A The Leash hasonló drótkötéllel gazdagít bennünket, mint a Desktop Microsaver, de zár tekintetében eltér attól. A záró fémhengert a „pórázról” nem lehet eltávolítani, a lezárás művelete pedig csak egy apró fémpöcök elfordítását jelenti. A védelem lényege az, hogy ez a fémpöcök a hordozható számítógép oldalába nyúlik be, így ha valaki erőszakkal akarná kitörni, minden bizonnyal súlyos károkat okozna a gépben, legalábbis eladhatatlanná téve azt. A The Leash kulcsa a MemoryLockéhoz hasonló, csillagszerű képződmény, melynek egyediségéről nem vagyok meggyőződve. A The Leash számzáras változatban is kapható, így választhatunk, hogy a zár elvesztésével vagy a számkombináció elfelejtésével tesszük-e mozdíthatatlanná a saját gépünket.
Miután mindent bombabiztos módon lekötöztünk, gondolnunk kell arra a bosszantó őskövületre is, amely még mindig ott lapul minden gépben: a hajlékonylemezes meghajtóra. Kapacitása kicsi, működését a mai viszonyok közt akár kezdetlegesnek is nevezhetjük, mégis kellemetlen lehet, ha valaki a megfelelő állományokat egyetlen gyors mozdulattal átmásolja lemezre, majd angolosan távozik.
Mindez másnak is eszébe jutott, ezért megszületett a FloppyLock. A kérdés adott, a megoldás most is lenyűgözően egyszerű: ha valami elfoglalja a lemez helyét, nem tudjuk használni a meghajtót. Ez a valami legyen mondjuk egy műanyag lap zárral a végén, ami hajlított végű fémdrótot forgat. A fémpöcök feladata az, hogy a meghajtó belsejében fennakadva megakadályozza a műanyag lap kihúzását. Joggal vetődik fel a kérdés: ha figyelmetlenül használjuk a FloppyLockot, nem tehetünk-e kárt a meghajtóban, illetve nem lehet-e egyszerűen kitépni a meghajtóból? Egy szétszerelt meghajtóval könnyedén meggyőződhetünk a FloppyLock biztonságos voltáról. A fémpöcök olyan helyre akad be, ahol semmilyen kárt nem okozhat, ha viszont ki akarjuk tépni, akkor egy, a meghajtónak csaknem teljes hosszában végighúzódó belső fémlemezzel kell megküzdenünk, ami a meghajtó tönkretétele nélkül lehetetlen. Mivel a hajlékonylemezes meghajtó íróolvasófejei nyugalmi állapotban a készülék hátsó részében parkolnak, ezeket sem sérthetjük fel, illetve az alul található motorban sem okozhatunk kárt. A FloppyLock minden szempontból biztonságos, kulcsa a MemoryLockéval megegyező kiképzésű. Természetesen adatot másolni CD-íróval, USB-s eszközzel, a különféle kapukon keresztül is lehetséges, de a FloppyLock segítségével a legkézenfekvőbb támadási pontot eredményesen védelmezhetjük.

Érdekesség, hogy külön kérésre a gyártó több terméket is készít azonos kulccsal, illetve olyan kombinációk is előállíthatók, ahol egy főkulccsal minden zár kinyitható, de az egyes alkalmazotti kulcsok csak egy-egy zárhoz használhatók.
A Kensington biztonsági termékeit hazánkban az L-Sys Kft. forgalmazza. A gyártó a termékekre élettartam-garanciát ad. Érdemes a Kensington vagy az L-Sys honlapján nézelődni, hiszen a cég további biztonsági termékeket is gyárt, illetve forgalmaz.
A MemoryLock hozzávetőlegesen 5000, a FloppyLock 4500, a The Leash 12 500, a Desktop Microsaver pedig 10 000 forint körüli áron szerezhető be.

➲ www.l-sys.hu
➲ www.kensington.com

Medgyesi Zoltán
(mzx@axelero.hu)
A BMGE 23 éves informatika szakos hallgatója. Szabadidejét legszívesebben barátnőjével tölti. Szeret autózni, és bográcsban főzni. A Linuxot öt éve ismeri, de még nem volt lelkiereje, hogy áttérjen rá.

Lásd a különbséget! – ViewSonic P225f monitor

„22 hüvelykes" madarak

Azt hiszem, egy ilyen monitor minden számítógépes grafikus, kiadványszerkesztő és CAD/CAM-felhasználó álma. A ViewSonic márkanév sokak számára a legmegfelelőbb megoldást jelenti. Ezt többek között az is bizonyítja, hogy az észak-amerikai államokban a húsz, huszonegy és huszonkét hüvelykes monitorok kategóriájában az utóbbi években ViewSonicokból adtak el a legtöbbet, megelőzve ezzel mind a Sonyt, mind a NEC-et.

Szerkesztőségünkbe a Pixel Multimédia Kft. jóvoltából jutott el a profi CRT-képernyők legékesebb darabja, a ViewSonic P225f típus.

Miután kifordítottuk hatalmas kartondobozából és „jókora" helyet teremtettünk neki az asztalomon, megkezdődhetett a kipróbálása.

A monitor szögletes formáival a kilencvenes évek elejét idézi. Elöl összesen öt nyomógomb található (1, le, fel, 2 és a bekapcsolás), amelyek tökéletesen illeszkednek a képernyő alján végigfutó csíkba. Itt található még a kombinált állapotjelző LED, amely szokásosan zöld színnel jelzi a megfelelő videojelet, narancssárgával pedig a videojel hiányát, illetve nem megfelelő voltát.

Amint csatlakoztattuk a kábeleket és elindítottuk a számítógépet, nem mindennapi élményben lehet részünk: a 22 hüvelykes monitoron – melyből húsz hüvelyk a látható felület – fejmozgatással kísérve kell átnéznünk a jobb felső sarokból a bal alsóba. A 0,24 mm-es résmaszk kiemelkedő képélességet biztosít. A monitor legnagyobb felbontása 2048×1536, amely 79 Hz-en működik. Mi az 1600×1200-as felbontást találtuk a legkellemesebbnek, ami 100 Hz-en is kiválóan működött, bár először megijedtünk tőle. A képcső négy sarka ugyanis homályos volt, de ezt kis idő múlva sikerült kiigazítanunk az OSD menürendszer segítségével. Akadt közülünk olyan is, aki az 1280×1024-es felbontásra esküdött (115 Hz-en), mivel így sokkalta nagyobbnak látszódott minden.

A behemót dobozban a használati utasítás mellett még egy CD-ROM-ra bukkantunk, amelyről feltelepíthettük a színbeállító programot. A színkalibrálás rendkívül egyszerű módon történt: egy próba és egy kis kék papírlapocska segítségével, amelyet a képernyőre illesztettünk, majd egy csúszkával hozzáigazítottuk a monitor színét.

A korongról egy további érdekes segédprogramot telepítettünk gépünkre: az „E-Color's True Internet Color and Colorific"-et, amelynek segítségével, ha E-Corrected (E-korrigált) szabvány által készült honlapokat látogatunk, az internetoldalt tökéletes színeiben láthatjuk.

A P225f rendkívüli mértékben „szembarát", sugárzási értékei kitűnőek, és megfelel az igen szigorú TCO'99 szabványnak is. A 350 MHz-es videobemenet-sávszélesség biztos alapját képezi az éles képeknek nagy felbontásban és frekvencián egyaránt. Színvilágára talán azt lehetne mondani, hogy az eddigi legjobb azok közül, amit mi láttunk.

Aki komolyan foglalkozik a számítógépes grafika bármelyik ágával vagy kiadványszerkesztéssel, annak nagy képátmérőjű és igen jó minőségű monitorra van szüksége. Miért ne választhatnák épp a legjobbat?

Adatok

- 22 hüvelykes átmérő (ebből 20 látható),
- felbontások:
  2048×1536 79 Hz,
  1920×1440 84 Hz,
  1600×1200 99 Hz,
  1280×1024 115 Hz,

- tökéletesen sík képcső (PerfectFlat),
- 0,24 mm-es résmaszk,
- 350 MHz videobemenet,
- energiagazdálkodás: Meets Energy Star, VESA DPMS, TCO'99, Energy 2000,
- méret: 504 mm×477 mm×501 mm,
- tömeg: 31 kg,
- szinte minden munkaállomáshoz használható (PC, MAC, SUN stb.),
- 3 év garancia,
- ár: 299 990 Ft +áfa

Forgalmazó

név: Pixel Multimédia Kft.
telefon: 266-6059
fax: 318-6651
e-mail: info@pixel.hu
➲ http://www.viewsonic.com
➲ http://www.pixel.hu

Csernák Ákos (csernak.akos@linuxvilag.hu) a Linuxvilág számítógépes grafikusa. A DTP mellett rajong a 3D-s grafikáért is. Fennmaradó szabadidejében természetjáró, utazó, és hébe-hóba gitározgat.

A „pingvintényező”

A processzorok és az operációs rendszerek összetartoznak, de a mesélők – az olyanok, mint mi – külön szokták választani a szereplőket, még akkor is, ha olyan nagy kivételek akadnak, mint a Wintel és a PowerPC. Most hogy a 64 bites lapkák az Intelnél és az AMD-nél egyaránt megjelenőben vannak, a történet sokkalta inkább a két cég egymás elleni viadaláról szól és nem a fejlesztőkről, akik a CPU-kat működtető operációs rendszerekre írnak programokat.

Május 29-én az Intel hivatalosan bejelentette az Itanium (azaz az IA64) gyártását. Történik ez majdnem pontosan hét évvel azután, hogy az Intel és a Hewlet-Packard azt nyilatkozta: újfajta processzoron fognak közösen dolgozni, amely a HP laboratóriumaiban látja meg a napvilágot „PA-RISC Wide-Word” néven. A közös fejlesztés Merced kódnév alatt futott, igazi márkanevét csak később nyerte el. A Sun Microsystems múlt ősszel jelent meg a piacon UltraSPARC III 64 bites processzorával, de ez a linuxos közösségeket kevésbé izgatta fel, mint az Itanium vagy az AMD Sledgehammer, amelyet előreláthatóan egy év múlva dobnak piacra.

Az Itanium megérkezett. Ahogy *Linley Gwennap* véli: „A kiszolgálók terén mindig lassabb a változás, de most, az Itanium megvalósulásával az Intel uralma már csak idő kérdése.”

Melyik operációs rendszernek jut majd a főszerep ebben az uralomban? Az első sajtóközlemény az Intel Itanium weblapján így hangzik: „Megindul az Intel Itanium-alapú rendszereinek a gyártása”. Az operációs rendszerekről pedig a következőket írják:

„Négy operációs rendszer fogja támogatni az Itanium-alapú rendszereket: a Microsoft Windows-rendszerei (64-bit Edition* a munkaállomásokon és 64-bit Windows Advanced Server Limited Edition 2002* a kiszolgálókon), a Hewlett-Packard HP-UX 11i v1.5*-e, az IBM AIX-5L*-je és a Linux. A Caldera International, a RedHat, a SuSE Linux és a Turbolinux egyaránt tervezi a 64-bites Linux-változat kiadását.”

Május 29-én a SuSE bejelentette a SuSE Linux 7.2 for IA-64-et, a RedHat pedig a Red Hat Linux for the Itanium Processort. A következő napon a Turbolinux nyilvánosságra hozta a Turbolinux Operating System 7 for the Itanium kiadás megszületését.

Mindhárom esetben jelen időben hangzottak el az állítások. Amikor ezeket a sorokat írom, az Intel weblapján olvasható négy szalagcím közül három a Linuxról szól. A negyedik így hangzik: „A Microsoft leleplezi a 64-bites Windows-rendszerek terveit”.

A gépek terén az SGI bejelentette (úgyszintén május 29-én) a Silicon Graphics 750-es rendszert Linuxszal, ami a cég szerint „az első Itanium-alapú rendszer, amely Linuxot használ”. Az IBM azt állítja, hogy az NCSA-ban a világ második leggyorsabb linuxos számítógépét építi meg, amely Turbolinuxot futtató itaniumos gépekből áll majd össze. A telepítést még a nyáron elvégzik.

Sokakat illet köszönet, de lehet, hogy egy csapással el lehet intézni a hálálkodást, ha a Trillian-projekthez fordulunk (ma már LinuxIA64.org-nak hívják és a ➲ http://linuxia64.org/ weblapon érhető el). A Trillian 1999 májusában indult és összehozta a Calderát, a CERN-t, a Cygnust (jelenleg RedHat), a HP-t, az Intelt, a Linuxcare-t, az NEC-t, az SGI-t, a Turbolinuxot és a VA Linuxot. E cégek többségének ma már van mit felmutatniuk.

Manapság is csak nehezen képzelhető el, hogy az Itanium nagy sikere ne a „pingvintényező” figyelembevételének lenne köszönhető.

Doc Searls

**A csillagok bejegyzett márkaneveket jelölnek.*

Amikor ezeket a sorokat írom, az Intel weblapján olvasható négy szalagcím közül három a Linuxról szól.

Vörös valóság

Az NBA első hárompontos dobóversenyén *Larry Bird* az öltözőben végignézett ellenfelein, és a következő kérdést tette fel: – Ki indul a második helyért?

Ez a kérdés Amerikában, a Linux-változatok világában is. A RedHat Larry Birdhöz hasonló helyezést tudhat a magáénak már évek óta, a kérdés csupán az, hogy ki következik a bajnok után.

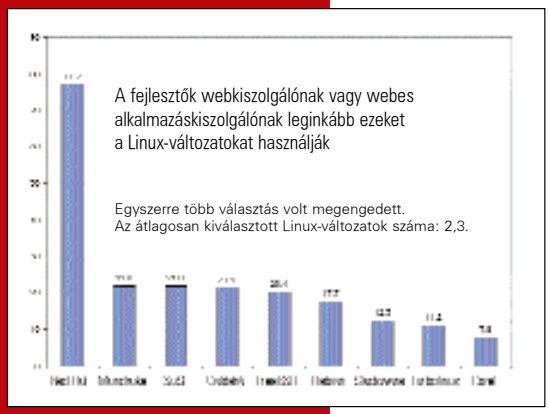
A közelmúltban a kaliforniai Santa Cruzban működő Evans Data Corporation munkatársai megkérdeztek háromszáz amerikai linuxos fejlesztőt, melyik Linux-változatot használnák szívesen a webkiszolgáló vagy a webes alkalmazáskiszolgáló feladatainak ellátására.

A válasz egyértelműen a RedHat volt. A második helyet holtversenyben a SuSE és a Mandrake foglalta el 21,8 százalékkal. Az ábra világosan mutatja, hogy a kérdés a fejlesztők számára valójában így hangzott: mit választanál a RedHat mellé?

Egy szavazó átlagosan 1,3 változatot jelölt meg második helyen (összesen tehát 2,3 változatra szavazott).

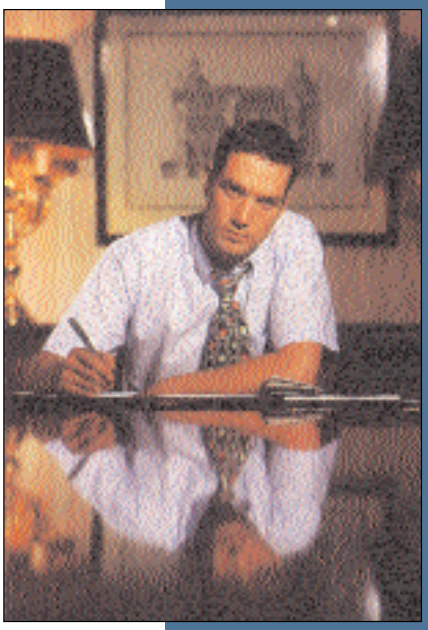
A Caldera, a Debian és a FreeBSD sem marad le sokkal a SuSE és a Madrake mögött.

A felmérés megtalálható az Evans weblapján ➲ http://www.evansdata.com/Linux01TOC.htm.



A Linux lekörözi a Solarist?

A pénzügyi szféra számára egyre jobb, olcsóbb és gyorsabb linuxos megoldások látnak napvilágot.



Nehéz dolog linuxos megoldásokat eladni a pénzügyi világban, ugyanis ez az a hagyományosan vaskalapos ágazat, mely a Nyílt Forráskód mozgalmát csak csigatempóban fogadja el. Mivel a folyamat lassú, jól megfigyelhető, hogy a pénzügyi szféra számára egyre jobb, olcsóbb és gyorsabb linuxos megoldások látnak napvilágot. Tavaly fejlesztette ki például a Reuters cég a „Value at Risk" eljárást, melynek segítségével statisztikai módszerekkel előre jelezhető, hogy adott pénzügyi helyzet esetén a befektetések mennyit veszíthetnek értékükből.

A termék motorjának céloperációs rendszere a Solaris 2.7 volt. A program elkészültével úgy döntöttünk, hogy a Linuxnak is adunk egy esélyt. A legtöbb harmadik fél által gyártott programkönyvtár létezett Linuxon, ami egyértelműen jelzi, hogy a Linux egyre nagyobb tekintélyt szerez a pénzügyi világban is, így a motort könnyen át lehetett ültetni Linuxra (2.2.14-es SMP rendszermagra, valamint a RedHat többprocesszoros változatára).

A pénzügyi motort Solaris 2.7-tel és négy 300 MHz-es processzorral felszerelt Sun 450-esen, Linuxszal pedig egy kétprocesszoros 733 MHz-es Pentium III-on próbáltuk ki. Az *1. táblázatban* láthatjuk a futás eredményét egy szokványos magánportfólió esetén (harmincnál több részvény).

1. táblázat A számítási teljesítmény összehasonlítása (futásidő) I.

	Linux	Solaris	Különbség
Hibakereső	1 mp	5,3 mp	530%
Javítptt	0,65 mp	1,8 mp	280%

Linuxon az egyszerűsített motort gcc 2.95.2.1-gyel és `-O3` kapcsolóval fordítottuk, Solarison pedig SPARCworks 5.0-val és `-xO3` kapcsolóval. A miénkhez hasonló pénzügyi motorok nem használnak sok I/O műveletet, inkább processzorigényes számításokat végeznek (a pénzügyi tulajdonságok a gyorstárban vannak). A program a legtöbb időt a pénzügyi eszközök feltételezett jövőbeli árainak kiszámításával és kezelésével tölti, ugyanis a pénzügyi algoritmus összeadásokat, szorzásokat és rendezéseket hajt végre. Ezek a próbák azt mutatták, hogy a Linux-Intel páros legalább kétszer olyan gyors, mint solarisos megfelelője. Ezt a programot nem arra hegyeztük ki, hogy arra nézvést szolgáljon bizonyítékkal, miszerint bármely rendszer lekörözi a másikat, hanem olyan feladatot hajtottunk rajta végre, amire a brókerek is jellemzően használják. A példák az életből származnak és szokványos portfóliókat vesznek alapul.

2. táblázat A számítási teljesítmény összehasonlítása (futásidő) II.

	Linux	Solaris	Különbség
Hibakereső	14 mp	48 mp	340%
Javított	7 mp	17 mp	240%

Az igazat megvallva, nem vártunk ekkora teljesítménynövekedést az áttéréstől. Legfőbb szempontunk a rendszer üzembe helyezési költségének mérséklése volt. Arra is számítottunk, hogy a költségcsökkenés a sebesség rovására fog menni, a próbafutások eredményei azonban meghökkentettek minket. Újabb ellenőrzésként a következő általános pénzügyi műveleteket mintázó algoritmust futtattuk le, és ugyanazt az eredményt kaptuk (lásd *2. táblázat*):

```
for(int j=0; j<10000; j++)
  {
    // idısor felØp tØse
    std::vector<double> ts;
    ts.reserve(2000);
    for(int i=0; i<2000; ++i)
    ➥ts.push_back(random());

    // vÆltozÆsok szimulÆlÆsa
    for(int i=0; i<2000; ++i)
    ➥ts[i]+=5.0;

    // idısorok rendezØse
    std::sort(ts.begin(), ts.end());
}
```

Bebizonyítottuk, hogy termékünk elkészítése során a Linux életképes választási lehetőség. Az ügyfelek megválaszthatják, hogy melyik rendszeren helyezzük üzembe a programot: Linuxon, Solarison vagy akár vegyes Linux-Solaris gépparkon, mivel a különböző típusú gépeken futó példányok gond nélkül együttműködnek. Tudtuk, hogy termékünk ezáltal olcsóbb lesz, és mára már azzal is tisztában vagyunk, hogy – legalábbis a mi programunkkal – gyorsabb is.

Választásunk egyértelműen a Linuxra esett, amivel a pénzügyi körökben is hozzájárultunk a szabad operációs rendszer terjesztéséhez, és ez nagyon jó dolog.

Sebastien Marc

Felhasználói szerződés kérdése

Na jó, beismerem, ezt a kérdést én magam írtam, mert olyan gyakran tették már fel, hogy a választ nyomtatásban szeretném látni. Nyílt forráskódot fejlesztő cégekkel és kezdeményezésekkel együttműködő jogászként gyakran kérnek fel arra, hogy az OSI által jóváhagyott nyílt forrással foglalkozó szerződések közül válasszak, vagy hagyjam jóvá ügyfelem választását. (Az itt bemutatott szerződések mindegyike megtalálható az OSI honlapján, a ➲ http://www.opensource.org címen.)

A kérdés azonban a sorrendet a feje tetejére állítja, ugyanis a szerződés kiválasztásának folyamatát az ügyfél üzleti terve vezérli, nem pedig fordítva.

A bevétel a program eladásából származzon vagy a kiegészítő szolgáltatásokból, mint például a telepítés vagy a betanítás? Ha az üzleti modell megkívánja, a kereskedelmi szerződések használatát semmi sem tiltja. A nyílt forráskód szószólójaként az ügyfelet természetesen megpróbálnám meggyőzni a nem kereskedelmi jellegű szerződések előnyeiről, de a végső döntés az övé.

Milyen mértékű szabadságot szeretnénk adni a program módosítására? Léteznek olyan nyílt forrásról szóló szerződések (például a BSD-típusúak), amelyek a felhasználókat szinte semmiben sem korlátozzák; ilyen felhasználói szerződés mellett a programot módosíthatják, és kereskedelmi változatokat is korlátozás nélkül hozhatnak létre. Más nyílt forrású szerződések (például a GPL-típusú, közismertebb nevén „szabad szoftver"-szerződések) megkövetelik, hogy a felhasználó változtatásaira ugyanazon szerződés feltételei vonatkozzanak. Ez az „öröklődés" akkor előnyös tulajdonság, ha az a szándékunk, hogy a közösség számára végzett munkánkhoz a felhasználók is hozzájáruljanak, amelyből így ők is részesednek. Akadnak olyan nyílt forrású szerződések (például az MPL-típusúak), amelyek a szabadság közbenső fokát nyújtják: a nyílt forráskódot tartalmazó állományok módosításai a nyílt forráskódú szerződés feltételeit öröklik, de az olyan új programállományok, amelyek csak használják azokat, másfajta szerződéssel is terjeszthetők.

Szándékunkban áll-e jótállni azért, hogy a program „meghatározott célra alkalmas"? Ha a program jogdíjmentes, akkor valószínűleg a garanciavállalást nem engedhetjük meg magunknak. Másrészt dönthetünk úgy is, hogy a nyílt forráskódú programot pénzért terjesztjük, és a biztosítékot, valamint az egyéb szolgáltatásokat a bevételből fedezzük.

Lehetséges, hogy a programunk annyira ismert, hogy a legfőbb, védelmet igénylő vagyontárgy maga a termék neve, nem pedig a kódja. Kitűnő példa erre az Apache. Felhasználói szerződésük megengedi, hogy az Apache-kóddal szinte bármit megtegyünk, de a módosított kód programnevét meg kell változtatnunk. Ha létezik olyan márkanevünk, amelyet védeni kívánunk, győződjünk meg róla, hogy a szerződés tartalmazza-e az erre vonatkozó megfelelő feltételeket.

Számításba vehetjük a kettős szerződés lehetőségét is: a programok szerzői jogának tulajdonosai számára mindig nyílt útként kínálkozik, hogy többféle felhasználói szerződést használjanak. A programot terjeszthetjük például a GPL feltételei szerint, felajánlva emellett egy kereskedelmi változatot azon vásárlók számára, akik félnek a GPL öröklődő tulajdonságaitól – így ez az indokolatlan félelem bevételi forrássá válhat.

Melyik nyílt forrású felhasználói szerződést használjam?

Eltérő felhasználói szerződéseket használhatunk a program különböző részeihez is: miközben a kiszolgáló kereskedelmi termék, az ügyfélprogramot MPL-típusú szerződéssel terjeszthetjük. Így a nagyobb felhasználóktól bevételre tehetünk szert, mert ők megveszik a kiszolgálót, ugyanakkor az ingyenes ügyfélprogrammal jelentős felhasználói kört építhetünk ki.

Magát a kódot akarjuk védeni vagy azokat a szabványokat, amelyeket a kód megvalósít? Az olyan felhasználói szerződések, mint a SISSL, a program módosítását mindenki számára engedélyezik, mindaddig, amíg az megfelel a szabványügyi testület előírta követelményeknek; azok a felhasználók pedig, akik nem kívánnak az előírásoknak megfelelni, a módosítások referenciáját kötelesek jogdíjmentesen nyilvánosságra hozni, így a szabványt harmadik fél nem sajátíthatja ki.

Ha a programmal szabadalmak is kapcsolatba hozhatók, meg kell fontolnunk, hogy a szabadalmakat is a felhasználói szerződés alá vonjuk-e. Védekeznünk kell a felhasználók azon ötletének megvalósítása ellen is, hogy saját szabad programunkat felhasználva, majd ellenünk fordulva pert indítsanak szabadalmi jog megsértése címén.

Az OSI által jóváhagyott felhasználói szerződések különbözőképpen közelítik meg ezt a kérdést: egyesek erős megtorló intézkedéseket foglalnak magukba, mások enyhébbeket, amelyek kevésbé fenyegetőek a sok szabadalmat birtokló ügyfelek számára.

Ez a lista azonban nem meríti ki az összes szempontot, amelyet figyelembe kell vennünk! A felhasználói szerződés kiválasztása előtt a döntéshozónak és jogászának teljes egészében át kell látnia az adott üzleti helyzetet. A fenti kérdések megválaszolása után még mindig döntenünk kell, érdemes-e saját szerződést létrehozni a szükséges jogi beruházással, vagy jogászunkkal a már létező felhasználói szerződést igazítsuk igényeinkhez? Kérjünk tanácsot a vállalkozásunkat jól ismerő jogásztól! Ne feledjük, a felhasználói szerződés kiválasztását üzleti célok vezérlik! Bárki, aki üzleti céljainkat semmibe veszi, nem a mi emberünk.

A jogi tanácsadás megfelelő kerete egy jogász-ügyfél kapcsolat, amely egy adott helyzet minden tényállását figyelembe veszi, és a helyileg érvényes jogoknak minden tekintetben megfelel. Bár ezt a cikket is jogász írta, a benne foglalt tájékoztatás nem helyettesítheti az esetre szabott, bejegyzett jogásztól származó tanácsadást.

Lawrence Rosen (www.rosenlav.com) magángyakorlatot folytató jogász a kaliforniai Redwood Cityben. A Nyílt Forrás Kezdeményezés (Open Source Initiative ügyvezető igazgatója és jogtanácsosa (➲ www.opensource.org).

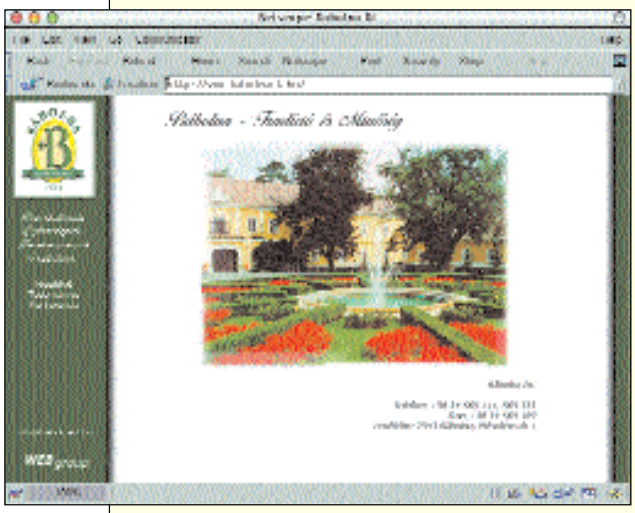
Cégcsokor

Olyan cégeket gyűjtöttünk csokorba, amelyek huzamosabb ideje számos területen Linuxot alkalmaznak, mert fontos számukra a megbízhatóság és a rendelkezésre állás.

Kövessük végig, hogyan zajlik egy cég átállása Linuxra – akár a munkaállomásokat is beleértve!

Bábolna Rt.

1997 nyarán kezdtek ismerkedni a Linuxszal. Már ekkor intranetkiszolgáló kialakításával próbálkoztak IIS és Microsoft Exchange segítségével, azonban hosszas kísérletezéssel sem sikerült megbízhatóan működő rendszert összeállítaniuk.
A Linux először egy 486 DX2 66 MHz-es gépen „kelt életre", amin eleinte RedHatet majd Debiant alkalmaztak. Utóbbi bizonyult megfelelő választásnak, azóta is ezt használják.



A Linuxot lépésenként vezették be, oly módon, hogy egyre több feladatot osztottak rá.

1. Egy meglévő modemet „közösítettek" és kialakítottak egy „kitárcsázó" kiszolgálót, amely rendszeres időközönként leszedte, illetve elküldte az összegyűjtött leveleket. Ezt a szolgáltatást a számítóközpont munkatársaira előbb csak a kimenő levelek esetén, később a bejövők kapcsán is kiterjesztették. Erre a célra a *Sendmail*-t és a *Fetchmail*-t használták.
2. Az internetelérés megosztása céljából egy proxykiszolgálót készítettek, melyhez a *Squid*-et használták fel.
3. Ezután a belső DNS elkészítése került sorra, amelyet az IRC, a web- (*Apache*) és az FTP-kiszolgáló követett. Már ekkor látszott, hogy a meglehetősen gyengécske kiépítésből megbízható, használható rendszert sikerült alkotni.
4. 1997 augusztusában adatbázis-kiszolgálóval (*PostgreSQL*) egészítették ki a meglévő gépen található webkiszolgálót, és erre alapozva már aktív HTML-eszközökkel készítették el a nemzetközi gazdanapok informatikai rendszerét (cégbemutatás, vásári tájékoztatás, vendégkönyv stb.). A fő szempont ismét nem a költség, hanem a megbízható működés volt. A vállalatnál a kezdeményezés sikere alapozta meg a Linux elismertségét.
5. A következő lépés a Novell emulátor (*Mars_NWE*) és a fájlkiszolgáló (*SaMBa*) használatba állítása volt. A feladatok sokasodása következtében szükségessé vált a számítógép bővítése, és valóban üzemszerű lett a Linux alkalmazása. A csak belső levelezésre használt postaládák száma ekkor már elérte a százas nagyságrendet, ezért áttértek a könnyebben beállítható *smail* programra.
6. Létrehoztak egy betárcsázókiszolgálót, amely a telephelyek és a központ közötti adatszolgáltatást bonyolította le. A legnagyobb nehézséget a DOS-on is futó FTP-ügyfélprogramok beszerzése okozta.
7. 1999 tavaszán lehetőség nyílt bérelt vonali internetelérésre. A bérelt vonal kiépítésének idejét ISDN-kapcsolattal hidalták át. Ezt a feladatot is a Linux látta el, és az addig csak belső használatra szánt postafiókok levéltovábbítás (mail relaying) segítségével az Internet felől is elérhetővé váltak. Ekkor már figyelmet kellett fordítani a védelemre, amelyre két megoldás is kínálkozott: saját vagy vásárolt tűzfal kiépítése. Az informatikusok a vezetőség számára az első módszert javasolták, és a Linux ingyenessége itt komoly előnynek bizonyult: az új web- és levelezőkiszolgálóval kiegészítve a szükséges gépek és alkatrészek ára feleannyit sem tett ki, mint amennyibe a legolcsóbb (és biztonságosnak alig nevezhető) kereskedelmi rendszer került volna. A fő érv itt is inkább a kézbentarthatóság és a frissíthetőség volt. Az informatikai csapat inkább vállalta, hogy meghibásodás esetén nem tudja kire hárítani a felelősséget, minthogy nem látja át a rendszert. Figyelemreméltó, hogy a Linuxrendszerekben még ma is hatalmas tartalékok rejlenek, és az indításkor beszerzett PC-k azóta is kiválóan ellátják feladatukat, nem szorulnak bővítésre. Eddig egyszer fordult elő alkatrész-meghibásodás, amit a PC-alapú rendszernek köszönhetően gyorsan sikerült elhárítani.
8. 1999 nyarán került sor egy Oracle Applications rendszer Unix-alapú kiszolgálókkal való bevezetésére. Ennek alapját az teremtette meg, hogy a Linux segítségével már eléggé jól kiismerték benne magukat, illetve könnyebben el tudták sajátítani más Unix-rendszerek használatát. Ez azért fontos szempont, mert az egyéb választások (Mainframe, Windows NT) az áruk, illetve a kedvezőtlen tapasztalatok miatt nem tűntek megfelelőnek. A kezdetben Sun-, később AIX-rendszerekhez ajánlott rendszergazdai X-terminál helyett természetesen saját linuxos gépeket használnak.
9. A fejlesztő és a bevezetést végző külsős munkatársak számára lényeges volt, hogy elérjék az AIX egyes könyvtárait, amit FTP-vel vagy NFS-sel oldottak meg. A Microsoft-eszközökhöz szokott munkatársak számára ez azonban túl bonyolultnak tűnt. Azért, hogy nekik kedvezzenek, a már bevált megoldáshoz folyamodtak: a Linux mindkettőt érti, fordítson. Ez már azonban sajnos túlterhelte a gépet (ugyanis a belső Web, FTP, IRC és DNS továbbra is ezen működött).

10. 2000 tavaszán az IBM 9672 eléréséhez a korábban használt Microsoft SNA kiszolgáló feleslegessé válásával felszabadult egy Dell PowerEdge 4200 gép. Az előző lépésben leírt szolgáltatás kivételével mindent erre a gépre tettek át.
11. A hálózatban lévő útválasztókat és a két AIX-es Unixot is Linux alól felügyelik a *Tklned*, *mrtg* és *NetSaint* programok segítségével.
12. További tervek és ötletek születtek a Linux „munkakörének" kiterjesztésére:

- Samba segítségével PDC (Primary Domain Controler) készítése a windowsos munkaállomások számára, miáltal egyszerűsödne a felhasználók felügyelete;
- Linuxos (esetleg lemez nélküli – diskless) munkaállomások üzembe helyezése.

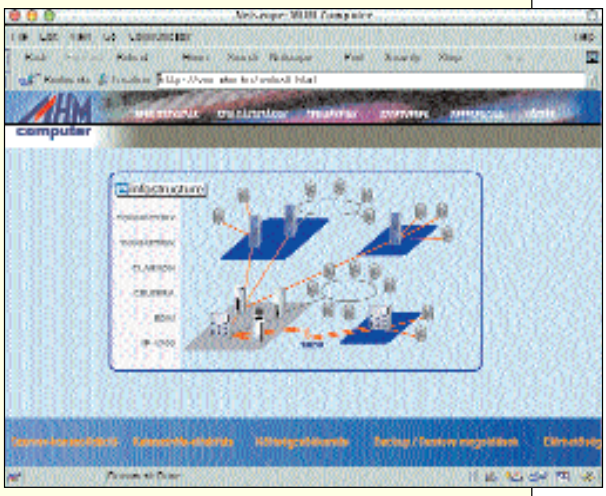
Mecsekfűszért Rt.

Egy Epox KP6-BS alaplapban lévő két 550 MHz-es Pentium III-as processzor, két Promise Ultra66 IDE veérlő és 256 MB RAM az alapja annak a gépnek, amelyben két 6,4 gigabájtos Western Digital merevlemez helyezkedik el RAID1-be kötve a rendszer számára, és két húszgigás IBM merevlemez, szintén RAID1-be kötve az adatoknak. SuSE Linux operációs rendszer gondoskodik a következő programok futtatásáról: főkönyvi rendszer – Micro Focus Application Server; nagykereskedelmi rendszer – Sea-Change 2 (iBCS2-vel futtatva); Cobol fejlesztői rendszer – Micro Focus Object Cobol Developer Suite. Az eddigieket körülbelül ötven felhasználó alkalmazza. Mostanában pedig a StarOffice-kiszolgáló szolgáltatásait próbálgatják, körülbelül 5–7 felhasználó számára. Ezenkívül kisebb alrendszerek is futnak a rendszeren, például Dataflex for Intel Unix segítségével. A programok 200–600 megás fájlokkal dolgoznak. Fájlrendszerként ReiserFS-t használnak.

MHM Computer Hungária Kft.

A feladat nagy rendelkezésre állású kiszolgáló létrehozása volt, amelyet a következő eszközökkel oldottak meg: 600 MHz-es Pentium III processzor, 256 MB RAM-mal felszerelt két gép, melyeken RedHat 7.0-s operációs rendszer futott. Az alkalmazott programok: *BIND* – a DNS-szolgáltatáshoz, *SaMBa* – fájlkiszolgálóként, *Sendmail* – levelezőkiszolgálónak, *Squid* – proxykiszolgálóul, *Apache* – webkiszolgálónak és *Kimberlite* – a fürtözéses szolgáltatásokat lehetővé tevő program. Egyéb alkatrészek: két külső hálózati kapcsoló, külső (megosztott) SCSI-ház, multi initiator SCSI bus (egy SCSI buszon egyszerre két vagy több SCSI vezetőkártya található). A gépek 10/100-as belső hálózaton helyezkednek el, de a két gép között egy külön 100 Mb/s-os összeköttetés is van. A SCSI-kártyák Adaptec 29160-asok, 160 MB/s átvitelt tesznek lehetővé, a merevlemezek 9 GB-os méretűek. (Egyelőre azonban nem kapható olyan merevlemez, amely ezt a sebességet ki is tudná használni.) A közös elérésű merevlemez-alrendszeren ReiserFS jegyzőkönyvező fájlrendszer található.

A nagy rendelkezésreállás azt jelenti, hogy a rendszer nagymértékben hibatűrő. Ha a kiszolgálást végző gép bármilyen okból leáll, feladatát „társa" veszi át helyette. Így a tervezett leállások esetén is lehetségessé válik a folyamatos szolgáltatás. Egymás állapotát a két gép a megosztott merevlemez-alrendszeren keresztül is figyeli, és hiba észlelése esetén a működő gép ki tudja kapcsolni a másikat, nehogy az valamit elrontson. A fürt (a beállítás függvényében) úgy is képes működni, hogy a leállt gép a visszakapcsolása után a feladatokat visszaveszi, és úgy is, hogy visszakapcsolása után tartalékban marad, amíg a másik üzemel.



Ennek a szolgáltatásnak a bevezetése nem a felhasználók nagy száma (a két gépet kiszolgálóként 15 felhasználó használja) miatt, hanem az üzletileg fontos adatok védelme érdekében vált szükségessé.
A beüzemelési próbán kívül nem volt leállás, sőt még tévesztést (failover) sem tapasztaltak. A fürt próbája a következőképpen zajlott: a kiszolgálást végző gépet kikapcsolták, a megmaradt gépnek pedig észre kellett vennie a másik leállását, valamint át kellett vennie a szolgáltatásokat. Körülbelül 15–20 másodperc alatt lezajlott az átállás, tehát ennyit „érzékeltek" a felhasználók a szolgáltatások kieséséből. Mivel a fürtben lévő két gép csak az adatokat osztja meg egymással (adatszempontú fürt), az élő kapcsolatok „elhalnak". Élő kapcsolatokat nem lehet átvenni, csak akkor, ha a két rendszer ezeket az adatokat is folyamatosan megosztja egymással, de ez már a nagygépek világa lenne. A bejelentkezett windowsos ügyfelek önműködően csatlakoznak újra a megosztásokra.
Sorozatunk következő részében folytatjuk a hazai linuxos megvalósítások bemutatását.

Kósa Attila
(atkosa@shinwa.hu) informatikus mérnök. Egy japán cégnél dolgozik rendszergazdaként. 1995-ben találkozott először a Linuxszal. Amikor csak teheti, két kisfiával játszik.

Linux-index

1. A befektetőket ért veszteség a dot-com cégek bukásakor: 4,6 trillió dollár
2. A Microsoft vagyona készpénzben: 30 milliárd dollár
3. A Microsoft helyezése készpénzkészlete alapján, minden amerikai céget figyelembe véve: 1
4. *Bill Gates* és *Steve Ballmer* szerint a feliratkozások (például HailStorm) helyezése a Microsoft jövőbeni bevételi forrásai közt: 1
5. A Linux Counter Project által jegyzett leghosszabb folyamatos üzemelési idő: 434 nap
6. Az Egyesült Államok postahivatalaiban ennyi linuxos gépet használnak a levelek válogatásához: 900
7. A ➲ LinuxCertified.com által jegyzett tanulók között a nők aránya (százalék): 5–8
8. Azon szövetségi számítógépek százaléka, amelyek már 1998-ban is Linuxot használtak: 25
9. Az amerikai Nemzeti Óceán és Légköri Hivatal (National Oceanic and Atmospheric Administration – NOAA) Előrejelzési Rendszerek Laboratóriumában üzemelő Linux Beowulf telepben működő Compaq Alpha munkaállomások száma: 277
10. Ugyanebben a laborban működő linuxos telepek száma: 2
11. A NOAA új rendszere ennyiszer hatékonyabb a régi telepnél: 20
12. A Dell vagyona készpénzben: 8 milliárd dollár
13. Az alkatrészek heti árcsökkenése a Dell szerint, 2001 májusában (százalékban): 1
14. A Dell alkatrészkészlete ennyi napra elég: 3
15. A Transaction Processing Performance Council TCP-H mérései alapján 2001 májusában az SGI 1450 Server által elért helyezés (IBM DB2 UDB EEE 7.2-vel, 2.4.3-as Linux-rendszermaggal): 1

Forrás
1–4: Business Week
5–6: Linux Counter project
7: ➲ LinuxCertified.com
8–14: ➲ CNN.com
15: Transaction Processing Performance Council ➲ http://www.tpc.org/

Ők mondták

Bulira fel!
Csak az őrültek veszik magukat teljesen komolyan. (*Sir Max Beerbohm*)

Mielőtt az őrült tudós megőrült volna, valószínűleg létezett egy olyan időszaka, amikor még csak félig volt őrült: ilyenkor rendezte a legjobb bulikat. (*Jack Handey*)

Jéghegy előttünk!
A szállítási hasonlatokat a vasútról sürgősen át kell ültetnünk a hajózásra. Bármennyire ügyesen kormányozzák is titanikjaikat az olyan „széles sávú" iparmágnások, mint *Michael Eisner*, *Rupert Murdoch* vagy *Steve Case*, a hatalmas jéghegy, melybe nevüket vésték, már levált az Északi-sark jégsapkájáról és menthetetlenül feléjük sodródik. Ezt a jéghegyet természetesen Internetnek hívják. (*Christopher Locke*)

A fogalmat illetően kissé bizonytalan
Szerintem ha azt mondod, hogy meg fogsz tenni valamit és mégsem teszed meg, ez a megbízhatóság jele. (*George W. Bush*)

Ésszerű mellébeszélés
A „Médiakapcsolatok"-nak a világ minden országában törvénytelennek kellene lenniük. A legjobb esetben is a médiakapcsolatok csak az unokatestvérek közötti vérfertőző viszonyhoz hasonlíthatók. A Weben szerepel a legrosszabb példák egyike: a hirdetési ügynökségek és a kétes tehetségű webgrafikusok szörnyűséges szövetsége. (*A. Head Lemur*)

Ne csupán kortársaidnál és elődeidnél legyél jobb. Próbálj túltenni önmagadon. (*William Faulkner*)

Nem azért használtuk a „nyílt forráskód" (open source) kifejezést, hogy felidegesítsük a Free Software Foundation embereit, hanem azért, hogy olyan jelentésteret hozzunk létre, ahol szabadon beszélhetünk és nem kell attól tartanunk, hogy elijesztjük azokat az embereket, akiket meg akarunk győzni. (*Eric S. Raymond*)

Ó, bébi!
A szex a szublimált matematikai ösztön. (*M. C. Reed*)

2500 mérföld 0,0134 fénymásodpercnek felel meg. Ha 10 Gb/s-t (OC192C) 1 GB/s-nek veszünk (kerekítés), akkor 13,4 megabájt lesz az eredmény. Ha 2,5 Gb/s-t veszünk alapul (OC48C), akkor néggyel kell osztani, így három megabájtot kapunk. (*Mike O'Dell*)

(Újra)indulás! Előre!
A Windows felismerte, hogy meggondoltad magad. A Windows most újraindul, hogy alkalmazza a változásokat. (*Gates_throws_tantrum*), a Slashdoton

Az irónia halott. (*Don Marti*)

Nem hiszem, hogy fenyegetést jelentene. Legfeljebb nem lehet rossz kódot eladni arra alapozva, hogy úgysem nézheti meg senki. (*Amy Wohl a Linuxról és a nyílt forráskódról*)

A második legősibb foglalkozás: a könyvelés. (*Craig Burton*)

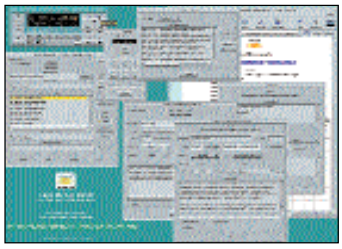
Új termékek

Opera 5.0

Az Opera 5.0 az Opera Software nagy tudású webböngészőjének hivatalos kiadása, amely a Qt fejlesztőkörnyezet segítségével készült. Az Opera böngésző gyorsan jeleníti meg a lapokat, a képekre és a dokumentumra vonatkozó beállításokat azonnal érvényesíti, sokoldalú „húzd és dobd" képességekkel rendelkezik, továbbá támogatja a többablakos üzemmódot és a nagyítást (1000%-ig). Az Opera mérete 1,5 MB dinamikusan kapcsolt Qt-vel, statikusan kapcsolt Qt-vel 3 megabájt. A következőket támogatja: 128-bites titkosítás, TLS 1.0, SSL 2 és 3, CSS1 és CSS2, XML, HTML 4.01 és JavaScript 1.3.
Opera Software A/S, Waldemar Thranesgt. 98, 0175 Oslo, Norway,
e-mail: commercial@opera.com,
➲ http://www.opera.com/

Xmcd 3.0

Az Xmcd CD-lejátszó alkalmazás – amely X Window System alatt a legtöbb Unix-változaton fut – elérte a 3.0 változatszámot. A 3.0-s csomag tartalmazza az Xmcd-programot, amely X11/Motif programkönyvtáron alapuló CD-lejátszó segédprogram, valamint a cda programot, mely



parancssorból vezérelhető nem grafikus CD-lejátszó alkalmazás. Az Xmcd rengeteg CD-ROM-, CD-R-, CD-RW- és DVD-eszközt, továbbá többlemezes egységeket és néhány régi SCSI-1 meghajtót is támogat, mint ahogyan CD-k felismerését is a Gracenote CDDB-n keresztül, valamint képes kapcsolódni az internetes CDDB-kiszolgálókhoz a szerző, a lemezcím, a számcímek és egyéb adatok megszerzésének céljából. Az `xmcd`-re a GPL felhasználói szerződés vonatkozik, és az alábbi weboldalról tölthető le:
➲ http://www.amb.org/xmcd/

Quick Restore adatvédelem

A Quick Restore 2.7.4-es változata adatvédelmi program, amely különböző összetevőkből álló hálózatban oldja meg az adatmentést és adatvisszaállítást. A Network Data Management Protocollon (NDMP) alapuló Quick Restore lehetővé teszi a NAS-rendszerek helyi és távoli biztonsági mentését szalagra és könyvtárakba. Új tulajdonságai közé sorolható, hogy az egyszerűbb működtetéshez több felületet támogat, bővítették a tűzfaltámogatást is, valamint továbbfejlesztették a DLT szalagformátum kezelését, és az újonnan elérhető szalagkönyvtárakat (tape libraries) szintén támogatják.
Workstation Solutions, Five Overlook Drive, Amherst, New Hampshire 03031,
telefon: 800-487-0080,
e-mail: info@worksta.com,
➲ http://www.worksta.com/

INTERNETpro Enterprise Stack

Az Internet Appliance terméke, az INTERNETpro Enterprise Stack olyan tároló- és biztonsági megoldás, amely egységbe foglalja a hatékonyabbá tett webkiszolgáló programot és a gépalkatrészeket. Az egész egység központilag karbantartható és felügyelhető, miközben magas elérhetőséget és méretezhetőséget, valamint biztonságot nyújt szolgáltatóknak és vállalkozásoknak egyaránt. Alkalmazási területei többek között: adatközpontok, kiszolgálók közötti VPN és távolból elérhető VPN.
Internet Appliance, Inc.,
40515 Encyclopedia Circle, Fremont, California 94538,
telefon: 877-986-6366,
e-mail: sales@internetappliance.com,
➲ http://www.internetappliance.com/

iXsystems 1U kiszolgáló

Már kapható az iXsystems (korábban BSDi) iXtreme 1400 1U állványba szerelhető internetkiszolgálója. Az iXtreme 1400 38 cm mély, és nagysága kevesebb, mint kétharmada egy átlagos 1U kiszolgáló méretének. A processzorokat, a lemezegységeket és a tápegységeket nyolc ventilátor hűti, ezért a kiszolgáló magas környezeti hőmérsékleten is üzemelhet. Az 1400-asba legfeljebb 4 GB memória, két ethernetkapu, két EIDE vagy két SCSI nagysebességű lemezegység és két, legfeljebb 933 MHz-es processzor szerelhető.

iXsystems, Inc., 1550 The Alameda, Suite 100, San Jose, California 95126,
telefon: 866-449-7978,
e-mail: info@ixsystems.net,
➲ http://www.ixsystems.net/

SysOrb 2.0

A Solit Solutions ApS kiadta a SysOrb 2.0-t, amely több felületen futó hálózat- és rendszerfelügyeleti eszköz. A támogatott felületek: RedHat, Debian, FreeBSD, Windows NT/2000 és Solaris. A beállítások, a valós idejű és a régi adatok adatbázisa webes felhasználói felületen keresztül érhető el. Gond esetén a rendszer SMS-ben, levélben vagy a személyhívóra riasztást küld.
A program és a felhasználói szerződés ingyenesen letölthető a ➲ http://www.sysorb.com/ címről.
Solit Solutions ApS, Maglebjergvej 5D, DK - 2800 Lyngby,
telefon: +45-45-880-888,
e-mail: sysorb@sysorb.com,
➲ http://www.sysorb.com/,
http://www.solit.dk/

MindRover: The Europa Project

A Loki Software kiadta a CogniToy népszerű játékának, a MindRover: The Europa Projectnek a linuxos változatát. A háromdimenziós stratégiai, illetve programozó játékban a játékosok önálló robotjárműveket építhetnek, amelyeket gyorsasági versenyeken indíthatnak. A menet üres járművel indul, amelyhez nekünk kell érzékelőket, fegyvereket és motorokat adnunk. Az összetevőknek együtt kell működniük, tulajdonságaik vizuális programozási rendszerrel állíthatók be. A játékosok egymás között a robotokat elektronikus levélben vagy a weben keresztül megoszthatják.
Loki Software, Inc., 250 El Camino Real, Suite #100, Tustin, California 92780,
e-mail: info@lokigames.com,
➲ http://www.lokigames.com/

A hónap szakmai tanácsai

Régi szalagok másolása

Jelenlegi munkahelyemen elődeim a biztonsági mentéseket nem kezelték egységesen: a DAT-tól az Exabyte-ig különféle adathordozókat használtak. Azt a feladatot kaptam, hogy a különféle formátumokat egységes alakba (jelen esetben AIT-formátumba) másoljam át. Próbáltam az egyik szalagot a másikra átmásolni, de nem leltem a megfelelő parancsot. Most a `tar` GNU-változatát használjuk, és a jövőben is ezt szeretnénk tenni. Próbáltam a

```
dd if=/dev/<eszk zfÆjl>
of=/dev/<eszk zfÆjl2>
```

parancsot, de ez még csak el sem olvasta a szalagot.
Charles Long, charlesl@wildbrain.com

Az általad használt parancs jónak tűnik, az egyetlen hiányzó elem a blokkméret (`bs`) lehet, amennyiben annak idején a szalagokat meghatározott blokkmérettel írták meg (ezt ki kell találnod). Ezután a parancs így néz ki:

```
dd if=/dev/XXXX of=/dev/YYYY
        ➥bs=<sajÆt_blokkmØret>
```

Felipe E. Barousse Boué, fbarousse@piensa.com

A `dd`-nek el kell olvasnia a szalagot, függetlenül attól a programtól, amellyel a szalagra írtak (például `tar`, `dump`, `NetBackup`). A gond a következők egyike lehet:
1. a szalagot egy másik, nem együttműködő meghajtóval írták,
2. a szalag hibás,
3. a meghajtó fejei piszkosak vagy
4. a szalag fejléce hibás.

Másolás előtt próbáld meg a szalagokat az eredeti programmal *csak olvasható mód*-ban beolvasni, hogy meggyőződj a szalag(ok) és a meghajtó(k) helyes működéséről. A biztonsági mentés semmit nem ér, ha a szalagokat semmilyen módon nem tudod elolvasni.
Keith Trollope, keith@wishing-well.demon.co.uk

A második merevlemez megvédése önműködő telepítésnél

Fő merevlemezemre RedHat 7.1-et szeretnék telepíteni oly módon, hogy csakis az legyen rajta. A másik lemezen Win 98 található. A lemez üres, nem szeretném mindkét rendszert menüből elindítani. A második lemez fizikai kiiktatásán kívül van-e lehetőség az első önműködő felosztására?
Mit javasoltok, hogyan osszam fel 40 GB-os lemezemet?
Paul Henman, linuxj@henman.ca

A RedHat egyértelműen támogatja, hogy lemezedet úgy oszd fel, ahogy szeretnéd. Általában a következő felosztást szoktam javasolni:

`/`	100 MB
`/safe`	100 MB
`/usr`	3–4 GB az esetedben, esetleg lehet több is
`/var`	a fennmaradó hely

Ezeket a közvetett hivatkozásokat kell létrehoznod: a `/tmp`-t a `/var/tmp/tmp`-re és a `/home`-ot a `/var/home`-ra. Ennek a felosztásnak az az előnye, hogy a rendszer szempontjából létfontosságú gyökérfájlrendszer kicsi marad, így a meghibásodás esélye kisebb, valamint a `/safe`-en állandó biztonsági mentést tarthatsz fenn.
Marc Merlin, marc_bts@valinux.com

Az X indítása az indítólemezről

A SuSE 6.3-at a merevlemez egy távoli részére (extended partition) telepítettem, ezért nem használhatok *LILO*-t. Később indítólemezt készítettem a CD-ről a `rawrite.exe` segédprogrammal. Amikor lemezről indítok, és a telepített rendszer helyét megadom, a parancssoros felületet tudom elérni. Hogyan indítható el a grafikus felület?
Manoj Ramakrishnan, mark2gp@sify.com

A grafikus felület több módon is elindítható. A `startx` paranccsal az X-et közvetlenül elindíthatod. Rendszergazdaként futtathatod az `xdm`-et, a `kdm`-et vagy a `gdm`-et; indítóállományuk általában a `/etc/init.d`-ben található, de parancssorból is indíthatók. Az utolsó és talán a legegyszerűbb lehetőség, ha a *YaST2*-ben beállítod a megfelelő futási szintet (3.).
Marc Merlin, marc_bts@valinux.com

A LILO az MBR-be (fő rendszerindító terület) is telepíthető, ezáltal elkerülhető a hajlékonylemezes rendszerindítás. A rendszer telepítése alatt ez a lehetőség valószínűleg elkerülte a figyelmedet.
Csontos Gyula, Csontos.Gyula@linuxvilag.hu

Fájlleíró bájtjainak beállítása

Szeretném tudni, hogyan kell beállítani a fájlleírók (inode) méretét a teljes merevlemez összes fájlrendszerén? Lehetőségem nyílik-e arra, hogy a fájlleírók méretét a RedHat 7.0 telepítésekor megadjam, vagy a `debugfs`-t kell használnom a fájlleíró méretének megváltoztatásához?
Matt Walters, matt.walters@syberos.com

A fájlrendszer létrehozása után a fájlleírók méretét már nem lehet megváltoztatni. Ha a RedHat telepítésekor szeretnéd ezt megadni, lépj ki a héjba (ALT+F2 a szöveges módú telepítésnél), és a fájlrendszereket magad hozd létre, például:

```
mke2fs -s 1 -b 4096 -i 8192 -m 1
➥/dev/sda1
```

ahol a `-s 1` értelmezi a szuperblokkot (Linux 2.2 vagy frissebb rendszermag), a `-b 4096` a lemez blokkméretét 4 K-ra állítja be, a fájlleírók méretét a `-i 8192` 8 K-ra állítja be, és a `-m 1` a fenntartott blokkok számát a fájlrendszer méretének egy százalékában határozza meg (a manapság elterjedt nagy lemezeknél ez megfelelő). Ezután térj vissza a telepítőbe, és válaszd azt a lehetőséget, hogy a fájlrendszereket ne formázza meg.
Marc Merlin, marc_bts@valinux.com

Teljesen kétirányú vagy sem?

A HP-UX rendszerekben egyszerűen megtudható, hogy egy ethernetkártya teljesen kétirányú-e (full-duplex), csak ki kell adni a `lanadmin -x lan0` parancsot. A válaszból kiderül, hogy a kártya teljesen kétirányú 100 Mb-es módban található vagy sem. Régóta keresek valami hasonlót Linux alá.
Boleslaw Mynarski, bman@bolek.com

Természetesen létezik, kettő is: az `mii-tool` és az `mii-diag`. Letölthetők például az
➲ ftp://ftp.valinux.com/pub/support/flory/mii-tool/ címről.
Marc Merlin, marc_bts@valinux.com

Szép billentyűzet, de hol vannak az ékezetek?

Hogyan lehet az ISO8859 karaktertáblázat felső részében elhelyezkedő betűt, például egy *á*-t X Window System alatt beírni? A karakteres konzolon az ALT+225 segítségével működik a dolog. Több leírást is olvastam az úgynevezett „Compose" gombról, de egyik útmutató alapján sem értem el eredményt.
Kevin Goess, knet@goess.org

Készítettem egy .Xmodmap nevű fájlt (ügyelj a pontra a fájlnév elején), ennek segítségével elérhetők a spanyol, angol, portugál és francia nyelvekhez szükséges karakterek az úgynevezett „dead-keys", a CTRL, az ALT és az ALT GR használatával. Az /usr/lib/X11/xinit/.Xmodmap fájlomat a ➲ http://www.piensa.com/xmodmapfile/ címen elérhetővé tettem.
Ez a fájl minden alkalommal „lefut", amikor az X-et elindítják. Az `xmodmap` parancs segítségével akár kézzel is futtathatod. (A magyar `xmodmap` fájl az alábbi címről tölthető le: ➲ ftp://ftp.cs.elte.hu/pub/magyar/X11/keyboard/)
Felipe E. Barousse Boue, fbarousse@piensa.com

A Gnome Character Map nevű grafikus segédprogramjával a különleges karakterek kiválaszthatók. Letölthető a ➲ http://www.gnome.org/ címről.
Paul Christensen, pchristensen@penguincomputing.com

Nem lehet bejelentkezni

Kis otthoni hálózatomban Linux, Win95 és WinNT él együtt. Újabban a Linux nem enged be Telneten keresztül, és közvetlenül a konzolon sem tudok belépni. Ezt az üzenetet kapom:

```
/sbin/login: /lib/libc.so.6: version
        ➥'GLIBC_2.1.3' not found
        ➥(required by /sbin/login)
```

A /lib/libc.so.6 fájl a /lib/libc-2.1.1-re mutató hivatkozás. Vajon mi hiányzik?
Larry Busse, ljb@one.net

Úgy tűnik, frissítetted a `/bin/login` programot (valószínűleg az util-linux RPM-csomagon keresztül), és most függőségi gondod támadt, mert a program csak a `glibc 2.1.3`-változattal képes együttműködni, neked pedig a `glibc 2.1.1` van telepítve. Nem tudom biztosan, hogy sikerült-e telepítened az RPM-et, hiszen annak panaszkodnia kellett volna a programkönyvtárak összeférhetetlensége miatt. Az egyik lehetőség, hogy valaki a tudtod nélkül hozzáfért a rendszeredhez, és egy módosított `/bin/login`-t tett fel. Ha nem így lenne, telepítsd újra a RedHat 6.0-s terjesztés részét képező `util-linux` csomagot (az `rpm -U --force` paranccsal), és minden rendbe jön.
Marc Merlin, marc_bts@valinux.com

A CD-ROM befűződik... de hová?

Megpróbáltam Sun StarOffice 5.1-et telepíteni a gépemre. Befűztem a CD-ROM-ot (`mount /mnt/cdrom`), de amikor a CD-ROM könyvtárába léptem (`/cd`), nem láttam ott a fájlokat.
Luis Embalo, zulobaby@hotmail.com

Győződj meg róla, hogy a megfelelő könyvtárat nézed-e. Ha a `mount /mnt/cdrom` hibátlanul lefutott, a CD-ROM tartalma a `/mnt/cdrom` könyvtárban tekinthető meg. Próbáld ki a `mount` parancsot kapcsolók nélkül. Az én rendszeremen többek között ezt a sort adja vissza:

```
/dev/hdc on /mnt/cdrom type iso9660 (ro)
```

Ebből értesülhetünk, hogy a CD-ROM-meghajtó (hdc) a `/mnt/cdrom` könyvtárba van befűzve. Ha a `/cd` könyvtárba szeretnéd befűzni, írd be:

```
mount /dev/cdrom /cd
```

Paul Christensen, pchristensen@penguincomputing.com

Az Apache jserv telepítése

Megpróbáltam telepíteni az Apache jserv-et Linux Mandrake 7.2-n. A könyvekben és az Apache jserv leírásában a /usr/local/apache vagy a /usr/local/src szerepel alapértelmezett könyvtárként, de nekem egyik könyvtár sincs meg. Kipróbáltam egy példát is a „Java Programming on Linux" című könyvből:

```
./configure --with-apache-install=/usr
➥--prefix=/opt/apache_jserv
➥--with jsdk=/usr/local/Java/JSDK2.0
```

A következő hibaüzenetet kaptam: ./configure no such file or directory.
Christopher Nallo, cnallo1@home.com

A /usr/local/apache akkor érvényes, ha az Apache-ot saját magad fordítod és telepíted. Ha nem ez történt, akkor valószínűleg RPM-csomagból telepítetted. Ha kiadod a következő parancsot

```
rpm -qa | grep apache
```

megtudod, hogy hány és milyen változatú Apache-összetevő került telepítésre. Ezután add ki a

```
rpm -ql <apache-csomag>
```

parancsot, ahol az `<apache-csomag>` az első parancs egyik válaszsora. Ez megadja, hogy az adott csomag fájljai hová települtek.
A `./configure`-ral kapcsolatos gondodra azt tudom mondani, hogy a beállítófájlokat mindig abból a könyvtárból kell futtatni, ahol elhelyezkednek. Valószínűleg más könyvtárban voltál, nem abban, ahol a telepítendő csomag beállítófájlja volt.
Felipe E. Barousse Boue, fbarousse@piensa.com

A Linux Journal honlapján számtalan gond megoldásához találhattok további segítséget. A Sunsite tükröoldalait, a gyakran feltett kérdéseket és egyéb útmutatásokat a ➲ www.linuxjournal.com honlapon olvashatjátok el. A rovatban közzétett válaszokat Linux-szakértők kis csapata készítette el. További kérdéseiteket szívesen fogadják (angol nyelven) a ➲ www.linuxjournal.com/lj-issues/techsup.html címen, ahol csak egy kérdőívet kell kitöltenetek, de a bts@ssc.com címre levelet is írhattok. A levél tárgyában szerepeljen a „BTS" kulcsszó.

Felület(függet)len alkalmazások

Havi főtémául a felületeket választottuk, mely döntés némi magyarázatot igényel, ugyanis így, a szó mindennapi jelentésében a legtöbben egyből valamiféle *fizikai* felületre gondolnak. Ez lehet tükörsima fém, érdes fal vagy tapéta, elképzelhetünk azonban vízfelületet is – vagyis ilyen és ehhez hasonló dolgok jutnak az emberek eszébe, ha a fenti kifejezést hallják. Jelen esetben szó sincsen ilyesfélékről, hacsak nem a sokféleséget vesszük figyelembe. Jelenlegi használatunkban a *felület* szó az angol platform (jól vagy rosszul) magyarított változata, ezért jelenthet például különféle számítógépeket, programokat vagy operációs rendszereket.

A Linux igazi többfelületes operációs rendszer, kedvencünket x86-os (a világon valószínűleg a legtöbb Linux ilyen gépeken fut) Amiga-, Apple-, Sun-, valamint Alpha-gépeken is futtathatjuk. Egyre több beágyazott rendszereket készítő cég használ Linuxot a készülékei működtetéséhez, könnyű alakíthatóságának és nyílt forráskódjának köszönhetően. A ➲ http://www.linuxlinks.com/Ports/ címen körülnézve számos érdekes hivatkozást találhatunk a Linux különböző kiépítésű számítógépeken történő futtatásához.

Ebben a számban olvashatnak a Power PC-s Linux telepítéséről, buktatóiról és ezek megoldásáról. Ezzel is megpróbáljuk a MacOS híveit arra rávenni, hogy egy próbát azért a Linux is megér (25. oldal). A beágyazott rendszerekről szóló cikkünkben pedig olyan készülékeket mutatunk be, amelyeknél a Linuxot választották operációs rendszernek (32. oldal). Néhány éve még nem is gondoltunk volna arra, hogy a televíziókészülékekbe is egy, a célnak megfelelően módosított Linuxot fognak elrejteni, így a korábbi szerepében továbbra is tündöklő csodadoboz tényleg csodákra lesz képes, segítségével elérhetjük az Internetet, elektronikus levelezésünket, mindemellett nézhetjük a televízióadást vagy éppen fordítva (természetesen ez gyártótól és készüléktől függ). Rendszermagot finomhangolunk Alpha-alapú gépeken, melynek következtében jelentős sebességnövekedés érhető el, ezáltal még nagyobb teljesítményre bírhatjuk az egyébként is komoly eredményeket felmutató rendszereket (28. oldal).

Az Apple G4 AltiVec egységének kihasználásáról, az erre történő programkódok írásáról és fordításáról az és utasítások használatával szintén jelentős erőforrásokat szabadíthatunk fel (77. oldal).

Az XML segítségével pedig megtanulhatjuk, hogyan készíthetünk minden felületen könnyedén feldolgozható, így teljes mértékben hordozható adatokat (41. oldal).

Csontos Gyula

A LinuxPPC 2000 Q4 telepítésének buktatói

Olvasd el a kézikönyvet, mielőtt a Linuxot rászabadítod a Macedre!

Nemrég jelent meg egy írásom a *Linux Journalban*, amely arról szólt, hogy elsődleges operációs rendszerként a Mac OS X-et használom (lásd „Mac OS X: First Impressions”, elérhető a ➲ http://www.linuxjournal.com/articles/misc/0035.html címen). A Mac OS X (az X „tíz”-nek olvasandó) kipróbálása és használatba vétele előtt a Mac OS 9/LinuxPPC kettős rendszert használtam. Régi Mac-felhasználóként – bár a szívem mélyén még mindig Unix-rajongó maradtam – a Mac OS X lehetőségeitől el voltam ragadtatva, hiszen ez esetben nem másról van szó, mint Mac grafikus felületről Unix-rendszermag felett. A Mac OS X-et lelkesen telepítettem az asztali gépemre, amely egy Macintosh Server G3. Kezdetben egész jól ment a rendszer, ám a mézesheték hamarosan véget értek, és néhány dolog egyre jobban kezdett zavarni. Fő gondjaim az új környezettel a következők voltak: szörnyen lassú, nem támogatja a hajlékonylemezeket, valamint gondok akadtak a GNU-programok átvihetőségével, és a „klasszikus” Mac OS emulálás gyengére sikeredett. Különösen zavaró, hogy a Mac OS X az AppleTalk fájlmegosztást a régi Macintosh számítógépekkel nem támogatja. Számomra ennek elviselése bizonyult a legnehezebbnek, mert végül is arról van szó, hogy az Apple új operációs rendszere nem tud kapcsolatba lépni a régivel az Apple kifejlesztette hálózati protokollon keresztül. A múlt év végén elhatároztam, hogy a LinuxPPC kedvéért letörlöm a Mac OS X-et. A Linux PPC 2000 Q4 (LPPC2000) általános elérhetőségét a 2001. januári MacWorld Expóra jelentették be, és a LinuxPPC weboldalán előzetes rendeléseket lehetett leadni. Habozás nélkül rendeltem egy példányt.

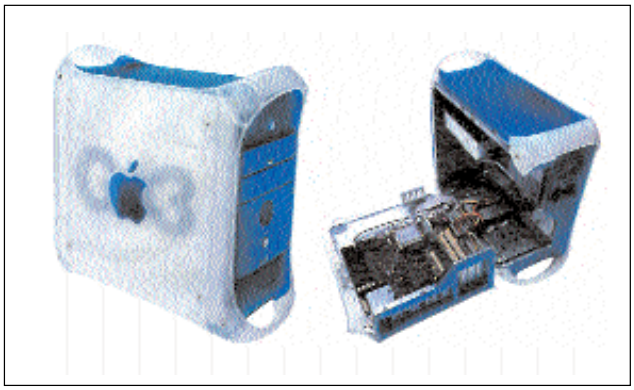
A LinuxPPC érkezése

2001. február elején a LinuxPPC-csomag postai úton érkezett meg. A dobozban négy CD-t, egy nyomtatott felhasználói kézikönyvet és egy LinuxPPC „Think Linux” feliratú pólót kaptam. A négy lemez a következőket tartalmazta: a telepítőt, a forráskódot, az FWB Software’s Hard-Disk Toolkit-PE-t, továbbá egyéb programokat. A lemezeken még 2000. decemberi keltezés szerepelt. Három gépre szerettem volna telepíteni: a G3-ra (128 MB RAM, 4 GB HDD), egy eredeti átlátszó zöld iMacre (32 MB RAM, 4 GB HDD), valamint egy piros iMacre (32 MB RAM, 6 GB HDD). A három közül csak a piros iMachez kellett az FWB-Toolkitet használni (ez egy harmadik fél által készített program, amely a Mac merevlemezt úgy osztja fel újra, hogy annak tartalma megmarad). A merevlemez felosztása a másik kettőnél nem okozott gondot: a G3 a Mac OS X-et futtatta, ezt úgyis le akartam törölni, a zöld iMac pedig a LinuxPPC egy régebbi változatával működött (az 1999-ből származó Q3 szintén megérett a törlésre). A piros iMac az otthoni gépem, amin gyerekeim használnak Macre írt oktatóprogramokat.

A zöld iMac

A zöld iMackel kezdtem. Mind a LinuxPPC webhelye, mind a felhasználói kézikönyv külön hangsúlyozta, hogy ez a változat rendszerindításra alkalmas CD-n kerül forgalomba. Csak annyi az egész, hogy a telepítőlemezt behelyezzük a meghajtóba, és újraindítás közben nyomva tartjuk a C gombot. Mi lehetne ennél egyszerűbb? Ezt tettem a zöld iMacen, és pár pillanat múlva elindult a Linux, betöltődött az LPPC2000 grafikus telepítője (az *X Linux Installer*), majd megjelent a képernyőn. Eddig minden rendben zajlott, ezt követően csak a képernyőn megjelenő utasításokat kellett követnem. Négy nagy és két kicsi gomb tűnt fel, a nagy gombok feliratai: *Instructions*, *Language*, *Select Partitions* és *Choose Packages*, a kicsiké *Options Menu* és *Reboot* voltak. Az Instructions gomb megnyomásával rövid bevezetőt kapunk a telepítési folyamathoz. A Language gombbal választható ki a használandó nyelv. Az igazi munka azonban a Select Partitions gomb használatával kezdődik: ha erre kattintunk, a *Mount Partitions and Setup Swap* ablak jelenik meg, amelyben a lemezrészek csatolási pontjait és neveit adhatjuk meg. A lemezrészek ebből az ablakból formázhatók és csatolhatók. Természetesen ez előtt rendelkeznem kellett a megfelelő lemezrészekkel, amelyeket az LPPC2000-hez adott Perldisk programmal készítettem elő. A Perldisk nem szép, viszont működik, bár egy kissé furcsának találhatjuk a használatát. Nem tetszett az *Add Partitions* nevű számlap, amelyen az újonnan létrehozandó lemezrész méretét kell beállítani. Nehéz volt például pontosan 20 MB-ot megadni, végül a lemezrész pontos méretét kézzel számoltam ki, és az eredményt a számlap alatti szövegmezőbe írtam be. A méret kiszámításánál fontos, hogy a képletet megfelelően alkalmazzuk. Például a 20 MB méretű lemezrészhez a helyes értéket a (1024×2)×20 kifejezés adja.
A lemezrészek adatai a lemezre íródnak, majd a telepítőprogram újraindítást igényel (a Mac indításakor ne feledd nyomva tartani a C gombot, hogy a telepítés folytatódjék!). Miután a telepítőprogram ismét futott, újra a Select Partitions gombot választottam, és bejelöltem, hogy a telepítés megkezdése előtt formázza és csatolja a lemezrészeket. A zöld iMac esetében felosztási stratégiám a következő volt: /boot (20 MB), cseterület (128 MB), / (1 GB) és /home (1,8 GB). További egy gigát hagytam Mac OS formátumú meghajtóként. Miután a *Done* gombra kattintottam, az ablak eltűnt.
A Choose Packages ablakban a csomagcso-

portok listája bukkant fel. Az alapértelmezett beállításokon felül a következőket jelöltem be: *Editors*, *Interpreters*, *Network Servers*, *Network Apps/Utilities* és *Development*. Figyelem, ne változtasd meg az alapértelmezett csomagok kijelölését, mert összekuszálhatod vele a rendszert! Rákattintottam az *Install* gombra, mire elindult



a LinuxPPC és csomagjainak másolása. Hosszú-hosszú ideig tartott a 600 MB-nyi csomag felmásolása, amit az eredeti iMachez adott „szörnyű” CD-meghajtó számlájára írtam. Körülbelül négy óra múlva megjelent egy párbeszédablak, amelyben a gyökérlemezrész neve szerepelt („szép gondolat”), valamint két további gomb: *Reboot* és *Additional Options*.

Az Additional Option a *Configure LinuxPPC* ablakot hozza fel, amely néhány telepítés után választható beállítást ajánl fel. Ezek a következők: *Set Password, Set Date & Time, Set Timezone, Configure inetd, Set Network Settings, Configure modem port, Configure PPP Settings, Configure Runlevels, Setup Users and Groups, Configure X modifier keys és Run linuxconf*. Beállítottam az órát, megadtam a rendszergazda jelszavát, az időzóna „Eire” lett, továbbá eltávolítottam a felesleges hálózati folyamatokat az `inetd`-ből, és beállítottam az IP használatát. (Megjegyzem, a `chkconfig` jobban tetszik, mint a *Runlevel Editor*.) Bezártam a Configure LinuxPPC ablakot és a Reboot gombra kattintottam.

A zöld iMac újraindult, és a rettegett Apple villogó kérdőjel jelent meg a képernyő közepén. A zöld iMac nem talált indítható operációs rendszert!

Az LPPC indítását szabályozó program neve yaboot. A felhasználói kézikönyv jól leírja a program használatát, de nézd meg a LinuxPPC webhelyét, amelyen a 40. oldalon található kicsi, ám fontos sajtóhiba javítását adják meg.

A felhasználói kézikönyv alapján az LPPC2000-et adtam meg a zöld iMac alapértelmezett operációs rendszereként. Ezt úgy állítottam be, hogy a Mac OS CD-ről indítva a rendszert a *Startup Disk* vezérlőpultot futtattam. A /boot lemezrészt alapértelmezettként jelöltem ki és a számítógépet újraindítottam. Az iMac azonban továbbra sem indult. Megnéztem a leírásban, hogy miért történhetett ez, és kiderült, hogy a /boot-ban egy „hamis” Mac OS rendszermappának kellene lennie, amelyet a telepítő másol oda. A telepítő azonban ezt nem tette meg, a /boot könyvtárban csak a yaboot binárist és pár rendszermagfájlt találtam. Azt hittem, itt elakadok, amikor eszembe jutott, hogy a rendszer a telepítő CD-ről is elindítható. Meg is találtam a CD-n a Mac rendszerkönyvtárt, átmásoltam a /boot-ba, átírtam a yaboot.conf-ot, újraindítottam a zöld iMacet, és lám, a Linux feléled. Nagyszerű!

Telepítés a G3-ra

Annak ellenére, hogy a zöld iMacre sikeres, bár kissé bonyolult telepítést hajtottam végre, a G3-ra telepítésnél újabb akadályokba ütköztem. Kezdjük azzal, hogy a trükk a C gomb nyomva tartásával rendszerindításkor nem működött. Csak a „New World ROM”-mal rendelkező Macek tudnak a telepítő CD-ről közvetlenül rendszert indítani. Ha iMacet vagy annál újabb Macet birtokolsz, nem lesz gondod. Az okozta számomra a fennakadást, hogy a G3 képes volt a Mac OS indító CD-jét betölteni, nem értettem hát, hogy a telepítő CD-vel ez miért nem működik. Természetesen ha elolvastam volna a nyomtatott kézikönyvet, láthattam volna, hogy a G3-nak e téren gondjai akadnak. Mielőtt az ember bármibe belefogna – most már számomra is egyértelmű –, megéri elolvasni a teljes nyomtatott kézikönyvet. A grafikus felületnek köszönhetően a telepítési folyamat egyszerűbb, ennek ellenére nem magától értetődő, ami hibás rendszerindulást eredményezhet. Ez történt velem is a G3 esetében. Miután a G3 a CD-ről nem indult, a kézikönyv javaslatára következő lépésként a Mac OS-t futattam, majd a telepítő CD-ről elindítottam a Mac-alapú telepítőprogramot. Betöltöttem CD-ről a Mac OS-t, és a G3 merevlemezét a következőképp osztottam fel: két Mac OS lemezrész (750 MB és 250 MB), /boot (20 MB), csereterület (128 MB) és / (3,1 GB). Telepítettem a Mac OS 9.0 alaprendszert a Mac OS rendszerterületre, hogy szükség esetén használhassam azokat a régi adatokat, amelyeket csak a Mac OS kezel. Miután a Mac OS feltelepült és működött, a telepítőkorongot betettem a CD-meghajtóba, rákattintottam, és kerestem a telepítőprogram ikonját. Döbbenten tapasztaltam, hogy az ikon nincs ott. Kezdtem pánikba esni, és a másik három CD-n is kerestem, de sikertelenül. A telepítőprogram nem volt a CD-n! Természetesen letölthető lett volna a LinuxPPC webhelyéről, de a Mac OS alaptelepítésem nem tartalmazta a hálózati összetevőket.

Ismét a nyomtatott kézikönyv segített ki. A 45. oldalon leírják, mi a teendő, ha a telepítőprogram nem működik. Bizonyos fájlokat és mappákat a merevlemez bizonyos területeire kellett másolnom, és újra kellett indítanom a számítógépet. Megjelent a Mac OS-alapú BootX OS-választó. Folytathattam a munkát.

Beállítottam a BootX-et, hogy a grafikus telepítőt indítsa, és megnyomtam a *Boot* gombot. Innentől zökkenőmentesen zajlott a telepítés. Másfél óra múlva a LinuxPPC már futott is a G3-mon.

Rendszerindítás a G3-on

Elhatároztam, hogy a rendszert a G3-on közvetlenül indítom ahelyett, hogy a BootX-ből választanék a Mac OS 9 és az LPPC2000 között. Másfél éve ez még elképzelhetetlen lett volna. Most azonban, amikor a mindennapi munkámat egyre inkább Linuxszal végzem, készen állok arra, hogy eldobjam a Mac OS-mankót és a Linuxot használjam alapvető asztali operációs rendszerként. Ha valaha szükségem lesz a Mac OS 9-re, a Mac-On-Linux emulátor programmal X-ablakban futtathatom. Hasonlóan a zöld iMachez, a rendszermappát a telepítő CD-ről át kellett másolnom a /boot-ba, és át kellett írnom a yaboot.conf fájlt. Újraindítás után a yaboot közvetlenül a grafikus Linux bejelentkező képernyőt jelenítette meg. Az LPPC alapértelmezett grafikus felülete a Gnome, de a KDE is választható. Én a Gnome-ot hagytam meg alapértelmezettként.

A Perl-döbbenet

Bejelentkeztem rendszergazdaként és létrehoztam magamnak egy felhasználót.

A *barryp* felhasználó nevében lefuttattam néhány parancsot, hogy lássam, mi van telepítve. Az `uname -a` parancs megmutatta, hogy az LPPC2000 a 2.2.18 rendszermagot tartalmazza. Megnéztem, telepítve van-e a vi, a latex és az ispell. Ahogy vártam, megvoltak. Ezután kiadtam a `perl -v` parancsot, hogy megtudjam, melyik Perl-változat fut. Megdöbbentett a látvány. Az LPPC2000 az 5.005_03 Perl-kiadást tartalmazza, és nem az 5.6.0-t, amely már 2000 közepe óta elérhető! Kiderült, hogy

az LPPC a RedHat 6.1 csomagjaira épült, innen származott a Perl 5.005_03. A 6.1 óta a RedHat kiadta a 6.2-t, a 7.0-t, és jelenleg a 7.1 a legfrissebb változat. Bár az LPPC2000 új, mégis elavult. Mac-felhasználóként már hozzászoktam ehhez: a programok új változatai először Intel-felületre jelennek meg, csak ezután várható a maces frissítés (ha egyáltalán elkészül). Azt hittem, a Linux esetében ez másként történik. Felháborodott és csalódott voltam.
A Linux-felhasználók természetesen tudják, hogy a nyílt forráskódú programok legfrissebb kiadását bármikor letölthetik, és csak egy `tar -zxvf` parancsot kell végrehajtani. A Mac-felhasználók túlnyomó többsége számára (és ők alkotják a LinuxPPC célközönségét) azonban ez túl bonyolult.

A nyomtatás beállítása

Következő feladatom a nyomtató telepítése és beállítása volt. Ez a *Printer Configuration* vezérlőpulttal X alól könnyen megtehető. Sokszor cselekedtem már ezt, és mindig jól működött. Most azonban nem. Minden úgy nézett ki, ahogy kell, mégsem tudtam nyomtatni. Megnéztem a LinuxPPC webhelyét, és az egyik súgóoldalon azt olvastam, hogy az LPRng-t kell telepíteni. Letöltöttem az LPRng-t, és megkíséreltem telepíteni. Nem sikerült, mert hiányolta az rhs-printfilters csomagot. Ezt a csomagot is megtaláltam a Hálón, majd miután telepítettem, az LPRng telepítése is sikeresen lezajlott. A nyomtatás azonban még mindig nem működött. Rákerestem a merevlemezen a printcap fájlra. Kiderült, hogy kettőt is tartalmaz, egyet a /etc-ben és egyet a /usr/etc-ben. A fájlok tartalmának ellenőrzése után rájöttem, hogy az LPRng a /usr/etc-ben levő üres fájlt használja, míg a Printer Configuraton vezérlőpult a /etc-ben levőt. Rendszergazdaként átmásoltam a /etc-ben levő fájlt a /usr/etc-be, újraindítottam az LPRng-t, és láss csodát, a nyomtatás működött. Természetesen jobban örültem volna, ha azonnal működőképesnek bizonyul.

Egyéb hozzáadott programok

Az egyéb programokat tartalmazó CD-n jó néhány programcsomag fellelhető a LinuxPPC-hez való RPM formátumban. Kezdetektől fogva nagyon érdekelt a Bochs x86 emulátor, gondoltam telepítem, így meglátom, mire képes. A grafikus RPM-telepítővel megkerestem a csomagot a CD-n. Amikor a telepítésre kattintottam, a program hibaüzenetet írt ki, miszerint a jcarr felhasználó és csoport nem létezik a rendszerben, ezért nincs mód az RPM telepítésére (*Jeff Carr* a LinuxPPC egyik kulcsembere). A jcarr bejegyzést hozzáadtam a /etc/password és a /etc/group fájlokhoz, ezután sikerült az RPM telepítése. Komolyan úgy vélem, hogy a LinuxPPC-s szakembereknek az ilyen jellegű hibákat jóval a CD kiadása előtt kellene felfedezniük és kiküszöbölniük.
Érdekelt az x86, de nem annyira, mint a Mac OS emulátor, a Mac-On-Linux (MOL). A Bochs telepítésének gondjaiból kiindulva valami hasonlóra számítottam. Nem így lett, a Mac-On-Linux simán települt. Ez a program teszi lehetővé, hogy a Mac OS-t X-ablakban futtasd (az X előtti változatokat).

A MOL jobban végzi a feladatát, mint a Mac OS X-be épített hasonló program. A klasszikus Mac OS több változatát képes kezelni, beleértve a 8. és 9. kiadásokon alapulókat is. A Mac OS X csak a 9.1-et tudja utánozni. A MOL nagyszerűen támogatja az AppleTalk protokollt, távoli Macintoshokon lévő meghajtókhoz voltam képes csatlakozni a MOL-on belül, ami a Mac OS X-ben működésképtelen volt. A nyomtatás is jobban megy. Mindent egybevetve a MOL nagyszerű példa a nyílt forráskód jobb minőségére.

A piros iMacen nem lehetett zsugorítani

Aggódtam a piros iMacre való telepítés miatt, elsősorban azért, mert a gyerekeim soha nem bocsátották volna meg, ha a gépükön tönkreteszem a Mac OS-t.
Ideje volt hát kipróbálni az FWB Software Hard Disk Toolkit-PE programját. A piros iMacen valaha volt egy LinuxPPC 1999 Q3, de igazán sose használtam. Így a 6 GB már fel volt osztva 4 GB Mac OS és 2 giga LinuxPPC részekre. A Toolkit-PE fő használati területe a meglévő lemezrészek átszervezése, ezért a négygigás Mac OS lemezrészt háromgigásra szerettem volna zsugorítani, hogy a felszabaduló 1 GB-on létrehozhassam a / lemezrészt. A másik 2 GB lett volna a /home. Mivel a LinuxPPC-hez kapott FWB segédprogramokhoz nem volt leírás, letöltöttem egy cikket a ➲ http://www.linuxppc.org/ címről. Követtem az utasításokat, de nem tudtam a lemezrészt zsugorítani. A program arra panaszkodott, hogy nem elég friss a meghajtóprogram, és válasszam az *Update Driver* menüpontot az FWB menüből. Amikor megpróbáltam, a Toolkit-PE „2-es típusú” Mac OS hibával összeomlott. Ha a programot egy Mac OS lemezrészre kíséreltem meg telepíteni, sorozatszámot kért. A LinuxPPC-vel nem kaptam sorozatszámot, és a ➲ http://www.linuxppc.com/ webhely sem tett erről említést. Az FWB-ről szóló cikkben szóba került, hogy nem szükséges a sorozatszám, ha a programot közvetlenül az FWB rendszerindító CD-jéről futtatjuk. Én így használtam, de mindig lefagyott. Végül is a régi LinuxPPC lemezrészére kellett mindent telepítenem, ami hibamentesen lezajlott. Nem tudom, mitévő lettem volna, ha a telepítés előtt mindenképp zsugorítanom kellett volna a lemezrészt, ugyanis semmi kedvem nem volt újrafelosztani a lemezt és újratelepíteni a Mac OS-t. Szerencsémre a lemezrészek már léteztek.

Záró megjegyzések

Ez a LinuxPPC-változat nagy előrelépés az előző kiadásokhoz képest. Ennek ellenére csalódott vagyok amiatt, hogy mennyi erőfeszítésbe kerül a program telepítése. A legtöbb dolog, amelyen keresztülmentem, annyira idegen egy átlagos Mac-felhasználótól, hogy valószínűleg egyáltalán nem is foglalkoznának a LinuxPPC-vel. Az Apple egyértelműen nyerő ebben a tekintetben, ugyanis a Mac OS X telepítéséhez csak be kell rakni a CD-t, és újra kell indítani a gépet. Természetesen az általa nyújtott szolgáltatások alulmúlják a várakozásokat. A LinuxPPC védelmében megemlítendő, hogy azt állítják, ez a kiadás volt az utolsó a „nagy átszervezés” előtt. A LinuxPPC vállalat most alakul át üzleti vállalkozásból nonprofit szervezetté, ami örömteli és támogatandó lépés. A kezdeti csalódottság és a beállítási gondok ellenére a LinuxPPC most már simán fut a G3-on. Az eredmény pedig megérte a fáradságot. Nem látok okot arra, hogy másik operációs rendszerre váltsak, és állandóan figyelem a frissítések oldalát a LinuxPPPC webhelyén. A következő kiadást is nagyon várom már. De adok egy megszívlelendő jó tanácsot: olvasd el a kézikönyvet, mielőtt nekilátsz a telepítésnek, és nézegesd a LinuxPPC webhelyét segítségért, ötletekért, javításokért és frissítésekért.

Paul Barry
(paul.barry@itcarlow.ie) számítógépes hálózatokról ad elő az Institute of Technology-n az írországi Carlowban (➲ http://www.itcarlow.ie/). A cikkben említett zöld iMac ad helyet Paul honlapjának: ➲ http://glasnost.itcarlow.ie/~barryp/index.html.

Új szerkentyűk a láthatáron!

Mint várható volt, 2001 a beágyazott Linux éveként tündöklik.

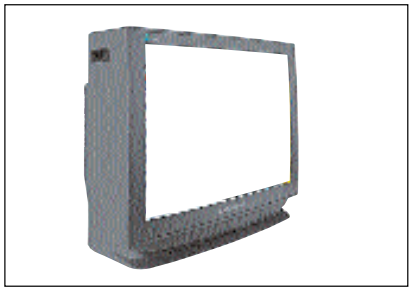
Még tavaly megjósoltam, hogy 2001 lesz az az év, amikor a Linux megjelenik a különféle intelligens eszközökben, akár a fogyasztóknak, akár a más rétegeknek szánt termékekről legyen szó. A jelek szerint jóslatom annak ellenére valóra válik, hogy a dot-com cégek nagyot buktak és a Linux-buborék kipukkadt, ami kisebbfajta gazdasági visszaesést okozott.
Három olyan, a fogyasztóknak szánt készüléket mutatunk be, amelyben beágyazott Linuxot is találunk: a Sylvania cég Internet/TV készülékét, a Memora személyi kiszolgálóját, valamint a Galleo Mobile Multimedia Communicator nevű termékét.

Sylvania Internet/TV

A Sylvania Computer Products idén nyárra tervezte új 27 hüvelykes digitális tévéjének bevezetését, mely televízióként és internetkészülékként egyaránt használható.
A készülék – a belső egylapkás PC-n futó Linuxnak köszönhetően – tanítható, és új mérföldkövet jelent a televíziógyártásban, hiszen az első olyan tévék egyike, amelybe internetkészüléket is építettek. Felvirradt az Internet/TV korszaka.
A Model SPC2700iHD névre hallgató Sylvania-termék a következő felszereltséggel dicsekedhet:

- nagy felbontású, HDTV-re is alkalmas képernyő, hagyományos tévés beépített hangolóegységgel;
- beépített, internetezésre alkalmas számítógép, mely telefonos – 56 kb/s sebességű – és széles sávú (ethernet) elérésre egyaránt képes;
- Linux-alapú operációs rendszer fut rajta könnyen használható, grafikus felhasználói felülettel, mely egyszerre tesz lehetővé tévénézést és internetezést;
- a rendszer irányítása egyszerűen kivitelezhető az összeépített távirányító-billentyűzet-egér kezelőegység segítségével;
- internetalapú vezérlési és adatbázis-szolgáltatásokkal rendelkezik.

Az Internet/TV a műsorszórás vagy a kábeltévé jeleit a beépített hangolóegységen keresztül fogadja, de nagy felbontású képernyőként külső mozgóképforrásokhoz is használható, így képmagnóhoz, DVD-lejátszóhoz, DSS-hez, illetve egyéb kábeles készülékekhez, vagy éppen számítógéphez (legfeljebb 800×600-as SVGA-felbontással). A rendszerben egy 266 MHz órajelű National Semiconductor Geode processzor található, mely 64 MB RAM-mal és 16 megányi



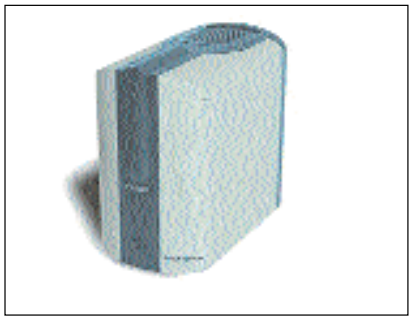
flashmemóriával rendelkezik. A ki- és bemeneti kapuk hosszú listáján infravörös vevőt találhatunk, mely az egybeépített kezelőegységgel való kapcsolattartáshoz szükséges. Továbbá felfedezhetünk egy infravörös adót, amellyel külső eszközökre továbbíthatunk adatokat, ezenkívül két USB-kaput, egy 100 Mb sebességű ethernetcsatlakozót, az 56 kb/s sebességű modem csatlakozóját, kompozit video ki- és bemeneteket jobb és bal oldali hangcsatornákkal, S-Video bemenetet kétoldali hangcsatornákkal, végül egy nagy felbontású RGB videobemenetet, melyen a készülék HDTV- vagy számítógépes SVGA-képet tud fogadni.
A Sylvania Internet/TV igazi ereje a belső program és a Ch.1 Inc. által biztosított, hálózaton keresztül elérhető szolgáltatások együttes használata során mutatkozik meg. A Ch.1 adta át a Sylvaniának a készülék és programok fejlesztéséhez szükséges támogatást és hátteret, ugyanakkor Internet/TV-szolgáltatóként is működik. A készülék úgynevezett Ch.1-képességeit használva tévét nézhetünk, böngészhetünk az Interneten, webes és tévés könyvjelzőket hozhatunk létre, személyre szabott műsorfüzetet állíthatunk össze, előre megadott szakportálokat érhetünk el, elektronikus leveleket küldhetünk, cseveghetünk, vásárolhatunk vagy éppen MP3-zenét hallgathatunk. A készülék arra is lehetőséget ad, hogy a Picture-in-Portal-módszer segítségével tévézés közben elérjük a műsorhoz tartozó honlap interaktív tartalmát, ezenkívül a Ch.1 webalapú vezérlőoldalán a képmagnót vagy a merevlemezes felvevőt is kezelhetjük. A képernyőn megjelenő gombok segítségével könnyedén hozzáférhetünk az Internet/TV felhasználói kézikönyvéhez, a termékhez kapható kiegészítők jegyzékéhez, az ügyfélszolgálathoz vagy a terméktámogatáshoz.
A tévékészülékekhez külön vásárolható, kizárólag internetezésre alkalmas internetkészülékekkel (set-top-box) ellentétben a Sylvania Internet/TV lehetővé teszi, hogy teljes 800×600-as SVGA-felbontással bóklásszunk a Hálózaton. Ha nem akarunk lemondani a teljes billentyűzet nyújtotta kényelemről, külön kiegészítőként megvásárolhatjuk. Az új készülék ablakot nyithat az Internetre a fogyasztók azon, közel ötven százalékot kitevő rétegének is, akik nem rendelkeznek saját számítógéppel. Ebben pedig az a legizgalmasabb, hogy mindannyian – akár tudtukon kívül – Linux-felhasználók lesznek!

➲ http://www.sylvaniapc.com/2Support/support.html (*1. kép*)
➲ http://www.ch1.com/

Servio személyi kiszolgáló

Az újonnan induló Memora Corporation azzal a reménnyel kezdte meg tevékenységét, hogy Linux-alapú Serviójával új termékosztályt sikerül létrehoznia: a *személyi kiszolgálókét*. Az elnevezés egy könnyen használható, kissé a számtalan feladatot ellátni képes háztartási gépekre hasonlító eszközt takar, ami egyesíti magában azokat a szolgáltatásokat, amelyekre egy korszerű, jó „összeköttetésekkel” rendelkező háztartásnak szüksége lehet: átjáró, tűzfal, vezetékes és vezeték nélküli kiszolgáló, levelező-kiszolgáló, valamint multimédia- és egyéb állományok tárolását végző megosztott tárhely is fellelhető lenne. A készülék nem csak a fenti képességekkel rendelkezik: az Interneten keresztül biztonságos külső kapcsolatot létesíthetünk vele elektronikus postaládánkhoz, weboldalakhoz és további

állományokhoz. A Memora szerint a személyi kiszolgáló úgy működik, mint egy „olyan egyedülálló hozzáférési pont a felhasználó digitális adatainak rendezésére, elérésére és megosztására, amit bárki akkor és ott használ, amikor és ahol csak akar”.



A Servio személyi kiszolgáló lényegében egy kisméretű és egyedi kinézetű Linux-alapú számítógép, amely alig különbözik egy átlagos asztali géptől, ám működéséhez nem igényel billentyűzetet és képernyőt. Belsejében 600 MHz órajelű – vagy gyorsabb – Intel Celeron processzor, 128 MB memória és harminc gigabájtos merevlemez kapott helyet. A ki- és bemenő adatforgalom két USB- és Fast ethernetkapun keresztül – 10/100 Mb/s sebességgel – bonyolítható le. Az egyik ethernetkapu jellemzően egy DSL-modemhez, míg a másik a házi (helyi) hálózathoz csatlakozik. A két USB-kapu segítségével a készülék vezeték nélküli helyi hálózathoz vagy egyéb támogatott külső felülethez kapcsolható.
A Servio operációs rendszere a Linux, de néhány más nyílt forrású program is fut rajta – köztük Apache webkiszolgáló, MySQL és az Exim levelezőkiszolgáló –, továbbá több Memora fejlesztette program vezérli a készülék beállításait és működését. Ez utóbbi programok böngészőalapú alkalmazások, így többek között az elektronikus levelezést, az adatbázis- és a webes szolgáltatásokat egyesítik.
A rendszer olyan gyári beállításokkal vásárolható meg, amelyekkel műszakilag képzetlen felhasználók is azonnal használni tudják: egyszerűen be kell iktatni a széles sávú kapcsolat – DSL vagy kábel – és a helyi vezetékes vagy vezeték nélküli hálózat közé. A telepítés második lépése a készülék bekapcsolása, ezt követően a beállítóprogram önműködően fel fogja ismerni a hálózati beállításokat. Végül valamelyik PC-s böngésző segítségével adjuk meg a nevünket, valamint jelöljünk ki egy nyilvános nevet a készülékeknek. Ha minden jól megy, azonnal létrehozhatjuk a barátok, ismerősök és családtagok személyes internetelérését, továbbá fényképeket, mozgóképeket, zenéket és egyéb állományokat oszthatunk meg, de a készülék egyéb szolgáltatásait is igénybe vehetjük.
➲ http://www.memora.com/ (*2. kép*)

Galleo Mobile Multimedia Communicator

A Galleo cég nemrég jelentette be Linux-alapú Mobile Multimedia Communicator készülékét. Számos újonnan felbukkant vezeték nélküli kapcsolatteremtésre képes kézi számítógéphez hasonlóan ez is ötvözi a zsebtitkárok, az internetkészülékek és a mobiltelefonok tudását. A Galleo tehát telefonként, internet-

elérésre, webböngészésre, személyi adatkezelésre, multimédiára (MP3-lejátszó, mozgóképfolyamok), játékra és további feladatokra egyaránt használható, de az IPsec-megfelelő VPN hálózati biztonságot is támogatja.
Yovav Meydad, a Galleo termékmenedzsere szerint a szabadalmazott mobil távközlési eljárások olyan „csomagkapcsolt” adatátviteli lehetőséget kínálnak, melyek „versenyképes választást nyújtanak a WAP-hoz képest”, és „a felhasználóknak ugyanazt a webböngészési élményt nyújtják, mint az asztali gépek, miközben mozgásszabadságukat nem korlátozzák”. A készülék programja egynegyed VGA-méretű kijelzőn is képes az átlagos weboldalak megjelenítésére, melyekre ráközelíthetünk vagy tetszőleges irányba gördíthetjük. További értékes segítség a hálózaton elérhető tartalmak módosítások nélküli megtekintéséhez a Java virtuális gép.
A webböngésző támogatja a HTTP 1.1, HTML 4.0, XML 1.0, JavaScript, SSL, TLS, RSA, VPN és Personal Java 1.2 JVM szabványokat és programozási nyelveket, valamint képes png-, gif- és jpeg-képek megjelenítésére. A már említett MP3-lejátszón és mozgóképfolyam-kezelő alkalmazásokon kívül a készüléken egy sor, a személyes adatok kezelését segítő – levelező, naptár, feladatlista, telefonkönyv, jegyzettömb és összehangolt megosztott adattároló – alkalmazás is megtalálható.
A felhasználói felület egy kézírás-felismerő program révén támogatja az érintőképernyő segítségével végzett adatbevitelt, de használhatjuk a képernyőn megjelenő billentyűzetet is, illetve beépített gombok is a rendelkezésünkre állnak. A játékot és a webböngészést a beépített botkormánnyal tehetjük még egyszerűbbé és élvezetesebbé.
A gép lelke egy 206 MHz-es Intel StrongARM SA-1110 típusú, az egész rendszert egyesítő processzor, mely 32 MB rendszermemóriával és 16 megányi nem felejtő flash-memóriával gazdálkodhat. A kijelző erős fényű, színes, 320×240 képpont felbontású TFT LCD. A készülék ki- és bemeneti csatlakozói között egy RS-232 soros, egy USB, egy infravörös (IrDA) kaput és fülhallgató csatlakoztatására alkalmas sztereoaljzatot találunk. További memóriát és biztonsági kártyát két külön aljzatba helyezhetünk. A beépített mobiltelefon-egység támogatja a kétnormás GSM-szabványt, a GPRS-t, a TriCodecet, valamint 14,4 kb/mp sebességű faxküldésre vagy adatátvitelre alkalmas. A Galleo – az észak-amerikai piacra készülve – a CDPD-szabvány támogatását is tervezi, de készülékét először az európai piacon kívánja kipróbálni, mert a GSM- és GPRS-hálózatok iránt ott mutatkozik nagyobb igény.
➲ http://www.galleo.com/ (*3. kép*)

Rick Lehrbaum (rick@linuxdevices.com) hozta létre a LinuxDevices.com „beágyazott Linuxok portálját”, amely nemrég tagja lett a ZDNet Linux Resource Centernek. Rick 1979 óta foglakozik beágyazott rendszerek fejlesztésével. Társalapítója az Ampro Computersnek, alapító tagja a PC/104 Consortiumnak, és fontos szerepet játszott abban, hogy elindulhatott az Embedded Linux Consortium.

A rendszermag finomhangolása

Assembly nyelvű programrészletek akár negyvenszázalékos sebességnövekedést is eredményezhetnek az Alpha-processzorokon.

A nagy teljesítményű kiszolgálók vezető fejlesztője az API Networks. Vevőinkkel együtt kifejezetten érzékenyek vagyunk a rendszer teljesítményére. Legtöbb vásárlónk Alpha-alapú rendszert futtat, ezért módfelett odafigyelünk a kulcsfontosságú nyílt forráskódú összetevőkre és csomagokra, hogy a gép által nyújtott teljesítmény ne maradjon kihasználatlan.

A kereskedelmi termékek zárt forrású világában a programok kiadása után felfedett hibák és a további fejlesztések mindig csak a következő változatban jelennek meg. A nyílt forrású programok világában a két fejlesztés közti idő órákban vagy napokban mérhető. Ráadásul a finomhangolásra szolgáló megoldásokat a minél fejlettebb közösségi tudásbázis létrehozása érdekében hamar elterjesztik. Az előző szempontok figyelembevételével vizsgáltuk meg a Linux-rendszermag assembly nyelvű programrészleteit. Kutatási témánk a legújabb Alpha 21264 (más néven ev6) processzor mikrofelépítésének lehető legjobb kihasználása volt.

Felfedeztük, hogy ha a 21264 tulajdonságait a Linux-rendszermag egyes felületfüggő részeinek átírásával kihasználjuk, jelentős sebességnövekedést érhetünk el, amely bizonyos feladatoknál akár negyven százalékig is elmehet. Írásunkban bemutatjuk, hogyan és miért működnek ezek a továbbfejlesztések, így tapasztalatainkat mások is felhasználhatják Alpha-alkalmazásaik felgyorsítására. Ezek a finomhangolások például egy Alpha 21264-en futó levelezőkiszolgálót nagyobb terhelés elviselésére teszik képessé, a teljesítmény pedig átlépheti azokat a határokat, amelyeknél a gép válaszideje általában megnövekedne. A végfelhasználók a rendszert gyorsabbnak fogják érezni, és a rendszergazdák szintén elégedettek lesznek, mert nem kell új gépet venni, mégha a forgalom növekszik is.

A Linux-rendszermagban a teljesítményre nagy befolyással bíró programrészleteket gondosan (és kényelmes módon) külön felületenként kezelik, és többnyire assembly nyelven valósították meg. Számos fejlesztő munkálkodik a rendszermag teljesítményének algoritmikus továbbfejlesztésén, emellett jó páran foglalkoznak azzal is, hogy megfelelő érdeklődés, tudás és tapasztalat birtokában az assembly programrészleteket finomhangolják, ezáltal a processzorból a lehető legnagyobb teljesítményt hozzák ki. A legtöbb fejlesztő az x86-os rendszerekhez tartozó kódot fejleszti, a megfelelő Alpha-kódhoz azonban előttünk jó ideje senki sem nyúlt.

A szájhagyományból és a mérésekből arra következtetésre jutottunk, hogy a web- és más hálózati kiszolgálók futtatásakor a rendszer aránytalanul sok időt tölt a Linux-, illetve az Alpha-rendszermagban. Mivel a teljesítményre nagy befolyással bíró programrészletek száma a Linux-rendszermagban viszonylag kicsi (20–30 assembly nyelvű programrész), átfésülésük érdemesnek tűnt annak kiderítésére, hogy a kódokat vajon a 21264 figyelembe vételével írták-e meg. A programrészek gyors átolvasása után azonnal kiderült, hogy gondosan, kézzel megtervezett kódról van szó, amelyet az Alpha-processzorok előző nemzedékéhez, a 21164-eshez (más néven ev5) készítettek.

A különbség jelentéktelennek tűnhet, azonban a 21264-es lényegesen eltér a 21164-től a mikrofelépítés (lapkamegvalósítás) szintjén, ennek köszönhetően azonos órajelen megközelítőleg kétszeres a teljesítménye. Mielőtt a teljesítménynövekedés részletezésébe bocsátkoznánk, célszerű áttekinteni a 21164-es és a 21264-es közötti legfontosabb különbségeket (*lásd 1. táblázat*).

Miután felismertük, hogy a gondos újraírás jelentős teljesítménynövekedést eredményezhet, hozzákezdtünk a Linux-rendszermag Alpha assembly nyelvű kódrészleteinek finomhangolásához. (Akadtak kísérletek annak a feladatnak dinamikus és önműködő megoldására, hogy a régebbi Alpha-processzorokra fordított binárisokat az újabb Alpha processzorokon hatékonyan lehessen futtatni, de ez a téma jelentősen túlmutatna írásunk keretein.)

Közelebbről nézve világossá vált, hogy körülbelül 20 programrészt a linux/arch/alpha/lib, 4–6 programrészt pedig a linux/include/asm.alpha könyvtárban szükséges átírni ahhoz, hogy teljesen kihasználjuk a 21264-es tulajdonságait.

1. táblázat **Az Alpha 21164-es és 21264-es processzorok összehasonlítása**

21164 (ev5, ev56)	21264 (ev6, ev67)
Kétutas	Négyutas
Egyszerű elágazásvizsgálat	Fejlett elágazásvizsgálat
Közvetlen leképezésű átíró gyorstár	Kétszeresen asszociatív gyorstár
Egyszerű utasításütemezési szabályok	Bonyolult utasításütemezési szabályok
	A sávszélesség nagyjából kétszerese a 21164-esének
	Új utasítások: WH64, CTLZ, CTTZ

A forrás fejlesztése

Belső nézőpontból szemlélve a 21264-es lényegesen összetettebb és hatékonyabb processzor, mint a régebbi 21164-es.

A különbségekből adódóan a teljesítmény növekedésének alapját a következő (később részletezett) átalakítások képezik:

- a kód átütemezése, hogy a processzor ne álljon meg utasításbetöltés közben;
- az elágazások elkerülése és az elágazások célcímeinek helyes megválasztása;
- ahol csak lehet, az ismétlési csapdák (replay traps) elkerülése;
- a 21264-es utasításkésleltetési és ütemezési szabályainak alkalmazása;
- a 21164-esben és a 21264-esben meglévő, eddig használaton kívüli utasítások alkalmazása;
- az 21264-esben újonnan megjelent utasítások használata és;
- lehetőség szerint bizonyos utasítások mellőzése a 21264-en.

Az utasításbetöltés leállásának (fetch stalling) megakadályozása

Ha az utasításbetöltés leállását el szeretnénk kerülni, biztosítanunk kell, hogy az utasítások a betöltési tömbben ne próbáljanak egy vagy több végrehajtóegységnél a rendelkezésükre álló kapacitásnál többet igénybe venni. A 21264-esnél az is fontos, hogy az elágazások célcímei a betöltési tömb határára legyenek igazítva. Mivel négy utasítást tölt be egyszerre, a 0mod4 utasításokat igazítani kell a csomópontokhoz, ami egyenértékű a 0mod16 igazításával.

Az elágazásokból származó lassulás elkerülése

A 21264-esen különösen fontos, hogy elkerüljük az elágazásokból származó lassulást. A bonyolult, tanítható elágazás-feltérképező beépített logika csak akkor működik hatékonyan, ha csupán egyetlen vezérlésátadó utasítás található a betöltési tömbben (a „négyes csomagban”). A 21164-eshez igazított rendszermagban több vezérlésátadó utasítás számos helyen előfordult egy négyes csomagon belül. Ráadásul az elágazások célcímei 8mod16 címekre voltak igazítva, amely gyakran azt eredményezte, hogy a cél címkéje a négyes csomag közepére került. Ezek az utasítássorozatok elég jól futnak a 21164-esen, ám a 21264-esen viszonylag lassúak.

Az ismétlési csapdák elkerülése

Ismétlési csapda akkor lép fel, amikor a processzor a soros feldolgozáshoz kénytelen visszaállítani a memória állapotát, kikényszerítve bizonyos memóriaterületek elérését, vagy amikor különböző méretű hozzáférési kérések érkeznek ugyanarra a memóriaterületre. A módosított programrészletekben ilyen gond nem akadt, ebből a szempontból nem kellett módosítani.

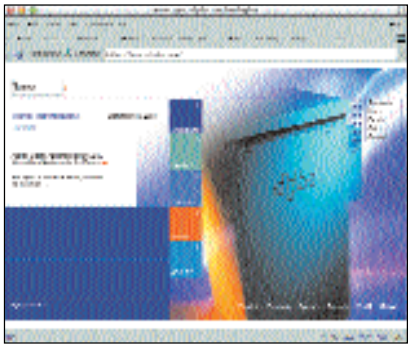
A 21264-es utasításainak késleltetése

A 21264-es utasítások ütemezésének szabályai túl bonyolultak ahhoz, hogy itt közöljük őket, akit azonban érdekelnek a részletek, a „21264 Compiler Writer’s Guide” című írásban utánanézhet.

Igénybe vehető, de használaton kívüli utasítások

A bájt- és szóméretű betöltések, valamint kiírások a 21164A (ev56) processzorban jelentek meg először, de az eredeti assembly programrészletekben ezeket nem használták. Előző tapasztalataink (az alkalmazások statikus binárisainak fordítása kapcsán) arra utaltak, hogy a teljesítmény – ha kihasználjuk ezeket az utasításokat – általában 10–20 százalékkal nő. Különösen igaz ez az stw (store word – „szó tárolása”) és az stb (store byte – „bájt tárolása”) utasításokra, mert ezek olyan módon küszöbölik ki a memóriaforgalmat, hogy nem lép fel ismétlési csapda. A rendszermag finomhangolásánál is hasznosnak bizonyultak ezek az utasítások, rendszerint a nagy területmásolások végére kerültek, míg az adatmozgatás nagy részét a nyolc bájtot egyszerre mozgató betöltő és kiíró utasítások végezték.

Alpha-felületen sokféle előbetöltő utasítás létezik. Az előbetöltő utasítások arra készítetik a memóriaalrendszert, hogy egy memóriatömböt későbbi feldolgozás céljából az adatgyorstárban helyezzen el. Ezek általában nem jelennek meg a lefordított kódban, mivel néhány fordítóprogram elég ismerettel rendelkezik ahhoz, hogy létrehozásuk szükségtelenné váljon. Például a Compaq fordítóprogramja létrehozza az előbetöltő utasításokat. Amikor nagy mennyiségű adatot kell mozgatni, az assembly nyelven programozó használhatja az előbetöltést (és ajánlatos is használnia). A gcc-ben az __asm__() lehetővé teszi, hogy a programozó a program kulcsfontosságú pontjaiba illessze be a megfelelő előbetöltő utasításokat, amennyiben a program megírása tiszta assemblyben nem kívánatos. Mivel ezek az utasítások csökkenthetik vagy megakadályozhatják az adatgyorstár leállását, használatukkal jelentős sebességnövekedés érhető el.



Újonnan elérhető utasítások

A 21264-es az első olyan Alpha-megvalósítás, amely támogatja a következő három, a teljesítménynövelés szempontjából hasznos utasítást: CTLZ, CTTZ és WH64.

A CTLZ és a CTTZ utasítások megszámolják egy 64 bites regiszter kezdő, illetve a végen elhelyezkedő nulláinak számát, de a karakterlánc-műveletek végrehajtása során is hasznosak. Amikor a program mintakeresést magában foglaló karakterlánc-műveletet hajt végre (az strlen a NULL-ra keres rá), gyakran előfordul, hogy a mintakeresés bájtnagyságú indexe egy nyolcbájtos regiszterben van. A CTTZ nélkül körülbelül tíz utasítást kellene végrehajtani, többek közt több CMOVxx-t (feltételes áthelyezés), hogy ezt az indexet meghatározzuk. Az eredmény egyrészt a kód méretének csökkenése (mindig hasznos), másrészt kevesebb órajelciklus szükséges a karakterlánc-műveletek végrehajtásához. Akad még néhány alacsony szintű fájlrendszerművelet, ahol ezek az utasítások ugyancsak jól használhatók a bitmezők lyukainak megkereséséhez.

A WH64 (write hint for 64-bytes – „felkészülés 64 bájt írására”) a memória-alrendszert értesíti, hogy a közeljövőben egy megadott 64 bájtos terület kerül kiírásra. A processzor ezt az értesülést átadhatja a memória-alrendszernek, amely érvényteleníthet a célterület tartalmát, és megtakaríthat néhány memóriaciklust a memóriaállapot folytonosságának fenntartásával. Mivel a folyamatok váltása nagy mennyiségű adat mozgatásával jár a memória egyik részéből a másikba,

2. táblázat **Terheléspróba eredményei a 2.2.18-as rendszermaggal**

2.2.18 fordítása	make -j 1	make -j 2	make -j 4	make -j 8
Foltozatlan: rendszermag	27,7	35,0	36,3	36,8
Foltozott: rendszermag	23,9	30,8	32,3	32,7
Gyorsulás	15,9%	13,6%	12,4%	12,5%

3. táblázat **Terheléspróba eredményei gcc-2.95.3-mal**

gcc-2.95.3 fordítása	make -j 1	make -j 2	make -j 4	make -j 8
Foltozatlan: gcc	50,2	65,3	67,4	70,2
Foltozott: gcc	43,7	56,8	59,1	61,3
Gyorsulás	14,9%	15,0%	14,0%	14,5%

4. táblázat **Terheléspróba eredményei a 2.4.2-es rendszermaggal**

2.4.2 fordítása	make -j 1	make -j 2	make -j 4	make -j 8
Foltok eltávolítva: rendszermag	22,5	23,1	23,3	23,6
Alap (foltozott): rendszermag	20,2	20,7	20,3	20,3
Gyorsulás	9,8%	11,6%	14,8%	16,3%

5. táblázat **Terheléspróba eredményei a gcc-vel és a 2.4.2 rendszermaggal**

gcc fordítása	make -j 1	make -j 2	make -j 4	make -j 8
Foltok eltávolítva: gcc	44,9	48,3	50,2	51,1
Alap (foltozott):	39,8	40,8	42,4	43,6
gcc				
Gyorsulás	12,8%	18,4%	18,4%	17,2%

6. táblázat **Terheléspróba eredményei az UP1000 rendszeren 2.4.0-test6-os rendszermaggal**

2.4.0-test6 fordítása	make -j 1	make -j 2	make -j 4	make -j 8
Foltozatlan: rendszermag	25,0	25,5	26,7	34,4
Foltozott: rendszermag	18,0	18,2	18,7	26,1
Gyorsulás	38,8%	40,1%	42,8%	31,8%

7. táblázat **Terheléspróba eredményei az UP1000 rendszeren gcc-vel és 2.4.0-s rendszermaggal**

gcc fordítása	make -j 1	make -j 2	make -j4	make -j 8
Foltozatlan: gcc	74,0	57,8	62,5	71,5
Foltozott: gcc	63,7	47,2	52,4	61,2
Gyorsulás	16,2%	22,5%	19,3%	16,8%

minden sebességnövekedés jól jön a rendszermag és a felhasználói memória között. A másik terület, amely nagymértékben a memória-memória forgalmának sebességétől függ, az operációs rendszerbeli programok betöltési ideje. A program bitjeit mind le kell képezni, és az összes kinullázott memóriába (a .BSS végrehajtható állományokban) nullákat kell írni.

Elkerülendő utasítások

A 21264-es a CMOVxx feltételesáthelyezés-utasítást úgy valósítja meg, hogy azt a processzor belsejében két külön részre bontja. Ennek eredményeként a CMOVxx utasítás késleltetése legalább két órajelciklus, de akár öt is lehet, attól függően, hogy az adott betöltési tömbben hány CMOV található. Bizonyos esetekben sebességnövekedés érhető el, ha a CMOV utasításokat jól kiszámítható feltételes elágazásokkal helyettesítjük. Jó ökölszabálynak tűnik, hogy amennyire csak lehetséges, csökkentsük a CMOV-utasítások számát.

Az adatgyűjtés és a kipróbálás módszere

Az adatokat egy API Network CS20 kiszolgálóról gyűjtöttük. A gép kiépítése: két 833 MHz-es processzor 4 MB DDR gyorstárral, 1 GB SDRAM és Ultra-160 SCSI lemez. Két terhelési próba futott le: ötször a 2.2.18 rendszermag fordítása és ötször a gcc-2.95.3 fordítása. Az átlagos rendszeridőt (`/usr/bin/time -p`) vettük figyelembe a `make` különböző párhuzamosításai mellett (lásd a *2.* és a *3. táblázatot*). Hasonló kísérletet végeztünk a 2.4.2 rendszermaggal (az összes sebességnövelő folt használatával). Az eredményeket a foltozatlan 2.4.2 rendszermaghoz hasonlítottuk, amelyből a legtöbb (de nem az összes) teljesítményfokozó változtatást eltávolítottuk.

Ezt a kísérletet eredetileg egy API Network UP1000 alaplappal rendelkező rendszeren hajtottuk végre, amelyben 700 MHz-es processzor volt 4 MB gyorstárral, 128 MB SDRAM és IDE lemez. Ismét öt rendszermag- és öt gcc-fordítást futtattunk, majd feljegyeztük a futási idők átlagát. A 2.4.0-test6-os rendszermagot foltokkal és anélkül is használtuk.

A szerény kiépítettségű rendszeren (az UP1000-n) a teljesítménynövekedés látványos, abban az értelemben, hogy csökkent a rendszermagban töltött idő, és bizonyos tevékenységeknél (a rendszermag fordítása) a sebességnövekedés a negyven százalékot is elérheti. A nagyvonalúbban kiépített CS20-as rendszeren a mért terheléseknél következetesen 14-15 százalékos sebességnövekedést észleltünk.

Az UP1000 és a CS20 közötti különbségeket a memóriának tulajdonítottuk: az UP1000 800 MB/s átviteli sebességgel és 64-bites sínnel, míg a CS20 2,65 GB/s átviteli sebességgel és 256-bites sínnel rendelkezik.

Összegzés

Ilyen vagy olyan formában az összes újraírt programrész része a 2.4.2 rendszermagnak (bizonyosakat ismét újraírtak). A 2.2.17-eshez is készítettünk foltot, és vállalatunk weboldalán a ➲ http://www.api-networks.com/products/downloads/developer_support/ címen a „Performance” pont alatt elérhetővé tettük. További erőfeszítések eredményeként a fejlesztések bekerültek a `glibc`-be, és ott segítik a felhasználó módban futó programok sebességnövelését.

Rick Gorton az API NetWorks szakembercsapatának tagja. A Linuxszal először az eredeti 32-bites Alpha-változat (BLADE) formájában találkozott. 1992 óta fejleszt Alphára bináris fordítókat, teljesítménynövelőket és egyéb binárisokat kezelő eszközöket. A rick.gorton@api-networks.com címen érhető el.

Ingyenes adatvédelem az új lapkákkal

Az egyre növekvő biztonsági igények és zuhanó lapkaárak mellett a Linux-rendszereknek is fel kell készülniük a jövő kihívásaira.

Mindenki szereti biztonságban tudni a titkait, de vajon mennyit hajlandó ezért anyagilag áldozni? Mostanra az Interneten a magánszféra védelme meglehetősen kicsire zsugorodott, a piacon megjelenő új lapkák azonban lehetővé teszik a felhasználók számára, hogy adataikat költségkímélő módon vagy éppenséggel pluszkiadások nélkül védjék meg.
Két szabvány uralkodik döntő többséggel a Hálón: az SSL és az IPsec. Az előbbit a legtöbb böngészőprogramba beépítik, ily módon biztosítva a biztonságos internet-tranzakciók megvalósulását. Az IPsec pedig virtuális magánhálózatokat (VPN) hoz létre, amelyeken keresztül a távoli adatbázisok is biztonságosan érhetők el.
Mindkét szabvány használ titkosítást, hogy a kódfeltörők, hallgatózók és betolakodók elől elrejtse a kényes adatokat. A tavalyi évig az Egyesült Államok kormánya megtiltotta a nagy biztonságú titkosítási módszerek más országokba történő exportját, de azóta a kormány szigora enyhült e tekintetben. A titkosítás általános elterjedésének legnagyobb akadályát jelenleg az a mérhetetlen nagy számítógépes kapacitásszükséglet képezi, amely az üzenetek titkosításához, illetve megfejtéséhez kell. Ha a titkosítás informatikai szempontból nem volna nehéz feladat, a kódokat egyszerű lenne feltörni. Szerencsére a legtöbb korszerű számítógép központi egysége bővelkedik annyi „lóerőben”, hogy lassabb ethernetkapcsolaton vagy akár széles sávú internetkapcsolaton is képes legyen titkosítani az üzeneteket. A gondot a kiszolgáló számítógép jelenti, amelynek egyidejűleg kell tudnia nagyszámú, sok felhasználótól érkező üzenetet kezelni. A kiszolgálógépek jellemzően 100 Mbites úgynevezett Fast ethernet- vagy ennél is gyorsabb hálózaton küldik az adatokat.
A szűk keresztmetszet megszüntetése érdekében a cégek olyan önálló titkosítóberendezések mellett döntöttek, mint például a VPN-gépek vagy az SSL-kártyák. Ezek a gépek sajátos, a célnak megfelelően kialakított lapkákat használnak, amelyek a titkosítási számításokat a szokványos processzoroknál sokkalta gyorsabban végre tudják hajtani.
A megfelelően nagy sebesség eléréséhez a jelenlegi biztonsági lapkák még nem eléggé gyorsak, ezért előfordulhat, hogy a nagyobb VPN- és SSL-hálózatokban több ilyen drága berendezést is össze kell kapcsolni – további járulékos berendezésekkel együtt. A végeredmény egy olyan VPN-csúcsgép, amelynek ára több százezer dollárra rúghat.
A segítség azonban máris úton van! A biztonság iránti fokozott érdeklődés újabb cégeket ösztönzött arra, hogy képviseltessék magukat a biztonsági lapkák piacán, ahol ez idő tájt tíz cég van jelen, de továbbiak is várhatók.
A lapkák többségét tavaly a Hifn állította elő, amely a programalapú tömörítéssel foglalkozó Stac cég melléküzletága. Olyan lapkagyártó óriások, mint az Intel, a Motorola és a Philips szintén betörni készülnek a piacra.
A kereskedelmi verseny tehát igen kedvező a műszaki újítások számára. Nem meglepő, hogy legjavuk olyan kis cégektől származik, mint a Chrysalis-ITS, a SafeNet és a Securealink; valamint az újonnan indulóktól, mint a Corrent, a NetOctave és a BlueSteel (ez utóbbi a Broadcom biztonságtechnikával foglalkozó leányvállalata). A piaci versenynek köszönhetően a jövő év közepére olyan 10 Gbit/s sebességgel működő lapkák jelennek meg, amelyek hálózati környezetben akár a legszélesebb keresztmetszeteken is képesek lesznek kiszolgálni.
Ez a sebesség százszorta nagyobb, mint a tavalyi év kezdete óta kapható legnagyobb teljesítményű lapka sebessége, ami valóban látványos fejlődés. A Moore-törvény átlagos fejlődési ütemével összemérve ez azt jelenti, hogy tízévnyi fejlődést két évnél egy kicsivel hosszabb időre sűrítettünk össze. Már csak egy rövidebb időszakra van szükség, amíg az új, gyorsabb lapkák a rendszerekben általánossá válnak, és jövőre a mostani VPN-csúcsgépek is csupán néhány ezer dollárba fognak kerülni.
Ha a titkosítás ilyen olcsóvá válik, valójában nem is különálló gépeken fog működni: ezek a nagy sebességű lapkák minden hálózati kártyán, illetve minden számítógépes hálózatban fel fognak tűnni. Az internetszolgáltatók csekély költséggel (körülbelül 1 dolláros havidíjért) kínálnak majd védelmi szolgáltatásokat a felhasználók számára. Ezért az árért a legtöbb ember biztonságosan fog tudni elektronikus levelet küldeni és fogadni, böngészni, illetve további elektronikus tevékenységet folytatni a Világhálón.
Néhány elemző azt jósolja, hogy 2004-re az internetes adatforgalomnak nagyjából a fele titkosított lesz.
Számos esetben ez csak kis vagy éppenséggel semmilyen hatást sem fog az alkalmazásokra kifejteni. Az IPsec az IP-szabvány részeként a hálózati verem 3-as rétegén dolgozik, vagyis nem egyszerűen az alkalmazási réteg alatt, hanem még a TCP-nél is lejjebb. Azt követően, hogy az operációs rendszer két számítógép között biztonságos kapcsolatot létesített, a két gép közti teljes adatforgalom titkosításra, illetve feloldásra kerül anélkül, hogy az az alkalmazásokra bármilyen hatást is gyakorolna.
Az SSL magasabb szintű protokoll, amit az alkalmazásnak közvetlenül kell kezelnie. Minthogy azonban már be van építve a böngészőprogramba, az azon keresztül elért bármilyen szolgáltatás felhasználhatja. A felelősség a webes rendszergazdákat terheli annyiban, hogy webfelületük minél nagyobb részét nagymérvű biztonságot szavatoló kiszolgálókra kell telepíteniük. Amint a biztonsági költségek csökkennek, a webhelyek teljes biztonsága ugyancsak megoldható.
Az IPsec használata akkor nagy fogás, ha az operációs rendszerben adunk neki helyet. A Linux-felhasználók már használhatják a FreeS/WAN-környezetet, amely az IPsec nyílt forrású, elismert megvalósítása. Ezzel szemben a legtöbb Linux-változat még nem tartalmazza ezt a programot, jóllehet a helyzet könnyen megváltozhat, amint a titkosítás egyre nagyobb népszerűségre tesz szert.
Összehasonlításképpen megemlíteném, hogy az IPsec szerepel a Windows 2000 szokásos szolgáltatásai között. A titkosításért felelős lapkák árának zuhanásával együtt a Linux-közösségnek is készenlétben kell állnia.
A fejlesztőknek, terjesztőknek és végfelhasználóknak egyaránt fel kell készülniük arra, hogy rendszerük képes legyen az olcsó titkosítási berendezések által nyújtott előnyök kihasználására.

Linley Gwenapp
(linley@linleygroup.com)
a The Linley Group szakmai elemző cég alapítója és vezetője.
A cég honlapja elérhető a ➲ http://www.linleygroup.com címen.

Betörésérzékelés mindenkinek

Telepítsd a Tripwire-t, hogy elkaphasd a betörőket!

Bár reménykedünk benne, hogy megerősített rendszerünk védelmi szempontból áthatolhatatlan, százszázalékos biztonság nem létezik. A réseket már a rendszer védelmi vonalán fel kell ismerni. A Tripwire Open Source szabadon felhasználható és nyílt forráskódú programcsomag, amely a lehetséges réseket meglehetősen jó eséllyel már megnyitásuk pillanatában felfedezi.
A Tripwire-hoz hasonló sértetlenség-ellenőrzők a titkosításban használt módszerekkel „ujjlenyomatot” vesznek a rendszer futtatható bináris állományairól, beállításokat tartalmazó fájljairól és minden másról, amelyek a behatolás során vagy annak következtében megváltozhatnak. Ezután ezeket a fájlokat időről-időre összevetik a tárolt ujjlenyomatokkal, és a különbségeket levélben elküldik a rendszer gazdájának.
A Tripwire a legjobban ismert és a legérettebb sértetlenség-ellenőrző rendszer, írásunkban ezt tárgyaljuk részletesen. Érdekes lehet még az Advanced Intrusion Detection Environment (AIDE) is, amely több felületen fut, mint a Tripwire Open Source, és az FCheck, és amelyik teljesen Perlben íródott, ezért még az AIDE-nél és a Tripwire-nél is kevésbé felületfüggő (ráadásul Windows alatt is fut). A cikk végén a *Kapcsolódó címek* részben találhatók a hivatkozások az AIDE és az FCheck weboldalaira.

Mi a sértetlenség-ellenőrzés?

A sértetlenség-ellenőrzés olyasmi, mint a biztonsági mentés: remélhetőleg sosincs rá szükség, de az ég óvjon attól, hogy éppen akkor ne álljon a rendelkezésünkre, amikor szükségünk lenne rá. Akárcsak a biztonsági mentés, a sértetlenség-ellenőrzés is egy nagyobb terv része. Ha rendszerünket megerősítettük, megfoltoztuk, és a legmagasabb szinten karbantartjuk (vagy legalább a józan ész elvárta szinten gondját viseljük), akkor a sértetlenség-ellenőrző biztosítja a végső védőhálót, amely felfogja a betörő támadását, bármilyen okos legyen is.
A sértetlenség-ellenőrző működési elve egyszerű: ha egy fájl váratlanul megváltozott, jókora az esélye, hogy egy behatoló változtatta meg. Az első dolgok egyike például, amelyet a betörők a rendszergazda jogainak megszerzése után meg szoktak tenni, az, hogy a gyakran használt rendszerprogramokat (például ls, ps és netstat) olyanokra cserélik le, amelyek látszólag a szokásos módon működnek, de elrejtik azokat a fájlokat, folyamatokat vagy kapcsolatokat, amelyek lebuktathatnák őket.
A sértetlenség-ellenőrzők segítségével létrehozhatunk egy ellenőrzőösszeg-adatbázist a fontos rendszerprogramokról, beállítási fájlokról vagy bármi másról, amitől nem várjuk vagy nem akarjuk, hogy megváltozzon. Ha rendszeresen összevetjük ezeket a fájlokat a sértetlenség-ellenőrző adatbázisával, rendszerünk feltörésének esélyét csökkenthetjük. Minél hamarabb szerez tudomást a rendszergazda a betörésről, annál nagyobb az esélye, hogy elkapja vagy legalább elkergeti a behatolókat.
Figyelem, a legjobb sértetlenség-ellenőrző sem ér semmit, ha nem megbízható az adatbázisa! Nagyon fontos, hogy a lehető leghamarabb létrehozzuk ezt az adatbázist az operációs rendszer biztonságos forrásból történő telepítése után. Megismétlem, a sértetlenség-ellenőrző telepítése, beállítása és fenntartása nem éri meg a fáradtságot, ha nem tiszta rendszeren hozták létre az adatbázisát.

Tripwire – az első és mindmáig a legjobb sértetlenség-ellenőrző

Számos más ünnepelt és hasznos dologgal együtt a Purdue COAST Project (➲ http://www.cerias.purdue.edu/coast/) „ajándékozta” a világnak a *Dr. Eugene Spaffort* és *Gene Kim* által írt Tripwire-t is. Eredetileg nyílt forrású, szabad program volt, majd 1997-ben kereskedelmi termék lett, és a díjmentes használat az akadémiai, illetve egyéb nem kereskedelmi célú felhasználásra korlátozódott.
Szerencsére tavaly októberben a Tripwire Inc. megjelentette a Tripwire Open Source Linux Edition névre hallgató terméket. A Tripwire kereskedelmi változatai ezelőtt olyan lehetőségeket tartalmaztak, amelyek az Academic Source Release-ből hiányoztak. Ezzel szemben a Tripwire Open Source a kereskedelmi termék többé-kevésbé friss változatának felel meg. Az olyan vállalati környezetben kihasználható lehetőségeket leszámítva, mint a sok gépből álló rendszer központi karbantartása, sokban hasonló a Tripwire for Servers termékhez.
A Tripwire Open Source csak a nem kereskedelmi Unixokra ingyenes, például Linuxra és Free/Net/OpenBSD-re. Tulajdonképpen csak a RedHat Linux és a FreeBSD támogatott, de semmi akadálya, hogy más Linux- és BSD-változatokon is ugyanolyan jól lefordítható és futtatható legyen. Csak a régebbi Academic Source Release használható szabadon a kereskedelmi Unixokon, például Solarison vagy IBM AIX-en, egyébként meg kell vásárolnunk a kereskedelmi változatot. Írásunk további részében a Tripwire Open Source Linux Edition változatával fogok foglalkozni.

A Tripwire beszerzése, lefordítása, telepítése

A cikk írásának pillanatában a Tripwire Open Source legújabb változata: a 2.3.1-2. A forráskód `tar`-csomagként letölthető a ➲http://sourceforge.net/projects/tripwire/ weboldalról. Javaslom, hogy ezt a csomagot töltsd le, fordítsd le és telepítsd. A Tripwire története során csupán egyetlen jelentős biztonsági hibája akadt (az is csak egy szolgáltatás-megtagadási kockázat volt). Ezt a programot azért használjuk, mert üldözési mániánk van. Az üldözési mániások számára csak a legújabb (megbízható) változat a megfelelő. Az eddig elmondottak figyelembevételével megemlítem, hogy a RedHat 7.0-ban található bináris változat szintén meglehetősen új. Amennyire meg tudom állapítani, a RedHat v2.3-55 RPM-je és a hivatalos forrás v2.3.1-2-változata között csak két, a biztonságot nem érintő hibajavítás történt, tehát valószínűleg nem vállalsz óriási kockázatot, ha a RedHat 7.0 RPM-jét telepíted. De ne mondd, hogy én nem szóltam!
A Tripwire Open Source lefordításához helyezd a forráscsomagot a /usr/src könyvtárba, és csomagold ki:

```
tar xzvf ./tripwire-2.3.1-2.tar.gz
```

Ezután ellenőrizd, hogy megvan-e a rendszereden a közvetett hivatkozás a `/usr/bin/gmake`-ről a `/usr/bin/make`-re (GNU-make ugyanis nem minden Unixnak része a, ezért a Tripwire kifejezetten a `gmake`-et keresi – természetesen a legtöbb Linux-rendszeren ez a `make`). Ha nincs meg, a következő paranccsal létrehozható: `ln -s /usr/bin/make /usr/bin/gmake`.
Ezenkívül ellenőrizni kell az alkönyvtárak teljes rendszerét a

`/usr/share/man` alatt, ugyanis a Tripwire a `man4`, a `man5` és a `man8` alá akar majd súgóoldalakat telepíteni. Az én Debian-rendszeremen hiányzott a `/usr/share/man/man4`, ezért a telepítő létrehozott egy /usr/share/man/man4 nevű fájlt, amely a súgóoldalt tartalmazta ahelyett, hogy az ilyen nevű könyvtárba másolta volna. Végül el kell olvasni a forrás README és INSTALL fájljait, belépni a forrásfa src könyvtárába (például /usr/src/tripwire-2.3.1-2/src), és a változók értékeit az src/Makefile-ban szükség szerint módosítanunk kell. Győződj meg róla, hogy a megfelelő SYSPRE sor elől törlöd-e le a megjegyzésjelet (SYSPRE = i386-pc-linux, SYSPRE = sparc-linux stb.)! Felkészültünk a fordításra, tehát írd be a `make release` parancsot. Mivel ez jó darabig eltart, közben akár egy szendvicset is bekaphatsz. Amikor a fordítás elkészült, a könyvtárszerkezetben lépj eggyel feljebb, például /usr/src/tripwire-2.3.1-2, továbbá add ki a következő két parancsot:

```
cp ./install/install.cfg .
cp ./install/install.sh .
```

Kedvenc szövegszerkesztődben nyisd meg az install.cfg-t, és ha az alapértelmezett útvonalakat rendben találod, vizsgáld meg a *Mail Options* részt. A Tripwire-nak itt adhatod meg, hova küldje a naplóbejegyzéseit. Ha a TWMAILMETHOD=SENDMAIL beállítást választod, a Tripwire a megadott helyi levelezőt (alapértelmezés szerint a `sendmail`-t) használja arra, hogy jelentéseit elküldje egy helyi felhasználónak vagy csoportnak.
A TWMAILMETHOD=SMTP beállítást választva, és a TWSMTPHOST és TWSMTPPORT értékeit megadva a Tripwire a külső levélcímre küldi el a jelentéseit a megadott SMTP-kiszolgálón és -kapun keresztül. Ha később meggondolnád magad, a Mail Options beállításait bármikor megváltoztathatod.
Ha a rendszert – amelyre a Tripwire-t telepíted – több felhasználó is használja, továbbá valaki rendszeresen belép és olvassa a rendszergazda leveleit, akkor a SENDMAIL-módszer a megfelelőbb. Ha a rendszer karbantartása egy másik gépen keresztül távolról történik, az SMTP-módszer a jobb.
Miután kedved szerint kitöltötted az install.cfg-t, ideje telepíteni a Tripwire-t. Írd be: `sh ./install.sh`. Két jelszót fog kérni a program, a *site* jelszó a Tripwire beállítási fájljait védi, a *local* pedig a Tripwire adatbázisát és jelentéseit. Így lehetséges egységes házirendet alkalmazni több gépre, míg az adatbázis karbantartásának és a jelentések létrehozásának felelősségét el lehet osztani.

Megjegyzés az RPM-ekről

Az RPM-ek telepítése egyszerűbb és gyorsabb (de jegyezzük meg ismét, hogy a legfrissebb Tripwire-változat nem feltétlenül érhető el ebben a formában). Miután felraktad az RPM-et, az egyetlen dolog, amit meg kell jegyezned, az, hogy le kell futtatnod a `/etc/tripwire/twinstall.sh` parancsot, amellyel létrehozhatod a `site` és `local` jelszót. A parancsfájl a forrás `install.sh` parancsfájljához hasonlóan működik, lásd az előző bekezdést.

A Tripwire használata

Amennyire hasznos a Tripwire, olyannyira elterjedt róla az a hír, hogy nehéz beállítani (ez természetesen minden sokoldalú bonyolult eszközre igaz). A helyzet valójában nem olyan rossz; ha követed azokat az egyszerű lépéseket, amelyeket itt megadok, akkor hatékonyan használhatod a Tripwire-t a rosszfiúk elfogására. Lépj be hát a felhasználók előkelő csoportjába, akik nemcsak telepítették, hanem használják is a Tripwire-t!

A beállítófájl kezelése

Első feladatunk a beállítófájl, a tw.cfg ellenőrzése.
Ezt a fájlt a telepítő ugyan titkosította, de kényelmünk kedvéért twcfg.txt néven egy szöveges változat is bekerült a /etc/tripwire könyvtárba. Itt kell megváltoztatnod azokat a beállításokat, amelyekkel kapcsolatban a telepítés óta meggondoltad magad.
Ha valamit megváltoztattál, a beállítófájlt a következő paranccsal titkosítsd újra:

```
twadmin --create-cfgfile --site-
➥keyfile/site.key twcfg.txt
```

Ebben az esetben a site.key a telepítés idején létrehozott kulcs, és a twcfg.txt az a szöveges beállítófájl, amit az imént szerkesztettél és titkosítani szeretnél. Ez magától értetődőnek tűnhet, hiszen ezek a fájlok alapértelmezett nevei, de bármi másnak is elnevezheted őket. Ne mulaszd el megadni a kulcsfájlt, különben a `twadmin` hibát jelez (és ne feledd, a gyakorlat célja a beállítófájl titkosítása).
Figyelem, soha ne hagyd meg a Tripwire beállító- (tw.cfg) és házirend- (tw.pol) fájljainak szöveges változatát a merevlemezeden. A szerkesztés és a titkosítás után töröld a szöveges változatokat, később bármikor helyreállíthatod őket a következő paranccsal:

```
twadmin --print-cfgfile > mycfg.txt
```

ahol természetesen a mycfg.txt-t bármivel helyettesítheted. Bár nem ismertettem még a Tripwire binárisait (jobb őket a megfelelő környezetben tárgyalni), gondolom már kitaláltad, hogy a `twadmin` segítségével kezelheted a Tripwire beállításait, kulcsait és (legalábbis kezdetben) a házirendfájlokat.

A házirendfájlok kezelése

A házirendeket a Tripwire beállítófájljaihoz hasonlóan szövegfájlként szerkeszthetjük, telepítés előtt azonban alá kell írni és titkosítani kell őket. A beállítófájllal ellentétben a `twadmin` programot egy adott rendszeren csakis az első telepítéshez használjuk, a továbbiakban a Tripwire programot alkalmazzuk a házirendfrissítési módban.
A Tripwire telepítése után első ízben a következő paranccsal hozhatjuk létre a házirendet:

```
twadmin --create-polfile twpol.txt
```

ahol a twpol.txt annak a szöveges házirendfájlnak a neve, amelyet telepíteni szeretnénk.
Ahogy már a beállítófájloknál is említettük, szöveges házirendfájlt

sehol ne hagyj a rendszereden. Ha később olvasni vagy módosítani akarod a házirendet, visszaállíthatod a következő paranccsal:

```
twadmin --print-polfile > mypol.txt
```

A mypol.txt bármi lehet, a házirend szöveges változatának elnevezése a te dolgod.

Házirend létrehozása vagy szerkesztése

Kezdődjék a komoly varázslat! A Tripwire számára a házirendfájl olyan, mint az ember számára az agy: ez mondja meg, mit keressen és mit tegyen vele. Kissé barátságtalan a felhasználókhoz, talán nem annyira, mint a sendmail.cf, de azért készülj fel néhány rövidítés bemagolására.

A Tripwire Open Source tartalmaz egy alapértelmezett házirendfájlt, természetesen ezt is használhatod a sajátodként. Mivel azonban az alapértelmezett házirendet olyan RedHat-rendszerre dolgozták ki, amelyre szinte minden telepítve van, használatbavétel előtt a házirendfájlt alaposan át kell szerkesztened.

Ejtsünk egy-két szót a finomhangolásról. Ha a házirend nem ellenőriz elég fájlt, vagy nem nézi meg eléggé az ellenőrzött fájlokat, akkor a Tripwire nem teljesítheti a feladatát és a gondok észrevétlenül megmaradhatnak. A másik véglet sem szerencsés: ha a Tripwire nagyon odafigyel azokra a fájlokra, amelyek megváltozhatnak, „hamis találatokat” fog jelezni, melyek elterelhetik a figyelmedet a valódi gondokról.

Nem valószínű, hogy mindent elsőre sikerül értelmesen beállítani. Szinte biztos, hogy időről-időre igazítanod kell a házirenden, különösen az első sértetlenség-ellenőrzés futtatása után. Ezért, még ha pontosan ugyanazt a RedHat-rendszert használod is, amelyre a Tripwire Open Source házirendjét tervezték, meg kell tanulnod a Tripwire-házirend formai követelményeit. Munkára fel!

A házirendfájl szerkezetét úgy fogom elmagyarázni, hogy egy működő házirendfájlt kezelhető darabokra osztok. Az első darab a mintafájl legelejéről származik és néhány változónak ad értéket:

```
WEBROOT=/home/mick/www;
CGIBINS=/home/mick/www/cgi-bin;
TWPOL="/etc/tripwire";
TWDB="/var/lib/tripwire";
```

Kevesebbet kell gépelnünk, ha ezeket a változókat használjuk. A következő darabban néhány érdekesebb változónak adunk értéket:

```
BINS        = $(ReadOnly) ;  # Binárisok, amelyek
                             # nem változhatnak
SEC_INVARIANT = +tpug ;      # Könyvtárak, amelyek
                             # tulajdonosa/engedélyei
                             # nem változhatnak
SIG_MED        = 66 ;        # Fontos, de nem
                             # rendszerkritikus fájlok
```

Az első változócsoporttal ellentétben – amelyek egyszerű útvonal-rövidítések voltak – ezek kicsit érdekesebbek. Az első sorból kiderül, hogyan lehet egy változónak egy másik változó értékét megadni, hasonlóan a `bash`-héj formai követelményeihez, de itt zárójelbe kell tenni a második változó nevét.

A második sor adja meg a „tulajdonságmaszkot”. A tulajdonságmaszkok azoknak a fájltulajdonságoknak a rövidítései, amelyeket a Tripwire megvizsgál. Mivel a tulajdonságmaszkok sokszor érthetetlen és kezelhetetlen alakúak, a legtöbb ember változónevekkel hivatkozik rájuk. A Tripwire számos előre megadott változóval rendelkezik, amelyek a gyakran előforduló tulajdonságmaszkoknak felelnek meg.

1. lista Példa házirendfájl

```
WEBROOT=/home/mick/www;
CGIBINS=/home/mick/www/cgi-bin;
TWPOL="/etc/tripwire";
TWDB="/var/lib/tripwire";
BINS= $(ReadOnly) ;    # Binárisok, amelyek
                       # nem változhatnak
SEC_INVARIANT = +tpug ;# Könyvtárak, amelyek
                       # tulajdonosa/engedélyei
                       # nem változhat
SIG_MED= 66 ;          # Fontos, de nem
                       # rendszerkritikus fájlok
# Mick weboldala
(
  rulename = "MickWeb",
  severity = $(SIG_MED),
  emailto = mick@uselesswebjunk.com
)
{
  $(TWPOL)             -> $(Readonly) ;
  $(WEBROOT)           -> $(ReadOnly) (recurse=1) ;
  !$(WEBROOT)/guestbook.html ;
  $(CGIBINS)           -> $(BINS)      ;
  /var/log/httpd       -> $(Growing)   ;
}
```

Az első sor ezek egyikét tartalmazza, a *ReadOnly* tulajdonságmaszk azokra a fájlokra illik rá, amelyek semmilyen módon nem változhatnak, mint például a futtatható fájlok. Ha eljön az ideje, a tulajdonságmaszkokat részletekbe menően tárgyalni fogjuk.

A harmadik sor nevet ad a szigorúsági szintnek. Fontosságuk szerint a szabályok között szigorúsági szintekkel tehetünk különbséget. Amikor a `tripwire` parancsot a `--severity N` kapcsolóval adjuk ki, csak az `N`-nél nagyobb vagy egyenlő szigorúságú szabályok kerülnek értelmezésre. Ha nem használjuk ezt a kapcsolót, az összes szabály érvényes lesz. Amennyiben egy szabálynak a szigorúsági szintje nincs megadva, alapértelmezés szerint nulla lesz. Az ilyen szabály csak akkor kerül értelmezésre, ha a `--severity` kapcsolót nem használjuk. Most már tudunk egyet s mást a házirend változóiról, valamint használatukról, lássuk ezután a szabályokat!

```
# Mick weboldala
(
  rulename = "MickWeb",
  severity = $(SIG_MED),
  emailto  = mick@uselesswebjunk.com
)
{
 $(WEBROOT)    -> $(ReadOnly)(recurse=1) ;
  !$(WEBROOT)/guestbook.html ;
  $(CGIBINS)   -> $(BINS)      ;
  /var/log/httpd     -> $(Growing)   ;
}
```

Ez meglehetősen nagy falat, ezért kezdjük a szabályszerkezettel! A szabályok állhatnak egyedül vagy a közös jellemzők által meghatározott csoportban. Ez a lista több közös tulajdonsággal (zárójelben) rendelkező csoportot mutat (kapcsos zárójelek között). A szabálycsoport neve „MickWeb”, a csoport szigorúsági szintje 66, és a csoport-

1. táblázat **Tulajdonságmaszk-értékek**

Tulajdonság	*Leírás*
-	A következő tulajdonságok figyelmen kívül hagyása
+	A következő tulajdonságok feljegyzése és ellenőrzése
a	Hozzáférés ideje
b	Kiosztott blokkok száma
c	Csomópont időbélyege (létrehozás/változtatás)
d	A fájlleírónak helyet adó eszköz azonosítója
g	Fájltulajdonos csoportazonosítója
i	Fájlleíró (i-node)
l	A fájl mérete nő („növekvő fájl")
m	Módosítás ideje
n	Hivatkozások száma (fájlleírók hivatkozásszámlálója)
p	Engedélyek és módosító bitek
r	A fájlleírók által hivatkozott eszköz azonosítója (csak eszközfájlokra érvényes)
s	Fájlméret
t	Fájltípus
u	Fájltulajdonos felhasználóazonosítója
C	CRC-32 ellenőrzőösszeg (nem biztonságos, de gyors)
H	Haval-ellenőrzőösszeg (biztonságos, de lassú)
M	MD5-ellenőrzőösszeg (biztonságos, de lassú)
S	SHA-ellenőrzőösszeg (biztonságos, de lassú)

2. táblázat **Előre megadott Tripwire tulajdonságmaszk-változók**

ReadOnly	Széles körben elérhető, de írásvédett fájlok. Érték: +pinugtsdbmCM-rlacSH
Dynamic	A felhasználók saját könyvtárának és a gyakran változó fájlok megfigyelésére használható. Érték: +pinugtd-srlbamcCMSH
Growing	Olyan fájlokhoz való, amelyek csak nőhetnek. Érték: +pinugtdl-srbamcCMSH
Device	Jó az eszközökhöz és azokhoz a fájlokhoz, amelyeknek csak a tulajdonságait kell ellenőrizni, a tartalmát nem. Érték: +pugsdr-intlbamcCMSH
IgnoreAll	Ellenőrzi a fájl meglétét vagy hiányát, de semmi mást. Érték: pinugtsdrlbamcCMSH
IgnoreNone	Mindent ellenőriz. Egyéni maszkok létrehozására használható, például mymask = $(IgnoreNone) -ar; Érték: +pinugtsdrbamcCMSH-l

tal kapcsolatos jelentések a mick@uselessjunk.com címre érkeznek. A tulajdonságokat vesszővel választjuk el, továbbá az összes szabály pontosvesszővel ér véget.

Tulajdonságokat az egyes szabályokhoz külön is hozzárendelhetünk. A csoport első szabályánál a `recurse 1`-re van állítva, ami azt jelenti, hogy a /home/mick/www könyvtárat egy szintmélységig kell ellenőrizni (azaz a könyvtárat magát és közvetlenül alatta mindent, de semmi többet). Megjegyzendő, hogy a könyvtárakat a program alapértelmezés szerint teljes mélységükig ellenőrzi, ha a `recurse` tulajdonság alapértéke *True*.

A szabálykifejezésben megadott tulajdonságok általában felülbírálják szabálycsoport felett zárójelben megadottakat. Kivétel az `emailto` tulajdonság, amely összeadódó vonással bír: ha a csoportnak létezik közös levélcíme és a csoport egyik szabályának másik levélcíme, akkor az ahhoz a szabályhoz tartozó jelentéseket mindkét címre elküldi. Mindössze néhány tulajdonság létezik: `rulename`, `severity`, `emailto` és `recurse`. A tulajdonságokról további ismereteket a *Kapcsolódó címek* részből kiindulva szerezhetsz.

A MickWeb-csoport tulajdonságai után néhány szabály következik. Figyeld meg, miként használjuk a változókat az objektumok (a fájlok és a könyvtárak a Tripwire szóhasználatában) és a tulajdonságmaszkok megadására. Valójában egyik szabály sem használja a tulajdonságmaszkok „kézzel írott" formáját. Ez a bevett gyakorlat, amely elfogadható.

Közvetlenül az első szabály alatt, amely a Tripwire-rel közli, hogy WWW könyvtáram első szintjét írásvédettként kezelje, egy felkiáltójellel kezdődő kifejezés található. Ezt a kifejezést megállási pontnak hívják, feladata a szabály alól való kivételek megadása. Ebben az esetben a megállási pont arra utasítja a Tripwire-t, hogy ne vegye figyelembe a /home/mick/www/guestbook.html változásait. A tulajdonságok nem vonatkoznak a megállási pontokra (és nem is szükséges megadni őket).

Íme, itt egy teljes házirendfájl (legalábbis műszaki értelemben, mivel egyáltalán nem ellenőrzi a rendszer binárisait – az igazi házirendek sokkal hosszabbak). Az *1. listán* a szétvágatlan változatot olvashatjuk.

Talán észrevetted, hogy az egész fájl csak egy közvetlen hivatkozást tartalmaz egy tulajdonságmaszkra: a SEC_INVARIANT változó értéke `+tpug`. Mit is értsünk ez alatt?

A tulajdonságmaszk fájl- vagy könyvtártulajdonság, amelyet az adott objektumra nézve ellenőrizni kell vagy figyelmen kívül kell hagyni. A pluszjel után következő tulajdonságokat kell ellenőrizni, a mínusz utániakat pedig figyelmen kívül kell hagyni. A tulajdonságok rövidítései az *1. táblázatban* olvashatók.

A Tripwire leírása részletesen tárgyalja ezeket a tulajdonságokat. Ha nem ismered valamelyik titokzatos fájltulajdonságot (például: fájlleíró hivatkozások számlálója), akkor olvasd el Card, T'so és Tweedie „Design and Implementation of the Second Extended Filesystem" című cikkét (lásd még *Kapcsolódó címek*). Az ellenőrzőösszegekről csak annyit, hogy egy szabályban általában nem érdemes egy vagy két biztonságos ellenőrzőösszegnél többet használni, mert meglehetősen számításigényesek. Másfelől ne hagyatkozz teljesen a CRC-32 ellenőrzőösszegre, mert gyors ugyan, de sokkal könnyebb becsapni.

Ahogy már korábban említettem, a Tripwire rendelkezik néhány előre megadott (bedrótozott) változóval, amelyek tulajdonságmaszkokat írnak le. Ezek láthatók a *2. táblázatban*.

A legtöbb esetben egyszerűbb ezeket az előre megadott értékeket használni, mint saját magadnak kirakosgatni egyet. A változókat további tulajdonságokkal egyesítheted, például:

```
/dev/console   -> $(Dynamic) -u ;
# Dynamic, de az UID változhat
# Ez ugyanaz, mint a következő
/dev/console   -> +pingutd-srlbamcCMSH-u
```

Amit Mick kihagyott

A cikkben végig a Tripwire parancsainak „hosszú formáját” használtam. Minden két mínuszjellel (--) kezdődő kapcsolónak létezik „rövid” megfelelője, például a következő két parancs egyenértékű:

```
twadmin --print-cfgfile
twadmin -m f
```

Ha már belejöttél a Tripwire használatába, javaslom, hogy sajátítsd el a rövid formák alkalmazását. Ahogy *Neal Stephenson* rámutat az „In the Beginning Was the Command Line” (Kezdetben volt a parancssor) című esszéjében, az ínhüvelygyulladás a hozzánk hasonló kockafejűek számára olyan, mint a bányászoknál a fekete tüdő. Nem szeretném, ha bárki azt gondolná, hogy a kedves olvasóknál bármelyik kialakulásáért is felelős vagyok.
Kezdetben valószínűleg jobban használhatók a Tripwire angol szavakból összerakott hosszú parancsai. A Tripwire Open Source Reference Card (lásd *Kapcsolódó címek*) megfelelő táblázatot tartalmaz a Tripwire-programcsomag hosszú és rövid kapcsolóiról.

Miután létrehoztál valamit, ami értelmes házirendnek látszik, telepítened kell. A rendszer első Tripwire-házirendjét a következő paranccsal telepítheted:

```
twadmin --create-polfile policyfile.txt
```

A Tripwire beállításának utolsó lépése az adatbázis létrehozása. Fontos tudnunk, hogy semmi értelme olyan rendszeren létrehozni a Tripwire-adatbázist, amely működik egy ideje, mert lehet, hogy már betörtek rá! A Tripwire telepítése, beállítása és elindítása minél közelebb legyen az operációs rendszer telepítéséhez.
Az adatbázis létrehozásához használjuk a `tripwire` parancsot: `tripwire --init`. Ennél egyszerűbb már nem is lehetne, igaz? A `--init` kapcsolót csak új adatbázis létrehozására használd.
Ha a Tripwire-házirendet a későbbiek folyamán meg szeretnéd változtatni, célszerűbb a következő parancsokat használni:

```
twadmin --print-polfile > mypolicy.txt
  # kiírja a pillanatnyi házirendet
vi mypolicy.txt
  # változtasd meg a házirendet
tripwire --update-policy mypolicy.txt
  # telepítsd az új házirendet --
  # NE HASZN`LD ERRE A TWADMINT!
```

Tripwire-ellenőrzések futtatása

Az adatbázis telepítése után rendszeresen futtathatjuk az ellenőrzéseket, amelynek legegyszerűbb módja a `tripwire --check` parancs kiadása. Ez az összes védett fájlt összehasonlítja az adatbázissal, és a jelentést kiírja a képernyőre, valamint egy bináris fájlba. A jelentés a következő paranccsal tekinthető meg:

```
twprint --print-report --report-level N
        ➥--twrfile /path/file
```

ahol `N` egy szám 0-tól 4-ig, 0 az egysoros összefoglaló, 4 az összes részletet tartalmazó jelentés. A /path/file a legfrissebb jelentés teljes elérési útja és neve (a jelentés alapesetben a /var/lib/tripwire/report könyvtárba kerül).

Ha azt szeretnéd, hogy a Tripwire levélben értesítse a házirendben megadott személyeket, az ellenőrzést a következő módon futtasd:

```
tripwire --check --email-report
```

A jelentés ez esetben is megjelenik a szabványos kimeneten és a /var/lib/tripwire/report könyvtárban. Ha a Tripwire RPM-et RedHat 7-esre telepítetted, a rendszer már be van állítva a Tripwire elmaradhatatlan ellenőrző futtatására. Az RPM tartalmazza a `/etc/cron.daily/tripwire-check` parancsfájlt. Ha az emailto tulajdonságot a Tripwire-házirendedben használtad, akkor az utolsó előtti sort a parancsfájlban szerkeszd át, hogy így nézzen ki:

```
test -f /etc/tripwire/tw.cfg &&/usr/sbin/tripwire
➥--check --email-report
```

(A sorban a `--email-report` kapcsoló eredetileg nem szerepelt.)
A Tripwire nem sokat árul el, ha nem rendszeresen futtatod, akár kézzel, akár a cronból/anacronból, vagy a kettő valamilyen együttes alkalmazásával.

Megszegték a szabályokat, mi a teendő?

Mi történik, ha a Tripwire a szabályok megsértését jelenti? Ez rajtad múlik. A szabályok megszegése gyakran a túlságosan megszorító házirend miatt történik, és nem tényleges támadás áll a háttérben. Neked kell eldöntened, melyik veszélyes, és mit teszel ellene.
Ha csak vaklárma volt, a szabályszegés után valószínűleg frissíteni akarod a Tripwire-adatbázist, hogy a megfigyelt fájlok és könyvtárak törvényes változásainak megfeleljen. Két módszer létezik erre.
Az egyik a `tripwire` parancs futtatása frissítési módban:

```
tripwire --update
➥--twrfile/var/lib/tripwire/report/myhost-date.twr
```

Az utolsó érték az a jelentés, amelyet a frissítés alapjául akarsz felhasználni. A parancs megnyitja a jelentést a tw.cfg-ben megadott szövegszerkesztővel, ahol megadhatod, hogy a Tripwire az adatbázisában melyik megváltozott fájlt vagy könyvtárt ne frissítse. Más szavakkal, amikor kilépsz a szövegszerkesztőből, a Tripwire csak azoknak a bejegyzéseknek az ellenőrzőösszegét változtatja meg az adatbázisában, amelyek mellett „x” található. Kezdetben mindegyik mellett ott az „x”.
Íme, egy kivonat egy Tripwire-frissítés munkafolyamatából:

```
Remove the "x" from the adjacent box to prevent
updating the database with the new values for
this object.

Modified:
[x] "/home/mick/www"
```

Ha a bejegyzés mellől letörlöm az „x”-et, kilépek a szövegszerkesztőből és futtatom az ellenőrzést, a /home/mick/www változása újra a jelentésbe kerül, mert az adatbázis nem frissült. Ha a változás megfelel a házirendnek, hagyd ott az „x”-et, ha nem megfelelő, esetleg nem vagy benne biztos, töröld.
A Tripwire-adatbázis frissítésének másik módja, hogy az ellenőrzést „interaktív” módon végezzük, amely után azonnal frissítés következik. Ezért a

```
tripwire --check -interactive
```

ugyanaz, mint a

```
tripwire --check
tripwire
➥--twrfile/var/lib/tripwire/report/reportname.twr
```

de az első azzal az előnnyel bír, hogy nem kell megkeresni a jelentés fájlnevét (ez nehezen található ki, mert az időbélyeget is tartalmazza).
Ha hamis találatokat kapsz, fontold meg házirended finomhangolását. A finomhangolást a *Házirend létrehozása* vagy a *Házirend szerkesztése* fejezetben megismertek szellemében végezd.
Lépj tovább, és nehezítsd meg a betörők dolgát!
Mielőtt írásomat befejezném, szeretnék két ragyogó ötletet adni, amelyeket *Ron Forrestertől,* a Tripwire Open Source projektvezetőjétől hallottam:

1. A tw.cfg-ben mindig legyen MAILNOVIOLATIONS=TRUE, így meghallhatod a Tripwire szívverését, vagyis ha a Tripwire a cronból óránként fut, és több mint egy óráig nem kapsz jelentést, akkor tisztában lehetsz vele, hogy történt valami.
2. Mindig hagyj meg egy-két szabályszegést (mondjuk a /etc/sendmail.st-t), így a behatolónak nehezebb jelentést hamisítania. Igencsak egyszerű olyan jelentést létrehozni, amelyben nincs szabályszegés, de ha akad benne egy-két ismert szabályszegés, a hamisítás már sokkal nehezebb.

Remélem, kezdetnek elég lesz ennyi, hiszen sok mindenről még szó sem esett, mást is csak érintettünk. Higgy nekem, a munka, amit az eszköz kiismerésébe és kezelésének elsajátításába fektetsz, megéri a fáradságot. A Tripwire Open Source Manual (lásd *Kapcsolódó címek*) részletesen és jól tárgyal mindent. Sok szerencsét!

Mick Bauer (mick@visi.com) hálózati biztonsággal foglalkozó szaktanácsadó. 1995 óta a Linux elkötelezett híve, 1997 óta pedig OpenBSD prófétaként tevékenykedik. Mick minden kérdést és megjegyzést szívesen fogad.

Kapcsolódó címek

A Tripwire Open Source projekt lapjai ➲ http://sourceforge.net/projects/tripwire. Itt található a legfrissebb Tripwire Open Source forráskód és leírás.

Tripwire Open Source Manual és Tripwire Open Source Reference Card ➲ http://prdownloads.sourceforge.net/tripwire/tripwire-2.3.0-docs-pdf.tar.gz. Ez a lap kötelező olvasmánynak tekintendő (pdf formátum). Ha ez a hivatkozás nem működik, próbáld ezt: ➲ http://sourceforge.net/project/showfiles.php?group_id=3130.

A Linux-binárisok letöltési helye a Tripwire Open Source ➲ http://www.tripwire.org/.

A Tripwire Academic Source Release letöltési helye ➲ http://www.tripwire.com/downloads/tripwire_asr/index.cfml?

Advanced Intrusion Detection Environment (AIDE) ➲ http://www.cs.tut.fi/~rammer/aide.html.

FCheck ➲ http://www.geocities.com/fcheck2000/.

Perlben írt és könnyedén hordozható sértetlenség-ellenőrző: „Tripwire – The Only Way to Really Know" ➲ http://securityportal.com/topnews/tripwire20000711.html.

Cikk *Jay Beale* (Bastille-Linux fejlesztője) tollából a Tripwire Academic Source Release használatáról „Design and Implementation of the Second Extended Filesystem" címmel ➲ http://web.mit.edu/tytso/

Rémy Card, Theodore T'so és Stephen Tweedie kiváló írása a Linux ext2 fájlrendszeréről. A "Basic File System Concepts" fejezet különösen érdekes lehet a Tripwire felhasználói számára ➲ www/linux/ext2intro.html.

„Vigyél magaddal! Szobatiszta vagyok, csendes és vettem pár számítógépes leckét. Ha segítségre van szükséged a programoddal kapcsolatban mindig számíthatsz rám!"

Az XML (1. rész)

Gondolatok a hordozható adatokról, azaz hogyan készíthetünk mindenki által könnyen használható dokumentumokat.

Az XML a World Wide Web Konzorcium (W3C) ajánlása, mely egy sokkal régebbi SGML (Standard Generalized Markup Language = szabványos általánosított jelölőnyelv) elnevezésű ISO 8879/1986 szabványnak felel meg. Az SGML-megfelelőség azt jelenti, hogy minden XML-dokumentum egyben SGML-dokumentum is, de fordítva már nem igaz, azaz van olyan SGML-leírás, ami nem XML-megfelelő.

Az XML (Extensible Markup Language = bővíthető jelölőnyelv) adatleíró nyelv létrehozását a nyílt rendszerek térhódítása és az Internet tette szükségessé. A Java nyelv és az XML között érdekes hasonlóság figyelhető meg: a Java a futtatható formában hordozható programok nyelve, míg az XML a hordozható adatok készítésének az eszköze.

Az XML-dokumentumok (hasonlóan a Java-forráskódhoz) unicode-alapú szöveges karaktersorozatok, ahol az UTF-8 (tömörített unicode) az alapértelmezés, de a dokumentum elején ettől eltérő kódolási eljárást is választhatunk.

Az XML formátumú dokumentumok fokozatosan kezdik leváltani a jelenleg széles körben elterjedt ASCII-alapú szövegfájl-alkalmazásokat (például a beállítófájlokat). Ennek oka, hogy az ASCII-fájlok semmilyen leíró- és kezelőinformációt (metaadatot) nem tartalmaznak saját magukról (még az az alapvető tulajdonságuk sem őrződik meg, hogy a fájl készítője melyik kódlapot használta). Ezzel szemben az XML-alapú dokumentumok a benne tárolt adatok vagy hivatkozások (szöveg, kép stb.) értékein felül pluszinformációkkal is rendelkeznek, ugyanakkor leírásuk – hasonlóan a HTML nyelvhez – továbbra is rendelkezik azzal a kellemes tulajdonsággal, hogy szövegalapú.

Az XML-dokumentumokban az adatok értékein túl további címkéket és hivatkozásokat helyezhetünk el, amelyek utalnak az adat természetére, a dokumentum szerkezeti és tartalmi felépítésére, és a dokumentum érvényességének vizsgálatához is felhasználhatók.

Egy XML-dokumentum nem tartalmaz utalást arra nézve, hogy egy alkalmazás (például Internet Explorer 5, a továbbiakban IE5) hogyan jelenítse meg, de vannak kiegészítő W3C-ajánlások, amelyek pótolják ezt a hiányosságot.

Az XML-t úgy tervezték, hogy formális nyelvtanát könnyű legyen meghatározni, így az XML-adatfolyam elemzése egyszerű. Ez utóbbi tulajdonság az, ami miatt az XML ma szinte az egyetlen széles körben elterjedt, általános eszköz adataink hordozható formában történő tárolására és továbbítására.

Az XML ebben a megközelítésben a különféle jelölőnyelvek készítését leíró nyelv (metanyelv). XML-alkalmazásnak nevezzük, amikor XML segítségével készítünk el egy új jelölőnyelvet.

Ebben a cikksorozatban az XML összes lényeges vonatkozására kitérünk, melyek következő összetevőket jelentik:

- A jólformáltság feltételei.
- Az érvényességvizsgálat eszközei (DTD, Schema),
- A dokumentum elemzése és feldolgozása (Parse),
- A dokumentum megjelenésének meghatározása (CSS – stíluslap, XSL – bővített stíluslapleíró nyelv),

Első XML-dokumentumunk létrehozása

Annak érdekében, hogy az eddig leírtak ne hangozzanak túlságosan elvontnak, nézzünk egy példát! Legyen egy `WEBADATOK` adatbázisunk, amiben a `VKONYV` vendégkönyvtábla felépítése a következő:

```
NEV     VARCHAR(50)
EMAIL   VARCHAR(50)
DATUM   DATE
SZOVEG  VARCHAR(500)
```

Vizsgálódásunk pillanatában a `VKONYV` tábla a következő két bejegyzést tartalmazta:

Nyiri Imre	inyiri@mol.hu	2001.01.31	Üdvözletem a webmesternek!
Koller József	jkoller@mailbox.hu	2001.04.30	Tetszett a site :)

Az XML jellegzetessége, hogy az adatokat függőségi viszonyukban képes megjeleníteni, így egy XML-adatfolyam egyben mindig egy fát is meghatároz. A `VKONYV` tábla tartalmát a következő XML-formában állíthatjuk elő:

```
<?xml version="1.0" encoding="WINDOWS-1250" ?>
<VKONYV>
  <VENDEG sorszam="1">
     <NEV>Nyiri Imre</NEV>
     <EMAIL>inyiri@mol.hu</EMAIL>
     <DATUM>2001.01.31</DATUM>
     <SZOVEG> dv zletem a webmesternek!</SZOVEG>
  </VENDEG>
  <VENDEG sorszam="2">
     <NEV>Koller J zsef</NEV>
     <EMAIL>jkoller@mailbox.hu</EMAIL>
     <DATUM>2001.04.30</DATUM>
     <SZOVEG>Tetszett a site :) </SZOVEG>
  </VENDEG>
</VKONYV>
```

A fenti XML-adatfolyamot vizsgálva láthatjuk, hogy eddig két vendég írt a vendégkönyvbe. Itt az ideje, hogy elgondolkodjunk azon, miért éppen ebben a formában hoztuk létre a VKONYV tábla tartalmát, hiszen megtehettük volna például így is:

```
<?xml version="1.0" encoding="WINDOWS-1250" ?>
<VKONYV>
  <VENDEG sorszam="1" NEV="Nyiri Imre"
     ➥DATUM="2001.01.31">
     <EMAIL>inyiri@mol.hu</EMAIL>
     <SZOVEG> dv zletem a WEB
     ➥mesternek!</SZOVEG>
```

Próbaűrlap

```
<!DOCTYPE UI><UI>
<class>frmTeszt</class>
<widget>
    <class>QWidget</class>
    <property stdset="1">
        <name>name</name>
        <cstring>frmTeszt</cstring>
    </property>
    <property stdset="1">
        <name>geometry</name>
        <rect>
            <x>0</x>
            <y>0</y>
            <width>342</width>
            <height>155</height>
        </rect>
    </property>
    <property stdset="1">
        <name>caption</name>
        <string>Teszt Form</string>
    </property>
    <widget>
        <class>QPushButton</class>
        <property stdset="1">
            <name>name</name>
            <cstring>btnOK</cstring>
        </property>
        <property stdset="1">
            <name>geometry</name>
            <rect>
                <x>20</x>
                <y>20</y>
                <width>104</width>
                <height>28</height>
            </rect>
        </property>
        <property stdset="1">
            <name>text</name>
            <string>OK</string>
        </property>
    </widget>
</widget>
</UI>
```

```
  </VENDEG>
  <VENDEG sorszam="2" NEV="Koller J zsef"
      ➥DATUM="2001.04.30">
    <EMAIL>jkoller@mailbox.hu</EMAIL>
    <SZOVEG>Tetszett a site :) </SZOVEG>
  </VENDEG>
</VKONYV>
```

E többféle lehetőség annak a következménye, hogy az XML-ajánlás az adatok értelmezését (szemantikáját) és a használható megjelölő jelölőtagok (címkék: VKONYV, NEV stb.) körét nem határozza meg, így annak kialakítása tervezési megfontolásokon alapul. A második megoldás jobban épít a címkék tulajdonságmegadási lehetőségeire, melynek következtében a leíró címkék száma csökkent. A fentiek alapján belátható, hogy a VKONYV tábla tartalmát sokféle XML-adatfolyamként elő lehet állítani. A tervezés során ki kell választani azt az XML-formátumot, amelyet az adott feladat szempontjából jónak tartunk, majd el kell készíteni hozzá azokat a nyelvtani szabályokat, melyek azt egyértelműen rögzítik (például hogy a továbbiakban mi a VKONYV első változatú XML-adatfolyamát szeretnénk használni). A nyelvtani szabályok megfogalmazására a DTD (Document Type Definition = dokumentumtípus-meghatározás) nyelv szolgál, amit a későbbiekben részletesen is leírunk. Példánkból arról is meggyőződhettünk, hogy az XML-leírás valóban nem tartalmaz olyan nyelvi elemet, ami a dokumentum kinézetét határozná meg, hiszen csak a dokumentum szerkezetét, értelmezését és adatait tartalmazta.
Az első XML-dokumentumot egy vkonyv.xml fájlban tárolva az IE5 alapértelmezettként faformában jeleníti meg, ahol a +/- jelekkel lehetséges a fa ágait kinyitni vagy becsukni. (Ez Netscape 6 alatt is hasonlóan működik, csak ott nincs lehetőség az ágak kibontására, illetve bezárására, a fa teljesen nyitott állapotban jelenik meg.) A fájl első sorában a „version=1.0” megadása kötelező, a kódolásé viszont választható, de ha nem adjuk meg, akkor az alapértelmezett érték UTF-8 lesz.

XML-alkalmazások

A fentiekben már említettük, hogy az XML segítségével megalkotott jelölőnyelveket XML-alkalmazásoknak nevezzük.
Mielőtt részletesen leírnánk az XML-adatfolyammal kapcsolatos alapvető tudnivalókat, tekintsünk ki egy kicsit a világba, és nézzünk néhány példát arra vonatkozóan, mi is készül napjainkban az XML segítségével. Alaposan körbenézve az egyes programok között, látható, hogy az XML szinte már alapvető beállító- és adattároló eszközzé vált.
A Linux GNOME gnumeric és az AbiWord dokumentumtárolási formájául az egyaránt XML-t választotta. AbiWordöt használva mentsünk egy dokumentumot a „Helló Világ” szöveggel. Nézzük meg egy szövegszerkesztővel a kapott fájlt (a megjegyzéseket kivettem belőle), értelmezése már nem okozhat sok nehézséget:

```
<?xml version="1.0"?>
<abiword version="0.7.11">
<section>
<p><c props="font-size:14pt">Hell&#xf3;
      ➥Vil&#xe1;g!</c></p>
</section>
</abiword>
```

A Linux alatti egyik legjobb grafikus elemkészlet a QT, amihez létezik egy QT Designer (a QT-elemkészlet és a QT Designer a Trolltech alkotása). A QT Designerben – a Delphihez vagy a Visual Basichez hasonlóan – grafikusan lehet felhasználói felületeket készíteni, majd azokat szabványos *.ui fájlokba menteni. Amikor a QT Designer használatával egy „Teszt Form” címsorú és „frmTeszt” nevű űrlapba „OK” feliratú, „btnOK” nevű gombot tervezünk, a *listán* látható XML-fájl jön létre.
A fenti két példán kívül most csak megemlítek néhány további, elterjedt XML-alkalmazást:

- Az XHTML nyelv: a HTML nyelvet írja le, illetve kissé módosítja annak érdekében, hogy a HTML XML-megfelelő legyen.
- Az XSQL nyelv: adatbáziskezelő műveletek XML-megfelelő leírására szolgáló jelölőnyelv.
- MathML (Mathematics Markup Language).

A szabványosított XML-alkalmazások köre egyre szélesebbé válik, amit a W3C honlapján folyamatosan figyelemmel követhetünk.
Az egyik legfontosabb XML-alkalmazás az XSL (Extended Style

Language = bővített stíluslapleíró nyelv), amivel – a CSS-t helyettesítve – az XML-dokumentumok képi megjelenítésének kinézetét határozhatjuk meg.
Az XML adatleíró képességének általános volta miatt egyre gyakrabban találkozunk XML-alapú beállítófájlokkal is. Mindenkinek ajánlom tanulmányozásra a Jakarta-Tomcat web.xml nevű fájlját, ami tartalmazza a webhely aktuális kialakításának beállítását.
Ezen a ponton abbahagyjuk az XML-megfelelő jelölőnyelvek felsorolását, hiszen végtelen sok ilyen létezik, illetve készíthető. Befejezésként érdemes kiemelni, hogy a Microsoft Office termékek következő változatai lehetővé fogják tenni a dokumentumok XML-megfelelő formátumban való tárolását. A szakemberek egyre inkább úgy tartják, hogy az Interneten a HTML nyelvet a közeljövőben az XML-megfelelő jelölőnyelvek fogják felváltani, melynek egyik előhírnöke az XHTML.

Az XML-adatfolyam felépítése (XML-fájlok)

Az XML-adatfolyamot formai és nyelvtani szempontból is vizsgálhatjuk. Ebben a pontban a formai elemzés szempontjait tekintjük át. Egy XML-dokumentumot jólformázottnak (well formed) nevezünk, ha a következőkben részletezett formai feltételeknek eleget tesz.

Elemek (Elements)

Az elemek az XML-dokumentumok legalapvetőbb részei. Ilyen elem például a `<NEV>Koller J zsef</NEV>` részlet. Egy XML-elem általában a következő részeket tartalmazza:

- az elemet bevezető `<ElemCimke>` tagot, amit kisebb-nagyobb jelek között kell megadni;
- az elemhez tartozó adatot, például: Koller József (nem minden elem tartalmaz adatot);
- az elemet záró `</ElemCimke>` tagot.

A nyitó- és zárócímkék feladata, hogy megjelöljék és elnevezzék az általuk közrefogott adatot. Ezzel adataink természete és tartalma is nyomon követhetővé válik, így fontos, hogy a címkenevek kifejezők legyenek. Lényeges kiemelnünk, hogy a hagyományos XML-meghatározás nem ad olyan pontos adatábrázoló- és kezelőeszközt a kezünkbe, mint a programozásban megismert típus fogalma, azonban már létezik olyan W3C-ajánlás (XML Schema meghatározás), ami ennek a szigorú követelménynek is eleget tesz.
Időnként előfordul, hogy egy címke nem fog közre semmilyen adatot. Erre példa az, ha valakinek nincs levélcíme: `<EMAIL></EMAIL>`. Ezt a terjengős leírást az XML-ben a következő módon rövidíthetjük: `<EMAIL/>`. A címkenevekre vonatkozóan a következő szabályokat kell betartani:

- Csak betűvel vagy aláhúzásjellel kezdődhet.
- Tartalmazhat betűt, számjegyet, pontot, kötőjelet és aláhúzásjelet.
- A kis- és nagybetűk különbözőnek számítanak.

Megjegyzés

XML-dokumentumban a HTML nyelvben megszokott megjegyzést használhatjuk.

```
<!-
Ez egy XML megjegyzØs, ami formailag olyan, mint
a HTML megjegyzØs
-->
```

Ezeket a részeket az XML-feldolgozó (elemző, érvényességvizsgáló, megjelenítő) alkalmazások figyelmen kívül hagyják.

Tulajdonságok (Attributes)

Az XML-elemek tetszőleges számú tulajdonsággal rendelkezhetnek. A VKONYV XML második változatából tekintsük a következő részletet:

```
<VENDEG sorszam="2" NEV="Koller J zsef"
        ➥DATUM="2001.04.30">
```

Itt három tulajdonságot adtunk meg a `VENDEG` elem részére: sorszam, `NEV, DATUM`.
A tulajdonságok általában így néznek ki:

```
<elem nev1="ØrtØk1" nev2="ØrtØk2" ...>
...
</elem>
```

Egyedhivatkozások (entity references)

Az egyedek olyanok, mint a makróhelyettesítések. Minden egyednek egyedi névvel kell rendelkeznie. Legyen az `EgyedNev` egy egyednév, melynek meghívása a következő: `&EgyedNev;`, azaz `&` jellel kezdődik és `;`-vel fejeződik be. Ha megnézzük az AbiWord által készített XML-dokumentum `&#xf3;` részletét, ott egy, az „ó” betűre vonatkozó egyedhivatkozással van dolgunk.
Léteznek előre meghatározott egyedek, mint például a `gt` nevű. Az `&gt;` hatása olyan, mintha az XML-adatfolyamba egyből a `>` jelet írtuk volna. Az egyedek fogalma azt is lehetővé teszi, hogy bármilyen unicode karaktert helyezzünk a szövegbe. Ekkor az egyednév `#szÆm` vagy `#xszÆm` alakú, ahol az első a decimális, a második a hexadecimális megadási mód. A kis „ő” hexaunicode kódja 0x0151. Ezt mint egyedet a következő módon hívhatjuk meg: `&#x0151;`. Az XML-ajánlás lehetővé teszi, hogy saját egyedeket is létrehozzunk. Ez az `<!ENTITY myEgyed " rtØk">` szerkezettel lehetséges, amit `&myEgyed;`-vel hívhatunk meg és ugyanaz a hatása, mintha közvetlenül az `rtØk`-et írtuk volna le. Az egyedek meghatározása a jelölőnyelv nyelvtani szabályait meghatározó DTD-leírás része, ezért erre ott még visszatérünk.

Feldolgozási utasítások (processing instructions)

A feldolgozási utasítások nem részei az XML-nek. A megjegyzésekhez hasonlóan nem részei a dokumentum szöveges tartalmának, de az XML-feldolgozókkal szemben követelmény, hogy ezeket az utasításokat továbbadják az alkalmazások számára.
Megadási módjuk: `<?target name?>`. Egy alkalmazás csak azokat az utasításokat veszi figyelembe, amelyek célját felismeri, az összes többit figyelmen kívül hagyja. A célt (target) követő adatok az alkalmazásnak szólnak, megadásuk nem kötelező.

A CDATA-szakaszok

Az ebben a szakaszban lévő CDATA, azaz karakteres adat értelmezés nélkül átadódik az XML-adatfolyamot fogadó alkalmazásnak. A szakasz írásmódja a következő:
`<![CDATA[Itt van a tetszıleges sz veg]]>`. A megadott szöveg a `]]` kivételével bármi lehet.

Az XML-adatfolyam érvényességének ellenőrzése

Az előző részben azt vizsgáltuk, hogy egy XML-dokumentum mikor jólformázott. A helyes formát nagyon könnyen ellenőrizhetjük különféle XML-feldolgozó programokkal, például az IE5 böngészővel.
A vkonyv.xml jólformázott, mert az IE5 elemezni tudta, és a faszerkezetet is megjelenítette.
A jólformázottság azonban még nem jelenti azt, hogy az XML-dokumentum helyes, hiszen ha mi a VKONYV adathalmazt az első formátumú XML-felépítésben szeretnénk látni, akkor például a második formátum nem érvényes, ezért az elemzőprogramok vissza fogják utasítani.

A mai XML-elemzők a jólformázottság mellett általában az érvényesség vizsgálatát is lehetővé teszik. Az XML egyik legnagyobb erőssége, hogy saját címkeneveket használhatunk, de a VKONYV példája arra is rámutat, hogy szükség ugyancsak van egy nyelvtani szabályokat meghatározó leírásra, amit az XML-elemző program az érvényesség ellenőrzéséhez feltud használni.
Az XML-elemek egy faszerkezetet építenek fel, amelynek természetesen mindig van egy gyökéreleme. A VKONYV XML karaktersorozatban a VKONYV elem a gyökér. Ez az a pont, ahonnan az XML-dokumentum nyelvtani elemzése indul (a formális nyelvtanok START-jának felel meg). A DTD nyelven megfogalmazott nyelvtani szabályok egy karaktersorozatot alkotnak, tárolásukra két lehetőség adódik:

- egy fájlban tároljuk őket (például vkonyv.dtd);
- az XML-dokumentum elején tároljuk őket.

Tegyük fel, hogy valaki elkészítette nekünk (hiszen mi még nem ismerjük az elkészítés módját) a VKONYV XML adatfolyam első változatának mint XML-dokumentumtípusnak DTD-ben megfogalmazott nyelvtanát, és ezt a vkonyv.dtd fájlban átadta nekünk. Ekkor XML-dokumentumunk első sora után a következőt lehet beszúrni:

```
<?xml version="1.0" encoding="WINDOWS-1250" ?>
<!DOCTYPE VKONYV SYSTEM "vkonyv.dtd">
<VKONYV>
  <VENDEG sorszam="1">
...
</VKONYV>
```

A beszúrt sor a `<DOCTYPE` résszel kezdődik, a következő szó pedig az XML-dokumentum gyökéreleme. A harmadik szó `SYSTEM` vagy `PUBLIC` lehet. A sor utolsó része azt adja meg, hogy melyik fájlban tárolódnak a nyelvtani szabályok.
Amennyiben a nyelvtant nem akarjuk külön fájlban tárolni (például azért, mert nagyon egyszerű), akkor az eredeti XML-adatfolyam elejét így kell kiegészíteni:

```
<?xml version="1.0" encoding="WINDOWS-1250" ?>
<!DOCTYPE VKONYV [Ide j n az a karaktersorozat,
    ➥amit egyØbkØnt a vkonyv.dtd-ben tÆrolnÆnk]>
<VKONYV>
  <VENDEG sorszam="1">
...
</VKONYV>
```

A beágyazott DTD-szabályokat tehát a fent látható módon `[ … ]` zárójelek között adhatjuk meg.

Ennyi előzetes után tekintsük át röviden, hogyan lehet a DTD szabályrendszereket megfogalmazni.

Elemtípus-bevezetés (Element Type Declaration)

Az elemtípus-bevezetések azonosítják az elemek neveit és tartalmát. A DTD-ben ez az azonosítás nem éri el a programozási nyelvek típusfogalmának megfelelő részletezettségi szintet. A szemléletesség kedvéért nézzük meg a VKONYV XML adatfolyam első változata `<VKONYV>` elemének típusbevezetését:

```
<!ELEMENT VKONYV (VENDEG)* >
```

A fenti sor szavakban kifejtve: a `VKONYV` elem `VENDEG` típusú elemek sorozata, ahol a `*` jelöli, hogy ennek a sorozatnak 0 vagy több tagja lehet (a mi vkonyv.xml fájlunkban a sorozatnak most két tagja van).
A számosság kifejezésére a `*`-on kívül még a `+` (1 vagy több elem) és a `?` (0 vagy 1 elem) jelek szerepelhetnek. Amikor egy név mellett nincsenek írásjelek, akkor pontosan egyszer kell előfordulnia. Most nézzük meg, hogy a `<VENDEG>` elemre milyen nyelvtani szabályok állapíthatóak meg!

```
<!ELEMENT VENDEG (NEV, EMAIL?, DATUM, SZOVEG)>
```

Ez azt jelenti, hogy egy `VENDEG` elem a `NEV`, `EMAIL`, `DATUM`, `SZOVEG` elemek ebben a – és nem más – sorrendben vett szerkezetéből áll. Észrevehetünk itt egy fontos dolgot: ez a nyelvtani szabály már kizárja, hogy a második típusú VKONYV XML-folyam érvényes legyen, hiszen ott a `VENDEG` elem csak az `EMAIL` és a `SZOVEG` elemeket tartalmazhatja.
Az elemneveken felül egy különleges szimbólumot tartottak fenn a karakteres adat jelölésére, a `#PCDATA`-t. Ez teszi lehetővé, hogy például a `NEV` elem nyelvtani szabályát megadjuk:

```
<!ELEMENT NEV (#PCDATA)>
```

azaz a `NEV` elem egy elemzett karakterfüzér.

Már láttuk, hogy léteznek üres elemek is, azaz olyan címkék, amelyekhez nincs adat (tartalom) rendelve. Amennyiben a `<KEDVES_VENDEG/>` egy üres tag, ekképp adható meg hozzá a nyelvtani szabály:

```
<!ELEMENT KEDVES_VENDEG EPMTY>
```

Az `EMPTY` a kulcsszó, azt jelzi az elemzőprogramnak, hogy a `KEDVES_VENDEG` egy üres elem, így nincs további szerkezeti felépítése. Ebben a példában valószínűleg jelzőként szerepel.

Nézzünk egy másik üres elemet is: `<NEM_KEDVES_VENDEG/>`. Ekkor elképzelhető, hogy vendégkönyvünkben ezt a logikai tulajdonságot a következő XML-szerkezet segítségével fel akarjuk tüntetni:

```
<MINOSITES>                    <MINOSITES>
  <KEDVES_VENDEG/>   vagy        <NEM_KEDVES_VENDEG/>
</MINOSITES>                   </MINOSITES>
```

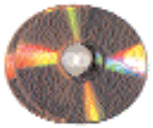
Itt a `<MINOSITES>` elem nyelvtani szabálya választható elemekből áll, amit a tartalom meghatározáskor a szűrő karakterrel fejezünk ki:

```
<!ELEMENT MINOSITES (KEDVES_VENDEG |
                    ➥NEM_KEDVES_VENDEG)>
```

A következő részben a tulajdonságlistáról, az egyedek létrehozásáról, a dokumentumok megjelenítéséről és az adatfolyam elemzéséről lesz szó.

Nyíri Imre
(inyiri@mol.hu) jelenleg a MOL Rt.-nél dolgozik. Informatikai vállalkozásában az Internet, a Linux és a Java programozás gyakorlati hasznosításával foglalkozik, ennek ellenére örök szerelme még mindig a C++. Kedveli a tudományos és a fantasztikus irodalmat, illetve filmeket (kedvencei: Solaris és 2001 – Űrodüsszeia). Szívesen sportol.

Linux teletanácskozás



A Linux a vezeték nélküli hálózatok területén is megállja a helyét.

A harmadik nemzedékbeli mobilszabványosító testület (3G) kitűzött célja az Internet- és más IP-alapú csomagkapcsolt hálózati szolgáltatások kiterjesztése a vezeték nélküli hálózatokra. Az úttörőmunka oroszlánrészét, amely még hátravan, a vállalatok átadták az Internet Engineering Task Force (IETF) szabványosítást végző testületeinek. Az IP átláthatósága következtében számos olyan szolgáltatást tesz elérhetővé a hordozható eszközökre is, amelyek már léteznek a vezetékkel Internetre kapcsolódók számára. Továbbá egyszerűsíti az ötletes és új, kimondottan vezeték nélküli szolgáltatások bevezetését. Nyilvánvaló eredményként könyvelhető el a kereslet növekedése a vezeték nélküli kapcsolatok iránt a 3G szolgáltatók ügyfelei részéről.

A sávszélesség – amely a széles sávú adatátvitel rendelkezésére áll – azonban korlátozott és költséges. A 3G-szolgáltatásokhoz szükséges hullámhosszengedélyeket az Egyesült Királyságban 36 milliárd font értéken árverezték el a hálózatüzemeltetőknek. Nemrégiben a Szövetségi Kommunikációs Bizottság (Federal Communications Commission – FCC) hasonló árverésen 17 milliárd fontért kínált hálózatot amerikai üzemeltetőknek. Ez arra ösztönzi az üzemeltetőket, hogy növeljék a sávszélesség kihasználásának hatékonyságát. Az IP hangtovábbításra való használata (például VoIP) megköveteli a hálózattól a valós idejű multimédia szállításának képességét, még akkor is, ha egyébként azon nem valós idejű átvitel zajlik. Egy teljes hullámhossz-tartományú hangkódoló – például a GSM jellegű hálózatoknál használt G721.1 – harminc ezredmásodperces hasznos jelet állít elő. Ehhez 40 bájt kombinált IP-, UDP- és valós idejű protokoll (RTP) fejlécet illeszt. Ezt a Nemzetközi Távközlési Szövetség (ITU) a multimédia továbbításáról szóló H.323 szabványában megadott protokoll írja elő. A 30 ezredmásodperc hasznos jel jellemzően 20 bájtra kódolódik, így 33 százalékos hatékonyságú. Az IETF számára ajánlott, a fejléc tömörítésére használható eljárás a fejlécet egyenletes terhelés esetén egy bájtra csökkenti, de összetettségének foka változó. A fejléctömörítési eljárásnak bitre pontos hasznos jelábrázolást, és egy, az adatokat magában foglaló IP-fejlécet kell biztosítania a mobileszközök számára. Erre az eljárásra adatvesztés nélküli tömörítésként is hivatkoznak.

A fejléctömörítés használata növeli a multimédiás távközlés hatékonyságát, különösen azóta, hogy a 90-es évek végén a Cisco által ajánlott CRTP elégtelennek bizonyult, illetve az IETF irányító bizottsága, valamint a 3GPP (Third Generation Partnership Project – a harmadik nemzedékbeli szabványok kidolgozásával foglalkozó testület) megbízhatatlannak minősítette.

Még nem dőlt el pontosan, az IETF melyik fejléctömörítési eljárást választja. A követelmények mindenesetre már kiforrtak, összefoglalásuk az *1. táblázatban* látható. Tény, hogy a jelenlegi javaslatok a követelmények nagy részét kielégítik, ennek ellenére legtöbbjüket még a magántulajdonnal kapcsolatos jogi kérdések terhelik, s ez épp az IETF fő célkitűzésével áll szöges ellentétben. Most azon gondolkodnak, hogy egy második kört is hirdetnek, s így más módszerek kristályosodhatnak ki. A jelenlegi eljárások az IP-, az UDP-, illetve az RTP-adatfolyamok tulajdonságait használják ki. A *3.* a *4.* és az *5. táblázat* az IP-, UDP- és RTP-fejlécek tulajdonságait osztályozza, a *2. táblázat* pedig az osztályozás szempontjait rögzíti.

1. táblázat **RoHC-követelmények**

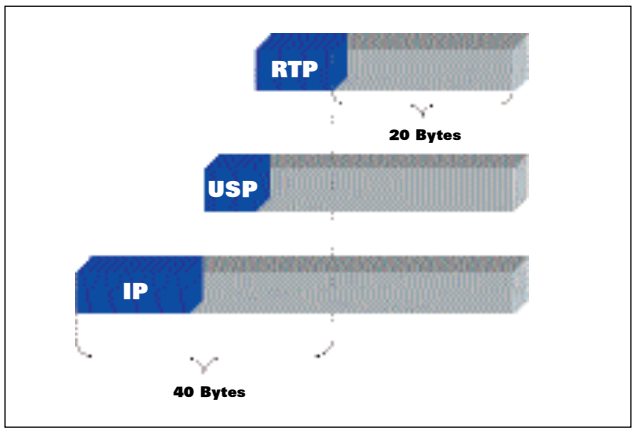
Tulajdonság	Követelmény
Működési sebesség	A lehető legkisebb késést kell biztosítania az elvárt működési körülmények között.
IPv6, vagy IPv4	Támogassa mind az IPv6-ot, mind az IPv4-et.
Összeférhetőség	Ne kelljen változtatni a meglévő IP-, UDP- és RTP-vermeken.
Cellák közötti átadás	Hatékonyan kell kezelje, minden mezőnek átláthatónak kell lennie a folyamatban.
Pontosság	A tömörítés és a kicsomagolás ne okozzon adatvesztést.
Hibaterjedés	A fejléctömörítés által okozott hibaterjedést a lehető legkisebb mértékűre kell visszaszorítani, vagy teljesen el kell kerülni.
Késleltetés	A fejléctömörítés nem növelheti meg jelentősen a késleltetés mértékét.
Csomagvesztés	A fejléctömörítő eljárás lehetőség szerint független legyen a csomagvesztési aránytól.
Támogatott formátumok	Az RTP-vel továbbított hasznos adatok típusától függetlenül működjön. Általános elv, hogy az átvitelhez ne kelljen ismerni a hasznos adatokat.
Hívástípus-függetlenség	Működjön mobil–mobil és mobil–POTS hívások esetén is.

A be- és kicsomagolás lényege

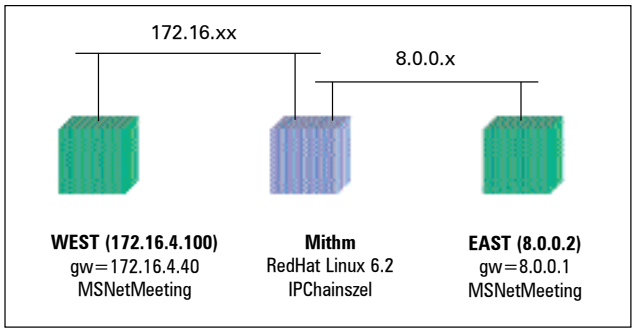
Ezen eljárások lényegét a fenti osztályozásból leszűrhető elvek képezik. Ezek a következők:

- ne küldd soha,
- közvetítsd legalább egyszer,
- közvetítsd legalább egyszer vagy frissítsd,
- közvetítsd, frissítsd, illetve frissítsd gyakran,
- biztosítsd a folyamatos terhelhetőséget,
- közvetítsd, ahogyan a csomagban van és hozd létre,
- állj készen a különbség frissítésére.

A fejléctömörítést ezen elvek szerint folytató távközlési vállalatok listája a Nokiától a Matsushitán át a Ciscóig terjed. Külön említést érdemel komoly erőfeszítéseiért az Ericsson. Az általuk kidolgozott eljárást részletesen kidolgozva benyújtották az IETF-hez. Az anyag a ➲ http://www.dmn.tzi.org.org/ietf/rohc/. címen olvasható.



1. ábra A H.323 csomag hatékonysága



2. ábra Gép a modell közepén

A megoldás

A vezeték nélküli csatorna hatásai és a tömörítő eljárás modellezhető néhány alkatrész és program segítségével. Szükséges hozzá egy linuxos gép, két H.323 jelgenerátor, valamint néhány szabadon elérhető programkönyvtár. Az egyszerű összeállítás azért nélkülözhetetlen, hogy bemutathassuk: az egész rendszer modellezhető három számítógéppel, egy pár hangszóróval és mikrofonnal, néhány hálózati kártyával és két összekötő vezetékkel. A programszükséglet is hasonlóképpen kicsiny. Ha a *2. ábrán* bemutatott összeállításban használjuk, különösen könnyen újraépíthető rendszert állíthatunk föl.
A nyílt H.323 projekt a szabványos csomagok előállításához szükséges programelemek kiváló lelőhelye. A ➲ http://www.openh323.org/ weblapon Linux- és Windows-rendszerre is két-két lehetőséget találhatunk. Azok számára, akik már rendelkeznek Windowszal, az operációs rendszer részét képező Net Meeting kínál másik, a H.323 szabványnak megfelelő multimédiás motort.
A csatorna modellezésének legfontosabb lépései:

- *Indítás* – a kapcsolat kiépítésénél nagy megbízhatóság szükséges, hogy a tömörítő–kicsomagoló pár összehangoltan működhessen, a fönt említett elvek szerint.
- *Átadás* – az átadás attól függ, hogy milyen gyorsan mozog a mobileszköz, az elvesztett csomagok száma pedig kilences értékű Poisson-eloszlást követ. Ez a viselkedés elfogadhatónak minősül, hiszen a harmadik nemzedékbeli hálózatok lehetővé a kapcsolat átadását annak újraindítása nélkül teszik.
- *Elhalkulás* – ez a vezeték nélküli kapcsolattartás legnagyobb ellensége. Az elhalkulások jellemzően a rádiójelek pillanatnyi leárnyékolásának számlájára írhatók. Okozhatja a túlzsúfolt területeken tapasztalható káros interferencia is. Elméleti szemszögből ezek a fő korlátozó tényezői a harmadik nemzedékbeli mobil hálózatok működésének.

2. táblázat **A fejlécmezők osztályozása**

Statikus-ismert	Soha nem szükséges közölni.
Statikus-meghatározó	Statikusként kezelt adatfolyam-meghatározási mező.
Statikus-állandó	Legalább egyszer közvetítendő.
Változó	Látszólag véletlenszerű és előre nem látható, változatlanul kell továbbítani.
Következtetett	Olyan érték, amely a fejléc más mezőiből kiszámítható.

Miután a linuxos gépet bekötöttük a hálózatba, foglalkozzunk a TCP/IP-veremmel, hogy elkészíthessük az illesztőeljárásokat, kétirányú, moduláris, folyamatos, valós idejű, kiszolgálóalapú stílusban. Ezek IP Chains-szabályokkal megállítják a csomagok áramlását és a libcap felület segítségével a felhasználóhoz vezetik őket, elemzés és tömörítés, illetve kicsomagolás céljából.
Ha a 2. ábrán látható rendszert felépítettük, akkor máris létrehozhatjuk a megszakítás nélküli telekonferencia-kapcsolatot, melynek csomagjai linuxos gépünkön haladnak át. Ennek működéséhez az IP-továbbítást is engedélyeznünk kell, legfőképpen azért, mert az IP Chains használatával szeretnénk megállítani a csomagok áramlását. Az IP-továbbítás engedélyezését a következő sor beírásával ellenőrizhetjük:

```
echo /proc/sys/net/ipv4/ip_forwarding
```

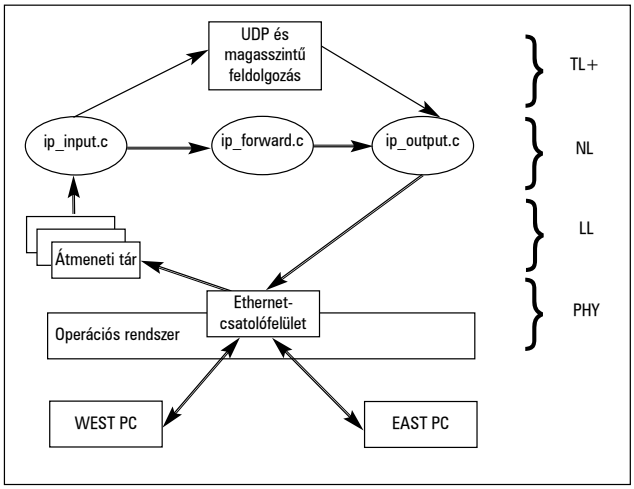
Eredményül egyet kapunk, ha a gép készen áll a csomagtovábbításra. A következő lépésben ellenőrizzük, hogy az adatáramlás leállítható és újraindítható-e anélkül, hogy lezárnánk a munkamenetet (session). Ez azt jelenti, hogy a TCP-csomagoknak megszakítás nélkül kell érkezniük. Most csak az IP-, UDP-, RTP-csomagokra kell figyelnünk. A következő parancs megállítja és újraindítja az adatfolyamot:

```
ipchains -P forward -DENY
ipchains -P forward -ACCEPT
```

A `-P` kapcsoló figyelmen kívül hagyja a protokollokat, tehát a parancs az ICMP-, az ARP-, a TCP- és az UDP-csomagokat egyaránt megállítja. Most már miénk az adatfolyam, vagyis meghatározhatjuk, hogy melyik csomagot és miként továbbítsuk.
A kapcsolati rétegben szeretnénk elcsípni a csomagokat, hogy megkapjuk az összes IP-mezőt. A csomagokat a hálózati verem kapja. Egy sk_buff nevű kapcsolt listaszerkezetben lesznek sorba állítva, ahol ezeket a rendszermag felsőbb programmegszakításai szolgálják ki önműködően az ip_input.c, ip_forward.c és ip_forward.c-ben.
A foglalat (socket) átmeneti tármemóriában történő kezeléséről *Alan Cox* „Network Buffers and Memory Management” (Hálózati átmeneti tárolók és a memóriakezelés) című cikkében olvashatunk (*Linux Journal*, 1996. október). A legtöbb felhasználói program a hálózati veremmel a Berkeley-féle csomagszűrőn (Berkeley Packet Filter – BPF) vagy INET foglalatokon keresztül tartja a kapcsolatot. Biztonsági okokból ezek a foglalatfelületek nem nyúlhatnak le az ethernet- vagy a fizikai rétegbe (PL). Megoldásként megnyithatunk egy nyers foglalatot, mellyel az IP-mezőket közvetlenül a verem IP-rétegéhez fordulva kaphatjuk meg. Noha a Libpcap támogatja a nyers csomagok olvasását, ezek továbbítása csak a Libpcap módosításával érhető el.
A TCP/IP verem eddigi legszínvonalasabb leírása *W. Richard Stevens* „UNIX Network Programming” (Unix hálózati programozás) című könyvének 1. kötetében található. A téma Linuxszal kapcsolatos

3. táblázat **Az IP-fejlécmezők és osztályozásuk**

IP-mező	Méret (bit)	Osztály
Változat	4	Statikus-ismert
Fejléchossz	4	Statikus-ismert
Szolgáltatástípus	8	Változó
Csomaghossz	16	Következtetett
Azonosító	16	Változó
Foglalt jelzőbit	1	Statikus-ismert
Töredék jelzőbit	1	Következtetett
Utolsó töredék jelzőbit	1	Statikus-ismert
Töredékeltolás	13	Statikus-ismert
Élettartam	8	Változó
Protokoll	8	Statikus-ismert
Fejléc-ellenőrzőösszeg	16	Következtetett
Forrás címe	32	Statikus-meghatározó
Cél címe	32	Statikus-meghatározó



3. ábra Linux TCP/IP-verem

részterületeiről a *David A. Rusling* által írt „The Linux Kernel” (A Linux-rendszermag) című munka 10. fejezetében olvashatunk, melyet a ➲ http://www.linuxhq.com/guides/TLK/net/net.html cím alatt találhatnak meg az érdeklődők.

A teljes, nyers csomagokat a hálózati kártya felé, illetve onnan kifelé is továbbítanunk kell, ehhez pedig a rendszermagban vagy a Libpcap-ban el kell végeznünk néhány módosítást. Ha azonban ezeket szeretnénk elkerülni, használjunk egy Perl5 CPAN csomagot, nevezetesen a RawIP-t (➲ http://quake.skif.net/RawIP/). A *3. ábra* egy diagramon ábrázolja a Linux TCP/IP-vermet és a tömörítendő csomagok kezeléséért felelős kódot a rendszermagban (2.2.x).

A Perl5 nyelven íródott első kódrészlet (➲ 17. CD/Magazin/Teletanacskozas/lista1) a megállított csomagok kiemelésének, az IP-azonosító mező megjelenítésének, a csomagok visszahelyezésének és végleges célja felé továbbításának alapvető eszköze. Csak a forrás- és a cél IP-címét szükséges értékként megadni. A héjprogram különbséget tesz a protokollok között, így külön-külön figyelhetünk a hang- vagy a mozgóképcsomagokra.

A parancsfájl egy id_dump.txt nevű szöveges állományt állít elő, amely

4. táblázat **Az UDP-fejlécmezők és osztályozásuk**

UDP mező	Méret (bit)	Osztály
Forráskapu	16	Statikus-meghatározó
Célkapu	16	Statikus-meghatározó
Hossz	16	Következtetett
Ellenőrzőösszeg	16	Változó

5. táblázat **Az RTP-fejlécmezők és osztályozásuk**

RTP-mező	Méret (bit)	Osztály
Változat	2	Statikus-ismert
Köz	1	Statikus
Kiterjesztés	1	Statikus
CSRC számláló	4	Változó
Jelző	1	Változó
Hasznos adat típusa	7	Változó
Sorozatszám	16	Változó
Időbélyegző	32	Változó
SSRC	32	Statikus-meghatározó
CSRC	0–480	Változó

megad egy – az adott kapcsolatban érvényes IP-azonosítóhoz tartozó – leírót. Ezt a többi mezőre is elvégezhetjük, amennyiben azt a parancsfájlban behelyettesítjük, ez az első lépés egy olyan gép beállítása felé, mely az IETF munkacsoportoknak benyújtott javaslatok bármelyikét megvalósítja. A 2. lista (➲ 17. CD/Magazin/Teletanacskozas/lista2) a Math::Random Perl5 CPAN csomagot használja az átlagosan húsz százalékos, egyenletes eloszlású adatcsomag-hibaarány, vagy – a VoIP esetében – kerethibaarány bevezetéséhez. Ennek hatása rögtön észrevehetővé válik az 1. kódrészlet átjárójának használatával, mely egyre inkább a magas szintű vezeték nélküli csatorna modelljére hasonlít. Ha valaki azt mondja, hogy az adatfolyam károsodására vonatkozóan megtalálta a középutat, annak annyit mondanék, hogy egy nagyon pontos, harmadik nemzedékbeli, széles sávú, osztott, többszörös hozzáférésű, többszörös Raleigh-féle elhalkulásokkal rendelkező modulált csatornamodell szabadon letölthető a ➲ http://w3.antd.nist.gov/wctg/3G/3G.html-ről. Ennek használata, bár nyomatékosan ajánlott, a Beowulf-telep hiánya esetén drámai következményekkel jár a kapcsolat valós idejű természetét illetően.

Továbbá ez a késleltetés csökkenti a rendszerekre egyénileg jellemző teljesítményt, melyet a mérnökök gyakran kiaknáznak a hangátviteli minőség meghatározásakor.

Most, hogy felszerelkeztünk egy mindenre alkalmazható rendszerrel, a következő lépés a mozgóképcsomagok vizsgálata és azok rugalmasságának tanulmányozása. A rendszer beágyazott környezetben is működhet tehát, mindenképpen olcsó teletanácskozás-eszközt biztosítana.

Izzet Agoren
villamosmérnök-hallgató a Pennsylvaniai Állami Egyetem Információ- és Kommunikációtechnológiai Kutatóközpontjában. Érdeklődési területe a harmadik nemzedékbeli mobil hálózati módszerek alkalmazása.

Az xdm beállítása

Az xdm kényelmes és rugalmas eszköz X-munkafolyamatok közvetett eléréséhez és kezeléséhez.

Előfordult már, hogy munkaállomásod felületét távolról kellett elérned? Mi történik akkor, ha ezt ráadásul egy kiszolgálón kell megtenned? A közelmúltban arra kényszerültem, hogy ehhez hasonló gondokat oldjak meg: épp egy halom Linux-kiszolgáló beállításaiért és működéséért voltam felelős. Különböző okok miatt már ötödször kellett a grafikus konzolomhoz elzarándokolnom, amikor ráébredtem: itt az ideje, választ kell találnom a különböző kiszolgálókon lévő X-felületek munkaállomásomról történő elérésének kérdésére. Néhányan talán úgy gondolják, hogy a hagyományos távoli munkafolyamatok kezelésére használatos X-eszközök elégségesek nehézségeim megszüntetéséhez, valahogy ilyen módon (esetleg egyéb munkafolyamat-kezelő is alkalmazható):

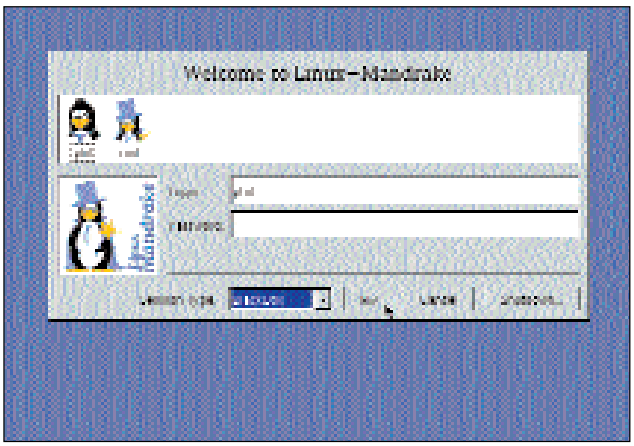
```
telnet host1
export DISPLAY=mywkstn:0
gnome-session
```

Én azonban olyan megoldást kerestem, amelyhez nem elegendő a csak alapszolgáltatásokat biztosító eszköz. Egy jobban kezelhető, minél inkább önműködő „szerszámra” volt szükségem, amelyet a Linuxszal csak most ismerkedő fejlesztők is könnyebben tudnának használni. A hitelesítéssel és a munkafolyamat-kezeléssel kapcsolatban azonban akad néhány olyan csapda, amelynek elkerüléséhez nélkülözhetetlen az X működésének bizonyos szintű ismerete, például a távoli X-ügyfelek használata során sokszor elfelejtem beírni az `xhost+ host1` parancsot. Gyakran előfordult az is, hogy zavart tekintetekkel találtam szemben magam, amikor egy-egy Linux-újoncnak az `xhost` hitelesítési rendszerét próbáltam elmagyarázni. Mivel a fejlesztőmunka mellett nemigen maradt időm arra, hogy a fejlesztőknek elmagyarázzam az X alapjait, olyan megoldást kerestem, ami a fenti kívánalmakat egyszerre elégíti ki.
Több olyan megoldás létezik, amely az X-munkafolyamatokat könnyebben kezelhetővé teszi. A feladat megoldásához én az *X display manager*-t (`xdm`) választottam, bár létezik egy másik népszerű megoldás is, a `vnc`. Két okból döntöttem így. Egyrészt a `vnc` kiszolgálóoldalról minden megosztandó gépen démon futtatását igényli. A másik tényező, hogy az X kiszolgáló már minden munkaállomásra fel volt telepítve, így a további telepítés szükségtelennek bizonyult. Szóba jöhetett volna még a `kdm` vagy a `gdm` is, amelyek a KDE-, illetve Gnome-csomagok részei.

Az X alapjai

Az X a unixos X-környezetekben elterjedt grafikus felületet támogató rendszer. Ha Gnome vagy KDE fut az adott Linux-rendszeren, a háttérben az X Window System dolgozik. Ennek készítésével és karbantartásával az X konzorcium foglalkozik ➲ http://www.X.org. A legtöbb Linux-felhasználó az Xfree86 Project (➲ http://www.xfree86.org) jóvoltából ennek valamely megvalósítását használja. Az `xdm` megjelenítésvezérlő, amely rugalmas vezérlőszolgáltatásokat nyújt a felhasználók számára. Bár sokan csak úgy gondolnak az `xdm`-re, mint „az a bejelentkező képernyő, amely önműködően indítja el X-felületemet”, meg fogjuk látni, hogy valójában ennél sokkal szélesebb körben használható eszközről van szó.

Az X világában az ügyfél és a kiszolgáló kifejezések kicsit összezavarhatják az embert. Az X kiszolgáló az az alkalmazás, amely a billentyűzetet, az egeret és a megjelenítést felügyeli. Az ügyfélalkalmazás kéréseket küld a kiszolgáló felé, hogy adott műveleteket hajtson végre (például jelenítsen meg bizonyos tulajdonságokkal



A kdm az xdm-hez hasonló grafikus bejelentkező felület

leírható ablakot). Ez kicsit furcsa lehet azok számára, akik a munkaállomásukon futó alkalmazásokra úgy gondolnak, mint ügyfelekre. Az X kiszolgálókkal történő kapcsolattartásra az `xdm` az X konzorcium XDMCP protokollját használja. Ez lehetővé teszi az X kiszolgálóknak, hogy munkafolyamat-szolgáltatásokhoz jussanak az `xdm`-et futtató kiszolgálóktól. Az X kiszolgálók háromféle kérést küldhetnek:

- Közvetlen – kérés közvetlenül a megnevezett hálózati géphez bejelentkező képernyő megjelenítésére.
- Üzenetszórás – üzenet küldése az összes hálózaton lévő számítógép felé, a beléptetési szolgáltatást az elsőként válaszoló nyújtja.
- Közvetett – kapcsolatba lépés egy megnevezett, `xdm`-et futtató géphez, innen lista kérése azokról a számítógépekről, amelyekkel kapcsolat létesíthető. Az `xdm` kiszolgáló listát készít azokról az elérhető kiszolgálókról, amelyek hajlandók az X-munkafolyamat kezelésére. Végül az X kiszolgáló befejezi a közvetlen kapcsolatot a kiválasztott géppel, hogy a beléptető szolgáltatást megkaphassa.

Az `xdm` létrejöttének okai közül a talán legkorábbi az X-terminálok kezelésének igénye volt. Ezek az eszközök csak egy kijelzőből, billentyűzetből, egérből, valamint egy beépített X-kiszolgálóalkalmazásból álltak, minden tevékenység a hálózaton lévő kiszolgálógépen összpontosult. Az `xdm` feladata ezen eszközök – a bejelentkező képernyő és az X-folyamatok – vezérlésének a biztosítása. Néhány évvel ezelőtt e terminálok még igen népszerűek voltak a Unix-munkaállomásokhoz való hozzáférés erősen korlátozott volta miatt, főleg azon felhasználók körében, akik íróasztaluk mellett szerettek volna grafikus felületen dolgozni – nekik vagy elég szerencsésnek kellett lenniük, hogy saját munkaállomásuk legyen, vagy kénytelenek voltak termi-

xdm-config

```
DisplayManager.errorLogFile:/var/log/xdm-errors
DisplayManager.pidFile:      /var/run/xdm-pid
DisplayManager.keyFile:      /etc/X11/xdm/xdm-keys
DisplayManager.servers:      /etc/X11/xdm/Xservers
DisplayManager.accessFile:   /etc/X11/xdm/Xaccess
DisplayManager.willing:      su nobody -c/etc/
                             ➥X11/xdm/Xwilling
DisplayManager*resources:    /etc/X11/xdm/Xresources
DisplayManager*session:      /etc/X11/xdm/Xsession
DisplayManager*authComplain:     false
! Tegye megjegyzØsbe ezt a sort,
! ha az X-terminÆlokat xdm-mel akarja vezØrelni
! BIZTONS`G: Ez azt jelenti, hogy a
! 177-es kapu figyeli az XDMCP CHOOSER kØrØseket.
#DisplayManager.requestPort:     0
```

nált használni. Később ezek az eszközök veszítettek a népszerűségükből: X-kiszolgálóalkalmazással telepített PC-kkel kezdték helyettesíteni őket, amelyeken a felhasználók Linuxot vagy egyéb Unixot (Solaris x86, xBSD stb.), illetve Windowst (Hummingbird Exceed és más hasonlót) futtattak.
Amikor az X-folyamatok vezérlésére az `xdm`-et használjuk, nem nélkülözhetjük a beállításokkal kapcsolatos alapfogások ismeretét. Első pillantásra úgy tűnik, hogy amikor az XDMCP előnyeinek kiaknázásához az `xdm`-et beállítjuk, vagy elindul egy helyi X kiszolgáló (például a konzol az `xdm` indulásakor grafikus módra vált), vagy amennyiben az `xdm` helyi kijelzését letiltjuk és a `startx` parancsot használjuk, nem tudjuk a gépválasztót elérni. A most ismertetésre kerülő beállítás bármely XDMCP-ügyfél számára lehetővé teszi a Linux-kiszolgáló munkaasztalának elérését (természetesen az X biztonsági beállításaitól függően). Emellett annak módját is bemutatja, hogyan lehet a fentivel egyidejűleg helyi X-munkafelületünk, és miként érjük el más kiszolgálók felületeit a munkaállomásról. Az `xdm` biztonsági és hozzáférési szabályozást is lehetővé tesz, de ismertetése túlmutatna e cikk keretein. A legcélszerűbb, ha az `xdm`-et csak biztonságilag felügyelt környezetben használjuk.
A 177-es bejövő kaput minden tűzfalon le kell tiltani. Az X felület biztonsági kérdéseivel kapcsolatban a következő súgóoldalak szolgálnak jó kiindulópontként:
`xdm(1)`, `xauth(1)`, `Xsecurity(7)`, `lbxproxy(1)`-Low Bandwidth X proxy, `xfwp(1)`-X Firewall Proxy, `ssh(1)` és `sshd(8)` súgóoldalak, különös tekintettel az X11 kapu átirányítására.

Az xdm beállítása

Az `xdm` gazdag beállítási lehetőségekkel rendelkezik, az alábbi csak egy példa, amellyel célkitűzésünk megvalósítható. Az én RedHat 7-es rendszeremen az `xdm` a `/etc/X11/xdm` útvonalon található. A fő beállítófájl az xdm-config (lásd *listánkat*). Lehetőségünk nyílik a különböző beállítóállományok elérési útvonalának meghatározására. Ezek közül mi azokkal foglalkozunk, amelyekre a `servers`, az `acessFile` és a `resources` beállítás mutat. A vállalkozó kedvűeket még a `session` és a `DisplayManager._X.setup` is érdekelheti, ahol a kijelző száma X.
A `DisplayManager.requestPort:0` sor fel van függesztve. Ez az erőforrás határozza meg, hogy melyik UDP-kapu figyel az XDMCP-kérésekre. Ha ez 0-ra (az alapértelmezett értékre) van állítva, az `xdm` az XDMCP-kéréseket figyelmen kívül hagyva csak a helyi megjelenítést vezérli (lásd még az Xservers állományt). Ezt azért tettük, hogy az `xdm` az alapértelmezett kaput használja (177-es UDP-kapu).
Az én Xservers-állományomban a kérdéses sor így néz ki:

```
#:0 local /usr/X11R6/bin/X
```

Ha ez a sor élne, az `xdm` minden egyes indításánál grafikus beléptetőkép jelenne meg, vagyis a `/usr/X11R6/bin/X` parancs futtatásával a 0-s kijelzőn egy helyi X kiszolgálót indítana el és vezérelne. Nekünk most az a célunk, hogy azt a gépet választhassuk ki, amelyikre kapcsolódni szeretnénk. Ezt az *Xaccess* állománnyal érhetjük el:

```
#any indirect host can get a chooser
*  CHOOSER BROADCAST
#
# Ha inkább közvetlenül adja meg azoknak a
# gépeknek a körét, amelyeket minden terminál
# elér, függessze fel ezeket a sorokat
# (és a fenti CHOOSER sort),
# és a %hostlist sort állítsa be megfelelően
#
#%hostlist    host-a host-b
#*     CHOOSER %hostlist        #
```

Bár ez a parancsfájl nagyon rugalmas eszköz, mi mégis csak a gépválasztó elindítására fogjuk használni (közvetett mód). A gépválasztó olyan kisméretű X-alkalmazás, amely a hálózaton elérhető számítógépeket sorolja fel, és lehetővé teszi, hogy közülük kiválasszuk azt, amelyikhez csatlakozni szeretnénk. Az általam kedvelt `BROADCAST` érték hatására az új gépek önműködően megjelennek a listán. Akadhatnak olyanok, akik inkább a `%hostlist` makró segítségével nevezik meg a kívánt számítógépet. Erre a módszerre olyankor is szükség lehet, amikor egy nagyobb hálózaton az üzenetszórás nem éri el az összes kívánt gépet.
Ha az elérést még ennél is finomabban akarjuk szabályozni, a `BROADCAST` helyett választhatjuk a kiszolgálók felsorolását, ekkor közvetlenül meg kell adnunk az elérhető kiszolgálók listáját.
Ha az `xdm`-et arra is fel akarjuk készíteni, hogy különböző X-kiszolgálóktól eltérő módon legyen képes kéréseket fogadni, meg kell adnunk a kérdéses gép nevét vagy a gépek listáját. Ezt szemléltetjük következő példánkkal: az alábbi sorok hatására az `xdm` a `host-a`, `host-b` vagy `host-c` gépről érkező kérelmeket egyaránt kezelni fogja:

```
host-a
host-b
host-c
```

Ha az `xdm`-et arra akarjuk utasítani, hogy közvetett kérést küldjön a `host-a` gépről a `server-a` vagy `server-b` nevű gépekre, az alábbi sort gépeljük be:

```
host-a        server-a server-b
```

Ezt megtehetjük a

```
%hostlist     server-a server-b
```

vagy a

```
host-a        %hostlist
```

sorokkal is.

Az xdm-et úgy is beállíthatjuk (ez a kedvenc módszerem), hogy a gépválasztó használatával kezeljen közvetett kéréseket. Következő példámban a host-a gép egy gépválasztó ablakot kap az összes olyan gép listájával, amelyik válaszolt a BROADCAST üzenetre, míg az összes többi gép csak a %hostlist által meghatározott gépeket kapja meg:

```
%hostlist     server-a server-b
host-a        CHOOSER BROADCAST
CHOOSER %hostlist
```

Az alapvető működés bemutatásának végére érve vessünk egy pillantást az *Xresources* állományra. A sajátomat változatlanul hagytam, de előfordulhat, hogy valaki a külalakot is testre szeretné szabni. Ebben a fájlban a színt, a betűtípust és az egyéb külső jellemzőket változtathatjuk meg. Leghasznosabbnak a Chooser*geometry lehetőséget találtam, melynek segítségével a gépválasztó ablakának mérete állítható be.
Módunkban áll felügyeleti beállításokat is végezni, például a DisplayManager.errorLogFile logfile határozza meg a napló állományt. Ez a naplóállomány a stderr output of xdm, az Xsetup, az Xstartup, az Xsession és az Xreset szövegeit tartalmazza.
A sikeres bejelentkezési folyamatot követően az xdm a munkamenet (session) beállításában meghatározott parancsállományt futtatja le. A felhasználók számára ez teszi lehetővé, hogy az X-folyamatokat testreszabják. A rendszergazdákat leginkább az *Xsession* parancsállomány fogja érdekelni, a felhasználók a folyamatvezérlő viselkedését pedig saját *$HOME/.xsession* vagy *$HOME/.Xclients* állományukkal akarják majd egyéni igényeikhez szabni (például ablakkezelő vagy óra elindítása stb.).

Beállításaink ellenőrzése

Beállításaink ellenőrzéséhez először saját X-állományunkat kell megtalálnunk, saját rendszeremen ennek helye: /usr/X11R6/bin/X. Minden esetben egy beléptető képernyő előtt kell találni magunkat. Az ellenőrzés közvetlen módjára a

```
/usr/X11R6/bin/X -query remotexdmhost
```

parancs begépelésével nyílik lehetőség.
Közvetett üzenetszórásos mód esetén az alábbi formát kövessük:

```
/usr/X11R6/bin/X -broadcast
```

A közvetett mód a gépválasztóval pedig a

```
/usr/X11R6/bin/X -indirect remotexdmhost
```

parancs hatására indul.
Amikor ezek működtek, az xdm-szolgáltatás önműködő be- és kikapcsolására elkészítettem a */etc/rc.d/init.d* parancsállományt.

Kapcsolódó címek

X Window System ➲ http://www.X.org

X Window System-címek
➲ http://www.rahul.net/kenton/xsites.framed.html

XDM and X Terminal Mini-GY
➲ http://www.linuxdoc.org/HOWTO/mini/XDM-Xterm/index.html

Ezt követően felhasználóim életének megkönnyítésére munkaállomásaikon egy */usr/bin/X11/startx.xdmcp* nevű állományt hoztam létre. Amennyiben a név *"wkstn1"*-ként fest, az állomány tartalma a következő lesz:

```
#!/bin/sh
/usr/X11R6/bin/X -indirect wkstn1
```

amelyben a *wkstn1* az xdm-kiszolgáló neve (esetemben a munkaállomások az xdm kiszolgáló és az X kiszolgáló szerepét is betöltötték). Ezután az alábbi sorok következnek:

```
mv /usr/bin/X11/startx
➥/usr/bin/X11/startx.original
chmod 755 /usr/bin/X11/startx.xdmcp
ln -s /usr/bin/X11/startx.xdmcp
➥/usr/bin/X11/startx
```

Ez minden felhasználó számára lehetővé teszi, hogy a szokásos bejelentkezés és startx parancs helyett az elérhető gépekről listát kapjon (amin a sajátja is szerepel).

Összegzés

Az eddigiek összegzéséül elmondhatjuk: sikerült beállítanunk az xdm használatát a hálózat munkaállomásain és kiszolgálóin az XDMCP mind közvetlen módú, mind gépválasztóval történő működésével. Ezzel a felhasználók számára lehetővé vált, hogy azt a kiszolgálót választhassák ki, amely X-munkafolyamataikat összehangolja. Ez amellett, hogy az X-folyamatokat finoman szabályozhatóvá teszi, az említett döntési lehetőséget a felhasználók számára egy egyszerű menüben biztosítja. Amennyiben más felület (például Windows) alatt lenne ilyen elérési módra szükség, és el akarjuk kerülni az X kiszolgáló megvásárlását vagy telepítését, további lehetőségként a vnc alkalmazása kínálkozik.

Ron Hume
(ronhume@ieee.org) független híradástechnikai szaktanácsadó. Szakterületei: softswitch-technológia és továbbfejlesztett szolgáltatásszerkezet a hagyományos és a feltörekvő telekommunikációs cégek számára.

Technikai tipp

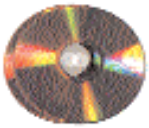
par

Nemcsak jó lesz, de a **par**-ral jól is fog kinézni. Egy levelezőlistával kapcsolatos kérdésre adott válasz megformázása meglehetősen bonyolult lehet, mivel az idézett szövegsorok elé néhányan > jelet raknak, mások az eredeti bevezető karaktert, továbbá a sorszélesség is erősen változó. Rendezett válasz elkészítéséhez, ahol az idézetek megfelelően jelennek meg, használd *Adam M. Costello* **par** segédprogramját. A **vim**-mel szúrd be az alábbit a .VIMCR állományba:

```
map * j{!}par^M}
```

ami a * hatására formázza az adott bekezdést (a ^M valódi CTRL-M; vim alatt üss CTRL-V CTRL-M-et). A **par** beállításaival az idézett szöveg pontosan úgy fog kinézni, ahogyan szeretnéd.
Elérhető a ➲ http://www.cs.berkeley.edu/~amc/Par/ címen.

Postfix-csemegék (3. rész)



A Postfix alapvető beállításai után kóstoljunk bele az ínyencségekbe is.

Tekintsük úgy, hogy a tévében jó kis sorozatok epizódjait nézzük – ugyanis mire minden lehetőséget végigpróbálunk, csaknem annyi idő telik el, mintha egy nagyot tévéztünk volna.

Első epizódunk: a webkiszolgálók

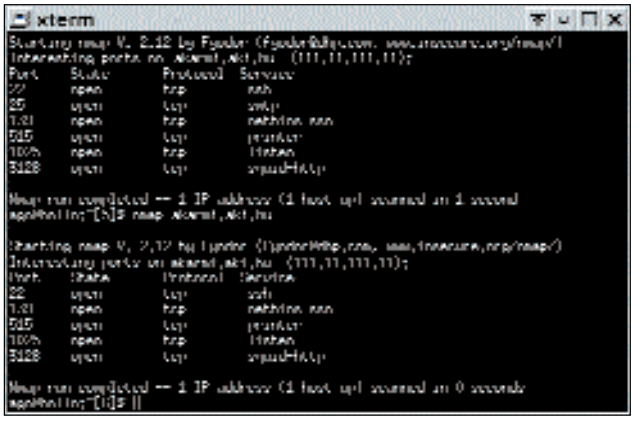
Tételezzük fel, hogy adott egy webkiszolgáló. Ha egy kiszolgálón csak webkiszolgálónak kellene futnia, semmi keresnivalója nem lenne ott egy levelezőkiszolgálónak. De vajon miként kerülnek ki a levelek? Az olyan nem moduláris felépítésű kiszolgálókkal, mint a Sendmail vagy az Exim, biztosan érdekes feladat ezt megoldani, a Postfix azonban képes arra, hogy a különböző levelezési szolgáltatások egymás nélkül is kitűnően tudjanak működni. A mi célunk az, hogy csak a webkiszolgáló vagy a naplófigyelő programok – például a Swatch (lásd még Linuxvilág, 2001. augusztus, 46. oldal) – tudjanak jelentést küldeni, azaz csak tőlük fogadjunk leveleket. Ehhez szükséges lépésként rá kell beszélni a Postfixet, hogy a 25-ös kapun (a kiszolgálók ezen a kapun keresztül kapják a leveleket) csak a 127.0.0.1-es címről fogadjon el kapcsolatot. Ehhez a /etc/postfix/main.cf fájlban keressük meg az `inet_interfaces` kezdetű sort, majd változtassuk meg az értékét (ez alapbeállításban `all`) `localhost` vagy `127.0.0.1` értékre:

```
inet_interfacels = 127.0.0.1
```

Milyen előnnyel bír ez más megoldásokhoz képest? Ha a kiszolgáló levélfogadását csomagszűrő tűzfallal akadályozzuk meg, az pásztázó programmal könnyen felderíthető. Látszik, hogy a kapu nyitott és a kapcsolat szűrt. A mi célunk azonban az, hogy tudomást se szerezzenek róla. Ha választható megoldásként a kiszolgálót más kapun (porton) indítjuk, akkor azt az összes kaput átvizsgáló pásztázás ugyancsak könnyen felderítheti. Nézzük meg azonban a különbséget az *1. képen*, ha a Postfix szolgáltatását használjuk: a felderített kapuk között a levélfogadás nem jelenik meg.

Második epizódunk: SMTP démon nélkül

Az előző részben leírt viselkedésmódot tovább fejleszthetjük. Egyáltalán nem fogadunk levelet az `smtpd`-n (a Postfix 25-ös kapuján figyelő démonon) keresztül, csak a helyi, fájlba írt leveleket továbbítjuk. A leveleket írhatja ember, de program is. Miért lehet erre szükség? Tegyük fel, hogy teljesen kényszeresek vagyunk (bár egy barátom szerint olyan nem létezik, hogy valaki túlságosan paranoid). Félünk attól, hogy valaki mégis megtudja, hogy a gépen levelezőkiszolgáló van, és a csomagszűrő tűzfalat sem mi állítottuk be, hanem egy nem túl hozzáértő „szaki". Számítógépünk hálózati kapcsolón (switch) vagy jelelosztón (hub) osztozik egy vagy több olyan számítógéppel, amelyet feltörtek, mondjuk a legutóbbi `bind` hiba miatt. Mivel a gép rendszergazdája a csomagszűrést rosszul állította be, a hálózati kártya felől bejövő kapcsolatokat is elfogadja a 127.0.0.1-es forráscímről (ez azonban valós körülmények között lehetetlen). Az egyetlen lehetőség ekkor az, ha kikapcsoljuk az `smtpd`-t. Könnyen megtehetjük, csak ismernünk kell hozzá a Postfix szerkezetét. Az összes levelezéssel kapcsolatos démont a `master` program felügyeli. Ebből pedig az következik, hogy amiről nem tud, az nem is működik. A megoldás így roppant egyszerű: a /etc/postfix/master.cf fájlban keressük meg az `smtpd`-re vonatkozó sort, és érvénytelenítsük vagy egyszerűen töröljük ki.
Egy módosított master.cf fájl tartalma látható az *1. listán*.



1. lista Postfix **`smtpd`** démon nélkül

```
#smtp      inet  n   -   -   -   -   smtpd
pickup     fifo  n   n   -   60  1   pickup
cleanup    unix  -   -   -   -   0   cleanup
qmgr       fifo  n   -   -   300 1   qmgr
rewrite    unix  -   -   -   -   -   trivial-rewrite
bounce     unix  -   -   -   -   0   bounce
defer      unix  -   -   -   -   0   bounce
smtp       unix  -   -   -   -   -   smtp
showq      unix  n   -   -   -   -   showq
error      unix  -   -   -   -   -   error
local      unix  -   n   n   -   -   local
```

A vastagon szedett rész mutatja, hogyan kell az adott sort megjegyzéssé tenni.

Harmadik epizódunk: a „nullügyfél"

Felépítése hasonló az előzőéhez, azzal a különbséggel, hogy míg ott lehetőség nyílik a helyi levéltovábbításra, addig itt erre nincsen mód, egyedül az SMTP protokollon keresztüli küldés engedélyezett. Összefoglalva: a nullügyfél olyan számítógép, amely a leveleket csak küldeni tudja. Alkalmazása azokon a kiszolgálókon célravezető, amelyeken a levelezés kimerül a levélnek egy másik kiszolgálóra történő elküldésében. Az első és jelen beállítás közti eltérés, hogy az előbbinél a rendszergazda helyileg is kaphat levelet, például a webkiszolgálótól, az utóbbinál viszont nem. A nullügyfélhez szükséges, hogy Postfixünk tudja, melyik másik kiszolgálót használhatja a levélküldéshez. Ezenkívül minden olyan levelet, amely a gépről „származik", úgy kell elküldenünk, hogy ne látszódjon a gép teljes neve (Full Qualified Domain Name). Az utolsó szükséges beállítás a helyi levéltovábbítás tiltása az `smtpd` mellett. A master.cf és a main.cf beállításait a *2. listán* láthatjuk.

Negyedik epizódunk: a kapcsoltvonali előfizető

Gyakori eset, hogy csak telefonon keresztüli internetkapcsolattal rendelkezünk. Ekkor két út közül választhatunk: az első, hogy olyan

2. lista A Postfix beállítása a nullügyfélhez

```
/etc/postfix/main.cf

# az elküldött leveleknél mindig a tartománynév
# fog látszani a @ jel után
# myorigin = $mydomain a tartomány levelezésért
# felelős számítógépét keressük meg, és rajta
# keresztül küldjük el a levelet
relayhost = $ mydomain

/etc/postfix/master.cf
#smtp     inet  n       -       -       -       -       smtpd
pickup    fifo  n       n       -       60      1       pickup
cleanup   unix  -       -       -       -       0       cleanup
qmgr      fifo  n       -       -       300     1       qmgr
rewrite   unix  -       -       -       -       -       trivial-rewrite
bounce    unix  -       -       -       -       0       bounce
defer     unix  -       -       -       -       0       bounce
smtp      unix  -       -       -       -       -       smtp
showq     unix  n       -       -       -       -       showq
error     unix  -       -       -       -       -       error
#local    unix  -       n       n       -       -       local
```

A vastagon szedett rész mutatja, hogyan kell az adott sort megjegyzéssé tenni.

levelezőprogramot használunk, mely addig gyűjti a leveleket, amíg azt nem mondjuk neki, hogy elküldheti őket. Ennek a megoldásnak előnye, hogy Postfixünket nem kell beállítani – de akkor mire jó? A kérdés megválaszolásához ki kell emelnünk a program hátrányát. Mi történik akkor, ha elfelejtjük elküldeni a leveleket, és akad köztük nagyon fontos is? Nem biztos, hogy újra tudunk csatlakozni, ha „csak” egyetlen ingyenes internetkapcsolattal (például Freestart, Kiwi) bírunk. Ha azonban helyi kiszolgálót használunk, nincs más dolgunk, csak a rendszer tudtára adni, hogy minden kapcsolatfelépítéskor küldje el a leveleket. A kapcsoltvonali beállításhoz az alábbiakat szükséges tudnunk: ki továbbítja majd a levelet; valamint hogy minden levél, amelyet az Internetre továbbítanánk, bent maradjon-e a sorban (queue). Fontos, hogy a kapcsolatot addig ne bontsuk le, amíg minden levelet legalább egyszer meg nem próbáltunk elküldeni. Ezt egy parancsfájlból fogjuk ellenőrizni. A szükséges Postfix-beállítások magyarázattal a következők:

```
/etc/postfix/main.cf
#az alábbi gépen keresztül küldjük el a
#leveleket, ez lehet például a szolgáltató
# smtp-átjárója
relayhost = smtp.szolgaltato.hu
#visszatartjuk az smtp-n keresztül küldendő
#leveleket, amíg nincs kapcsolat
defer_transports = smtp
```

A kimenő sor önműködő kiürítésére parancsfájlt hívunk meg abból a fájlból, amely a vonalas (ppp) kapcsolat felépülése után hajtódik végre. A Debian-rendszer alatt a végrehajtódó parancsállományokat a /etc/ppp/ip-up.d könyvtárban külön fájlokban helyezzük el.
A parancsfájl neve legyen connect. A connect fájl tartalma a következőképpen nézzen ki (a Postfix GyK alapján, de azt módosítva):

```
#!/bin/sh
#kiürítjük a sort
postfix flush

# várunk, hogy minden levél közvetítése
# elkezdődjön, azaz egy smtp-kapcsolat
# elinduljon a távoli kiszolgálók felé,
# várunk 30 másodpercet
sleep 30

# Egészen addig fenntartjuk a kapcsolatot,
# ameddig minden levelet legalább egyszer
# megpróbáltunk elküldeni, ennek keretében
# nem engedélyezzük a poff parancs
# küldését, mentjük más néven, majd
# később visszamásoljuk
mv /usr/bin/poff /usr/bin/poff.off

# amíg új levél tartózkodik a sorban,
# addig várunk, mindig 10 másodperccel
# növeljük a kapcsolat idejét
while mailq | grep '^[^ ]*\*' >/dev/null
       do
        sleep 10
       done
# ha már nincs olyan levél, ami ne
# próbált volna kimenni, akkor
# visszamásoljuk a poff parancsot és
# lezárjuk a kapcsolatot.
mv /usr/bin/poff.off /usr/bin/poff
➥/usr/bin/poff
```

Egyszer nekem szegezték a kérdést, hogy mi történik az alábbi két esetben:

1. Egy rosszindulatú felhasználó olyan parancsfájlt ír, amely a hálózati kapcsolat meglétét ellenőrzi (például egy `ping` parancs kimenetét vizsgálva), majd ha az él, a leveleket iszonyatos mennyiségben kezdi küldeni, mindehhez egy Perl-modult használva.
2. Hálózatba vagyunk kötve, számítógépünk a hálózati és levelező átjáró (smart host). Valaki rossz szándéktól vezérelve kapcsolat idején levelekkel kezd minket bombázni.

Úgy tűnik, ilyenkor sosem tudjuk lekapcsolni a vonalat, és a nappali telefonálás nem olcsó mulatság. Megnyugtatok mindenkit, ez nem fordulhat elő. Miért? Emlékezzünk vissza, hogy a Postfix beállítófájljába beillesztettünk egy olyan sort, amely minden SMTP-kapcsolatot addig késleltet, amíg külön parancsot nem adunk a sor kiürítésére. Ilyen parancsot pedig csak rendszergazdai jogosultságú felhasználó vagy az ő jogaival futó parancsfájl képes végrehajtani. (Ha rajtunk kívül más is rendelkezik rendszergazdai jogosultsággal, akkor már mindegy.) Ellenérv lehet, hogy ha hálózaton nem is, de az első módszert alkalmazva helyileg küldhetünk levelet, hiszen nem kell SMTP-kapcsolatot használni ahhoz, hogy a Postfix elfogadja a levelet. Ez igaz is, használhatjuk a maildrop könyvtárat, de ha nem helyi felhasználónak küldjük a levelet, akkor SMTP-kapcsolaton keresztül kell távoznia, ez pedig késleltetett (deferred) üzemmódban működik. A Postfix modularitása – mint számos más esetben – itt is kisegít bennünket.

Ötödik epizódunk: Postfix és MySQL

Előfordulhat, hogy nem szívesen veszünk fel felhasználókat a rendszerbe, még úgy sem, hogy nem létező héjprogramjuk (shell) van, például a /bin/false vagy a /dev/null. Az is megeshet, hogy már túl sok – például 10 000 – felhasználónk van, ezért a kiszolgáló működése nem hatékony, rendszere esetleg már nem is engedélyezi

újabb felhasználók felvitelét.
Ez esetben eszes megoldásra van szükségünk. Ilyen lehet egy SQL- vagy LDAP-kiszolgáló rendszerbe olvasztása. Ekkor nemcsak a felhasználók meglétének ellenőrzését végezhetjük el, de lehetőséget biztosíthatunk a vezetőség számára, hogy letiltsa a nem fizető ügyfelek azonosítóit, vagy külön kvótákat alkalmazhatunk a különböző előfizetői kategóriák felhasználói számára. A lehetőségek száma szinte a végtelenhez közelít: nem lesz olyan feladat a levelezésben, ahol ne tudnánk hasznosítani a központi adatbázist (többszörözés, adatok ki- és visszamentése stb). A Debian következő, Woody nevű terjesztésében a támogatás már a Postfix-csomagba lesz építve. Addig azonban saját magunknak kell fordítanunk a csomagot, ha azt akarjuk, hogy MySQL-támogatás is legyen. Ehhez az include és header fájlok miatt szükségünk lesz a MySQL fejlesztői csomagjára. Ha a MySQL-t is forrásból fordítjuk, ezek is felkerülnek a telepítés után. A Postfix fordításával már foglalkoztunk a legelső részben. Mivel még mindig nincs configure parancsállomány, a make programnak kell átadni változóként, de csak akkor, amennyiben a programba a MySQL táblaalapú adattárolását is be akarjuk építeni. Ha nem ez az első fordítás, mindenekelőtt takarítsunk egy kicsit:

```
# make tidy
```

Ezt követően adjuk ki a következő parancsot:

```
# make -f Makefile.init makefiles
➥'CCARGS=-DHAS_MYSQL -I/usr/local/include/mysql'
➥'AUXLIBS=/usr/local/lib/mysql/libmysqlclient.a -lm'
```

Ha a Makefile elkészült a fenti beállításokkal, kiadhatjuk a parancsot:

```
# make
```

A szokásos folyamat zajlik le, mindegyik démon külön-külön lefordítódik, és elindíthatjuk a telepítő és beállító parancsfájlt:

```
# sh INSTALL.sh
```

Ha elindítjuk a Postfixet, a postconf programot használva láthatjuk, hogy a használható adattároló-típusok között megjelent a MySQL-támogatás:

```
# postconf -m
nis
regexp
environ
mysql
btree
unix
hash
```

Amennyiben újabb kiadású MySQL-kiszolgálót használunk és csomagból telepítettünk, rendelkeznünk kell a zlib fejlesztői csomagjával (zlib-dev), ugyanis most már a libmysqlclient.a tárgykönyvtárat a zlib segítségével készítik. Ha viszont a MySQL-t forrásból telepítettük, a fordítást megelőző beállítóprogram alapértelmezettként a zlib-csomagokat használja. Ekkor az SQL-támogatás beépítéséhez a Makefile-t a következőképpen kell indítanunk:

```
# make -f Makefile.init makefiles
➥'CCARGS=-DHAS_MYSQL -I/usr/local/include/mysql'
➥'AUXLIBS=/usr/local/lib/mysql/
➥libmysqlclient.a -lm -lz'
```

Látszik tehát, hogy az egész mindössze egy `-lz` lehetőséggel bővült. Csak három karakter, de ha nem tudjuk, hogy oda kell rakni, sokat szenvedhetünk, mire rájövünk a hiba okára. A jelenlegi Postfix-terjesztésből hiányzik e tény megemlítése, mégis érdemes megnézni a forráscsomagban lévő MYSQL_README fájlt, mert ez a hivatalos leírás, és amennyiben később változtatnak valamin, ott biztosan megtaláljuk ezeket. A csomagban szintén találhatunk egy leírást arról, hogy milyen táblákat kell létrehozni az adatbázis-kiszolgálóban, és milyen formátumban kell kitölteni egy SQL-t használó mapfájlt. Ha ezt követően ilyen fájlra hivatkozunk, a típusnál a `mysql`-t kell megadni, ilyen formában:

```
alias_maps = mysql:/etc/postfix/mysql-alias.cf
```

Ha ezt a típust szeretnénk a hash helyett alapértelmezetté tenni, akkor a

```
default_database_type = hash
```

sort a következőre kell cserélni:

```
default_database_type = mysql
```

Éjféli híradó: helyi levéltovábbítás

A Postfix helyi levéltovábbításra saját helyi programját használja, de bármikor eltérhetünk tőle, erre ad lehetőséget a mailbox_command érték. Amíg ennek értéke üres, addig a helyi démon fog végrehajtódni, eltérni csak értékadással lehet. Ha a procmailt akarjuk levéltovábbításra használni, az értéket a main.cf-ben állítsuk be rá:

```
mailbox_command = /usr/bin/procmail
```

Figyelem! Postfix használata esetén a `procmail`-nek set-uid bittel kell rendelkeznie! Ez teszi lehetővé, hogy mindig a címzett felhasználói jogaival tudjon futni és beleírjon a felhasználó postafiókjába. Vigyázat! Ha nem `mbox`, hanem `Maldir/` formátumban (minden levél külön fájlban) tároljuk a leveleinket, a `procmail` nem alapbeállításaival fog működni. `Maildir`-be mentéshez használjuk inkább a helyi démont, ami el is készíti a szükséges könyvtárakat, amennyiben még nem léteznek.

A fogmosás: ha elsőre nem sikerül…

Mi történik akkor, ha a rendszeren kétféle felhasználó létezik? Egy, aki benne van a /etc/passwd fájlban, és egy, aki viszont nem, mert mondjuk a Cyrus-IMAP saját virtuális felhasználói között található. Ekkor nem a `mailbox_transport`, hanem a `fallback_transport` értéket kell beállítani. Ha a master.cf fájlban létezik például a `cyrus` bejegyzés (amennyiben nem töröltük ki, alapértelmezésben megtalálható benne), megtehetjük, hogy először megpróbáljuk a /etc/passwd-ben megtalálni a felhasználót, majd ha ott nincs, akkor a Cyrusnak átdobni a levelet. Az ehhez szükséges beállítások a /etc/postfix/main.cf-ben találhatók:

```
mailbox_command =
fallback_transport = cyrus
```

Remélem, mindenkinek sikerült megtalálnia az őt legjobban érdeklő témát. A következő alkalommal néhány nem szokványos beállítást, valamint egy irodai rendszer beüzemelését mutatjuk be.

Deim Ágoston (ago@lsc.hu)
Kedveli a sört, szereti a futást és imádja Szabó Lőrinc verseit. Nem hisz vakon egyik rendszerben sem. Vonzódik a BSD-hez is. Tagja az LME-nek és a MBE-nek. Mottója: a gép nem lehet fontosabb az embernél.

POP3- és IMAP-kiszolgálók

Írásunk a két protokoll közti különbségeket boncolgatja és segít a megfelelő alkalmazás kiválasztásában.

Sok ember csak annyit tud, hogy mindkét protokollt levelek elérésére használjuk, továbbá egyformán léteznek hozzájuk ügyfél- és kiszolgálóprogramok. A különbség valójában meglehetősen nagy. A POP3- és az IMAP-szolgáltatás közti éles különbség abban rejlik, hogy míg a POP3-kiszolgálókról alapértelmezésben letöltjük, majd töröljük a leveleket, addig az IMAP-kiszolgáló alapbeállítás szerint a leveleket rajta hagyja a távoli számítógépen. A másik nagy eltérés, hogy a POP3-kiszolgálón egyetlen fiókban található az összes levél, az IMAP-protokoll viszont lehetőséget biztosít a felhasználó leveleinek külön könyvtárban történő tárolására. Mivel a POP3-kiszolgálóról a leveleket letöltjük, ezért azok a továbbiakban saját számítógépünkön tárolódnak, egyúttal megfosztjuk magunkat attól, hogy azokat több számítógépről is elérhessük. Az IMAP megoldást kínál erre, a szolgáltatók azonban szokásosan nem kínálnak IMAP-elérést, csakis a POP3-at „választhatjuk". Hogy mi ennek az oka, ha egyszer az IMAP ennyire hasznos protokoll? A válasz: az erőforrásigénye. A POP3-kiszolgáló igényei össze sem vethetők egy IMAP kiszolgálóéval. A POP3-kiszolgáló feladata kimerül abban, hogy elérést biztosítson ahhoz a könyvtárhoz vagy fájlhoz, amelyben leveleink letöltésre várakoznak, az IMAP-rendszernek viszont tudnia kell arról, hogy milyen könyvtáraink vannak, mindet be kell olvasnia, és folyamatosan kapcsolatot kell tartania az ügyfelekkel. Mindenki el tudja képzelni, hogy ez – még egy bivalyerős gép esetében is – mekkora feladatot ró a kiszolgálóra: ha például csúcsidőben ezer felhasználó csatlakozik hozzá, nyomon kell követnie a kapcsolataikat és várnia kell a kéréseikre. Ha az IMAP-kiszolgáló igényei ekkorák, hol érdemes egyáltalán használni? Rendszerint olyan munkakörökben, ahol fontos, hogy a felhasználók el tudják olvasni egymás leveleit, tehát általában irodákban, de kis cégeknél is jól alkalmazható. Az utóbbi lehetőséget az IMAP-nak az a tulajdonsága teszi lehetővé, amely képes megosztott könyvtárakat létrehozni és kezelni. Hol lehet ez még fontos, ha közelebbről megnézzük? Például az ügyféltámogatást (support) nyújtó alkalmazottak körében. Ha többen ügyelnek, fontos lehet, hogy mindannyian tudjanak a beérkezett üzenetekről, és ha már olvasták az üzenetet, az eredményét is lássák. Egy üzenet pillanatnyi állapotának jelzését az IMAP egy további hasznos szolgáltatása biztosítja: az úgynevezett zászlók (flag). Ezek segítségével egy üzenetről azonnal megmondható, hogy olvasták vagy esetleg már meg is válaszolták-e. Használatakor további jelzőket is bekapcsolhatunk, például az important (fontos) zászlót. Ha a leveleket a megosztott könyvtáron keresztül a támogatást nyújtókon kívül a vezetőség is el tudja olvasni, lehetővé válik azon ügyfelek panaszának ellenőrzése, akik azt állítják, hogy nem válaszoltak időben kérdésükre, segítségkérésükre. A válaszadás megtörténtét jelző zászló bizonyítja vagy éppenséggel cáfolja az ügyfél szavának hitelességét.

Az ügyfelek szempontjából vizsgálódva tehát az IMAP jobbnak mondható, mint a POP3, a kiszolgálók karbantartói számára azonban kétségkívül az utóbbi sokkal rokonszenvesebb. Nézzük át, melyek azok a kiszolgálóoldali megvalósítások, amelyek a protokollokhoz fellelhetők és valamiért kiérdemlik a figyelmünket.

POP3-kiszolgálók

A legtöbb Linux-változatban vagy a qpopper nevű démonnal vagy az UW IMAP POP3 moduljával találkozhatunk. A qpopper előnyei közé tartozik az évek során köré kialakult bő leírás- és listaarchívum, és néhány folt, amely kibővíti a lehetőségeit, például az SQL-azonosító. Az UW IMAP kedvező tulajdonságai közé sorolható, hogy fejlesztőinek egyike az IMAP-protokoll alkotója. Én egyik programot sem ajánlanám, mert mindkettőnek megvannak a maga gyengéi. A qpopper néha „lefárad" és „elfelejt" működni, amin a többszöri újraindítás sem segít; az UW IMAP kódja pedig nagyon kusza és gyenge, számos biztonsági hibát találtak már benne. Az általam előnyben részesített POP3-kiszolgálók: a TeaPOP (➲ http://www.toontown.org/teapop), valamint a solid pop3d (➲ http://solidpop3.pld.org.pl). Mindkettő nagyon megbízható és támogatja a virtuális felhasználókat, valamint mind a hagyományos mbox, mind a Maildir/ levéltárolási rendszert is ismeri. Rendkívül üzembiztosak és gyorsak. A TeaPOP az Apache által használt htpasswd fájlformátumon kívül különböző nyílt forráskódú SQL-kiszolgálókon (MySQL, PostgreSQL) keresztül tud azonosítani. Ismerik a levélöregedést: ha felhasználó például tíz napon belül nem tölti le a levelét, azt önműködően törlik a kiszolgálóról. Ezzel akadályozhatjuk meg, hogy a felhasználók a POP3-ügyfelet úgy állítsák be, hogy a kiszolgálóról csak letöltse, de ne törölje a leveleket.

A tisztán POP3-as megoldások mellett nézzünk ismét két egyesített POP3/IMAP-megoldást. Az első a Courier-IMAP POP3-modulja, melynek erőssége üzembiztonságában, teljesítményében és a sok azonosító modulban rejlik. Vigyázzunk, mert csak a Maildir, illetve a Maildir++ tárolási formát ismeri!

A másik megoldás a Cyrus-IMAP, melynek fő előnye hihetetlen méretezhetősége és óriási teljesítménye, ez utóbbit egyedi levéltárolási formátumának köszönheti. E két programot azért hagytam utoljára, mert ha megpróbálunk áttérni egy másik megoldásra – amit egyébként nem tartok valószínűnek –, a sok egyedi formátum és beállítás miatt nem lesz könnyű dolgunk.

IMAP-kiszolgálók

Elemezzük a POP3-démonoknál felsorolt egyesített programokat egy kicsit az IMAP oldaláról nézve! Az első az UW IMAP, mely jelenleg a 2000-es változatnál tart. Minősége megegyezik a hasonló változatú programokéval. Hibáin kívül egyetlen említésre méltó tulajdonsága, hogy írói között az IMAP alkotója is szerepel. Minőség tekintetében sokkal jobb a Courier-IMAP és a Cyrus-IMAP.

A Cyrus a már említett méretezhetőséggel és teljesítménnyel, valamint az írói által alkotott SASL-azonosítással emelkedik ki a mezőnyből. A Courier előnye szintén méretezhetőségében és kis memóriaigényében rejlik, valamint beépített SSL-támogatással és virtuális tartománynév-kezeléssel rendelkezik. Tartalmaz továbbá sok beépített azonosító modult, ezek között a PAM is megtalálható – tehát gyakorlatilag majdnem mindenen keresztül tudunk azonosítani.

Az utóbbi kettőnek felróható ugyan a nem túl egyszerű telepítés és a beállítások nehézsége, de ezek is csak azért, mert egyéb hibát nem tudunk felfedezni bennük.

Végszó

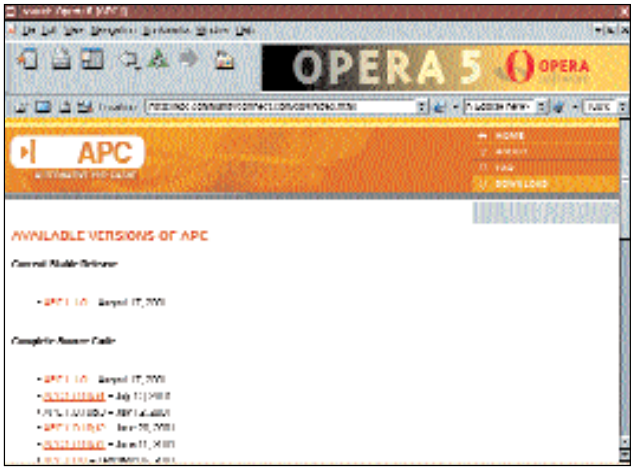
Remélem, sikerült feltárnom a POP3 és IMAP közötti különbségeket. Olvasóink igényeinek feltérképezése alapján írásomat a felsorolt kiszolgálók valamelyike (POP3 vagy IMAP) telepítésének bemutatásával folytatom. Az ezzel kapcsolatos javaslatokat és kéréseket levélben az ➲ imap@lsc.hu címre várom.

Deim Ágoston

Gyorstár-megoldások PHP-hoz

Öveket becsatolni – sebességbe kapcsolunk!

Sok weboldal-üzemeltető cég találkozhat azzal a nehézséggel, hogy oldalainak letöltési ideje a látogatók számának növekedésével jelentősen megnő. Természetesen mint minden gond megoldásánál, itt is több lehetőség közül választhatunk. Akad olyan cég, amely a túlterheléssel járó nehézségeket gépvásárlással orvosolja. Ennél elegánsabb módszer, ha a leírásokban keresgélünk olyan áthidaló megoldás után, amellyel mérsékelhetjük a költségeket, és a megváltoztatandó állapothoz képest programból megvalósított módon érünk el javuló teljesítményt.



➲ http://apc.communityconnect.com/download.html

Írásunkban most az utóbbi megközelítéssel élve próbálunk megoldást lelni a felmerülő gondokra: megismerkedünk az APC-vel (Alternative PHP Cache), a hozzá készített grafikus kezelőfelülettel, valamint a PEAR-rel. A Zend Technologies alkalmazásait pedig épphogy megemlítjük.

Az egyik lehetséges megoldás: az APC

Az APC készítői olyan nyílt forráskódú programot szerettek volna készíteni, amely megállja a helyét az olyan kereskedelmi megoldások mellett, mint például a Zend Cache.
A program működési elve egyszerű: a PHP-kód a fordítás után a memóriába kerül, kivéve, ha már ott is van, mert akkor nem szükséges újra lefordítani. Ha ismét szükségünk van a kód lefordított kimenetére, egyszerűen kiolvassuk a gyorstárból. A sebességnövekedés abból ered, hogy megtakarítjuk az újrafordítási időt. Mekkora mértékű lehet a teljesítményjavulás? Erre nem lehet pontos választ adni, mert a programjainktól is függ. Egyszerűbb programoknál a sebességnövekedés nem lesz szembetűnő, viszont ahol összetettebb a program és az általa megoldandó feladat, ott jó eséllyel számíthatunk kedvezőbb eredményre, mert esetükben lényegesen nagyobb a fordítási idő.

Telepítés

A fordítás és a telepítés kétféle módon történhet.
Ha már létezik a gépünkön PHP, és nem akarjuk az egészet újrafordítani, csupán az APC-t fordítsuk le:

```
./configure --enable-apc
        ➥--with-php-config=/a php-config helye
make
cp modules/php_apc.so /a kiterjesztések helye
```

A php.ini fájlba a következő bejegyzéseket kell beírni:

```
zend_extension= a kiterjesztések helye /php_apc.so
```

A következő három sorból csak egyet kell kiválasztani:

```
apc.mode = off
apc.mode = shm
apc.mode = mmap
```

Az első esetben kikapcsoljuk az APC-támogatást, a másodiknál a lefordított PHP-kódok a memóriába (shared memory) kerülnek, a harmadik esetben pedig egy-egy fájlba (memory mapped file). Erről a későbbiekben még szót ejtünk.
Ha azt szeretnénk, hogy az APC-támogatás bekerüljön a PHP-ba, vagyis az egészet újrafordítjuk, a következőket kell tennünk: csomagoljuk ki az APC-forrást a */PHP_forrás_elérési_útja*/ext/apc könyvtárba. A PHP-forrás legfelső szintjén adjuk ki a következő utasításokat:

```
autoconf
./configure --enable-apc <további kapcsolók>
make
make install
```

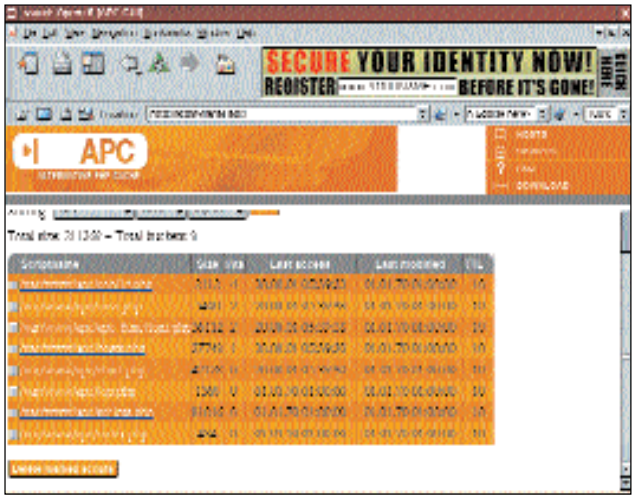
A php.ini fájlba ugyanazt a sort kell beilleszteni, amit az előzőekben a külön modulba fordítás esetén is.

Az APC használata

Először azt kell eldöntenünk, hol szeretnénk a lefordított fájlokat tárolni. A memóriában (shm) vagy fájlban (mmap)? Mielőtt erre válaszolnánk, ismerkedjünk meg az APC-hoz készített grafikus kezelőfelülettel. A programot PHP-ban írták és a PHP GD (Graphical Development) támogatását igényli, mivel a statisztikákhoz képfájlokat készít. Az APC grafikus kezelőfelületet magát a ➲ http://apc.neuropeans.com címről tölthetjük le, de az APC csomagjának *extras* könyvtárában is megtaláljuk. Telepítése egyszerű: másoljuk be az egészet oda, ahol webkiszolgálónk hozzáférhet a PHP-fájlokhoz. Az adatokat a webkiszolgáló vagy a PHP-kód segítségével érdemes védeni.
Az APC-vel való ismerkedés lezárásaképpen megosztanék néhány gondolatot a gyorstárban lévő PHP-programok tárolásával, érvényességével kapcsolatban:

- A memóriában tárolt PHP-fájlokról adatokat csak a grafikus kezelőfelület segítségével szerezhetünk, míg ha fájlban tároljuk őket, a változásokat könnyedén követhetjük.
- Fontos átgondolni a fájlok frissítését és élettartamát. Ha ugyanis valami már bekerült a gyorstárba, akkor érvényességének lejár-

táig mindig onnan fogjuk megkapni. Ez jót és rosszat is jelenthet: jót, mert rendszerünk ezáltal gyorsabb lesz, hiszen az ügyfelek hamarabb kapnak választ a kiszolgálótól; hiba keletkezik viszont, ha a kiszolgálón tárolt programot módosítjuk, miközben az a gyorstárban már megtalálható.



Az APC használat közben

Lehetőségek a hiba kiküszöbölésére:

- PHP programból.

```
if(apc_reset_cache())
    print "Töröltem az összes bejegyzést";
  else
    print "Nem sikerült törölni
          ➥a bejegyzéseket";
```

A függvény csak az shm-nél működik.

- A bejegyzés élettartamának megadása a php.ini-ben.

```
apc.ttl = 10   ;10 másodpercet
               ➥állítottunk be
```

A beállítás csak az shm-nél működik. Általánosan, minden gyorstárban lévő objektumra igaz lesz.

- Egyedileg is beállíthatjuk arra a PHP-programra, amelyből a függvényt meghívjuk.

```
apc_set_my_ttl(10);
```

- Bejegyzés törlése PHP-programból.

```
apc_rm(teljes_elérési_út_megadása);
```

A függvénynek a törölni kívánt PHP-fájlt és ne a gyorstárfájlt adjuk meg!

- A webkiszolgáló újraindítása.
  Ez is egy lehetőség, de inkább ne éljünk vele, ekkor ugyanis az összes élő kapcsolat megszakad.

Tapasztalatok

- Az mmap használata során fontos a gyorstár könyvtárának megadása. A grafikus kezelőfelület érthetetlen okokból a /tmp könyvtárat adta vissza, miközben a php.ini-ben nem volt erre vonatkozó bejegyzés. Ráadásul az APC sem működött rendesen, míg a hibát nem javítottam.

```
apc.mode = mmap
apc.cachedir= /tmp
```

- A grafikus kezelőfelület 1.0.2-es változatának hosts.php-jában érdemes a módosítani következő sort, ellenkező esetben hibaüzenetet kapunk, miszerint nem meghatározott függvényt próbáltunk meghívni.
  A hibás sor:

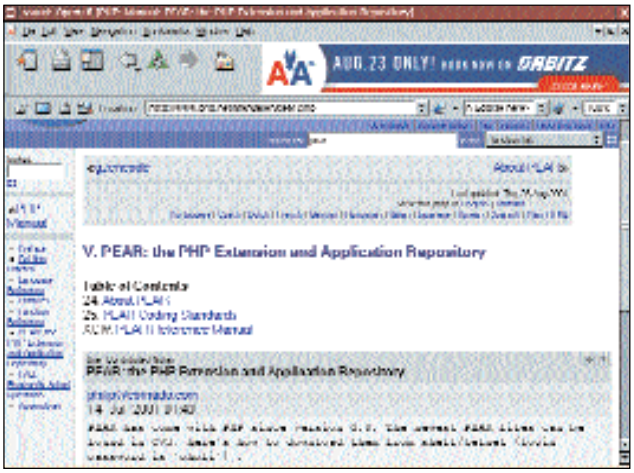
```
apc_restart_host($apc_hosts[$host]);
```

  Helyette:

```
apc_reset_cache();
```

A gyakorlat

Lehetőségünk nyílik arra is, hogy egyszerű PHP-programok segítségével magukat a programkimeneteket tároljuk, így szükség esetén a megfelelő helyről meghívhatók – ez a hely lehet fájl,

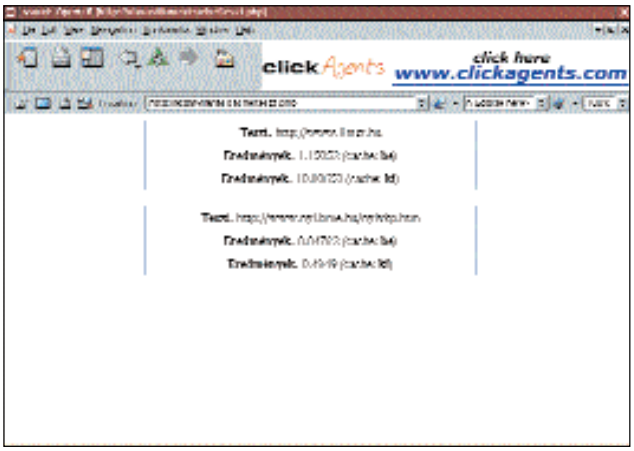


➲ http://www.php.net/manual/en/pear.php

adatbázis vagy memória. Több megvalósítás is ismert, mint a PEAR Cache modulja, a Toncarta Cache, CacheIT stb. Ez nem teljesen azonos a cikk első felében bemutatott módszerrel, de használhatjuk annak kiegészítőjeként.

A PEAR és egy teljesítménypróba

A PEAR segítségével adatok, függvények, URL-ek tartalmának gyorsabb lekérdezésére nyílik lehetőségünk. Két internetes oldal letöltési idejét hasonlítottam össze egy olyan sebességteszt során, amikor is a letöltést többször egymás után megismételtem, majd a letöltési időket a végén összehasonlítottam. A próbát a gyorstár használatával és nélküle is elvégeztem.



Két gép összehasonlítása

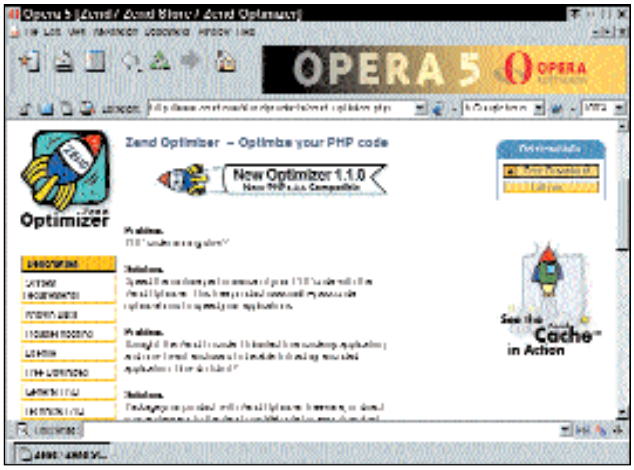
A próbaprogram kódja

```
<?
  include("Benchmark/Iterate.php");
  include("Cache.php");
  include("Cache/URL.php");

  $FUNCTION_CACHE_CONTAINER = "file";
  $FUNCTION_CACHE_CONTAINER_OPTIONS = array(
       "cache_dir" => "/tmp/",
       "filename_prefix" => "cache_");

  function with($cache) {
    $data = get_cached_url($cache,10);
  }

  function without($cache) {
    $fd = fopen($cache,"r");
  }

  $num = 20;
  $benchmark = new Benchmark_Iterate;

  $benchmark->run
➥($num,'with','http://www.php.net/manual/en/');
  $result = $benchmark->get();
  $eredmeny = $result["mean"] *
➥$result["iterations"];
  print $eredmeny . "<br>";
  $benchmark->run
➥($num,'without','http://www.php.net/manual/en/');
  $result = $benchmark->get();
  $eredmeny = $result["mean"] *
              ➥$result["iterations"];
  print $eredmeny;

?>
```

Maga a program rövid, nem bonyolult, viszont alkalmas arra, hogy az eddig leírtakat a gyakorlatban is megvizsgálhassuk. A programok forráskódja a *listán* látható.
Először beillesztjük a megfelelő PHP-kódokat, amelyek a méréshez és a gyorstár használatához szükségesek. Az ezt követő két változó adja meg egyrészt, hogy a letöltött oldalakat fájlokban tárolnám-e, másrészt hogy a kérdéses fájlok mely könyvtárban helyezkedjenek el és milyen előképzőt kapjanak. Ezután létrehozzuk a Benchmark_Iterate osztály egy példányát, majd meghívjuk a függvényeket az iteráció számával, a függvények nevével és a letöltendő oldal URL-jével. Az eredmények a *4. ábrán* láthatók és másodpercben értendők.
A gyorstár kikapcsolása miatt a kívánt oldal minden ciklusban letöltődik, míg a bekapcsolásnál csak egyszer, mert azt követően már a gyorstárból kapjuk meg. Ez adja az eredményekben felbukkanó eltérést.

Zend

Az előbb említett szabad forráskódú lehetőségek mellett kereskedelmi termékek is léteznek.
A Zend Technologies kifejezetten a PHP-hoz készít alkalmazásokat, így például a Zend IDE-t, a Zend Cache-t és a Zend Optimizert. Számunkra ez utóbbi kettő azért fontos, mert képesek PHP-programjaink végrehajtását felgyorsítani. A Zend Cache harmincnapos, szabadon használható ingyenes változatát bárki letöltheti Linux, BSD vagy Solaris operációs rendszerekre. Kódunkat hatékonyabbá teszi és képes a fájlokat a gyorstárban őrizni, hogy szükség esetén onnan tölthesse be őket. A Zend Optimizer teljesen ingyenes, nagy haszna, hogy egyszerűsíti és hatékonyabbá varázsolja programjainkat, így végrehajtásuk a megszokotthoz képest akár száz százalékkal is gyorsabbá válhat.



➲ http://www.zend.com/store/products/zend-optimizer.php

Összefoglalás

Két lehetséges megoldást láthattunk oldalaink felgyorsítására és mindkettővel javulást érhetünk el, főleg ha a gyakorlatban is alkalmazzuk őket. A következő számban egy másik érdekes PHP-modullal fogunk megismerkedni, de ennek neve addig maradjon titok.

Tóth László
(atom@lsc.hu) A BME utolsó éves hallgatója, webfejlesztőként dolgozik a saját vállalkozásában. Szabadidejét legszívesebben a barátnőjével tölti.

Kapcsolódó címek

Az APC hivatalos oldala
➲ http://apc.communityconnect.com

Az APC grafikus kezelőfelület oldala
➲ http://apc.neuropeans.com

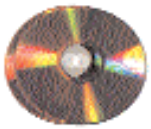
A GD-modul letöltésének oldala
➲ http://www.boutell.com/gd

PHP-ról szóló cikkek és programok gyűjteménye
➲ http://www.zend.com

Zend Optimizer
➲ http://www.zend.com/store/products/zend-optimizer.php

A PHP-nyelv honlapja
➲ http://www.php.net

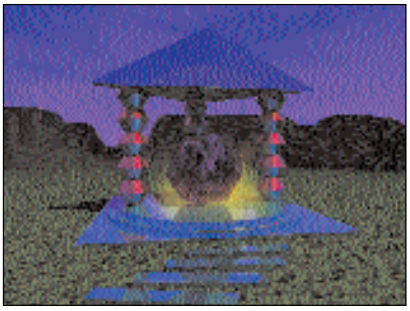
Magyar nyelvű PHP-könyv
➲ http://www.kiskapu.hu/kiskapu/search.phtml?detailed=120449801



PoV-Ray ismeretek (3. rész)

Ebben a részben az objektumok leírásának további módszereivel foglalkozunk.

Most részletesebben tárgyaljuk a Bezier-patch modelleket, a forgástesteket, a magassági térképek alapján készített domborzatokat, a metaball-objektumokat és a háromszög által meghatározott objektumokat. Az olvasottak elsajátítása után – feltéve, hogy a korábbi részeket is „befogadtuk” – elvileg képesek leszünk az alábbi képhez hasonlót alkotni.



A látvány megéri a fáradságot…

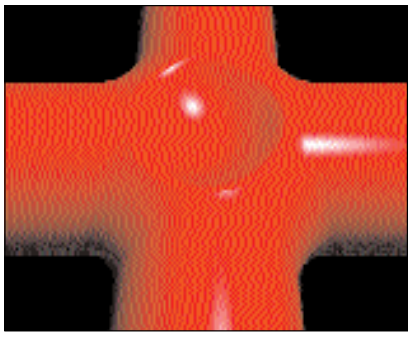
Kezdjük a metaball-objektummal, amit a PoV szóhasználatában egyszerűen *blob*-nak nevezünk. Aki használta valaha a LightWave valamelyik változatát, annak szeme előtt nyilván megjelenik szóban forgó objektumunk képe, a többiek pedig képzeljenek el pontszerű, egypólusú mágneses tereket, amelyek tetszőleges helyzetben egymás mellé vannak téve. A mágneses terek által alkotott eredő mágneses tér határozza meg a *blob* objektum formáját. Természetesen a „mágneseket” elmozdíthatjuk, elérve, hogy a tárgy formája rugalmasan változzék.
Ilyen objektumokat a PoV-Ray leírónyelvével a következő formában adhatunk meg:

```
blob {
     threshold THRESHOLD_ RT K
     cylinder { <EGYIK_V GE>,
     ➥<M`SIK_V GE>, SUG`R,
     ➥[ strength ] VONZER  }
     sphere { <K ZEPE>, SUG`R,
     ➥ [ strength ] VONZER  }
  }
```

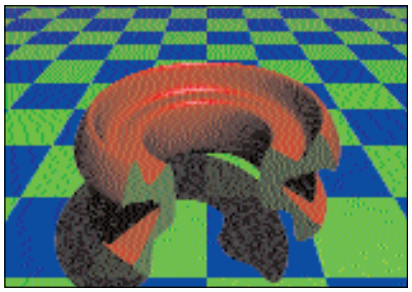
Az objektum látható részeinek kiszámítása során a PoV-Ray ott határozza meg a felület egy pontját, ahol az összetevők által alkotott erőtér és a TRESHOLD_ÉRTÉK megegyezik. Az egyes összetevők által keltett mágneses mező erőssége az összetevő középpontjában a VONZERŐ értékkel egyezik meg, és az objektum határáig folyamatosan csökken nullára. A VONZERŐ értéke negatív is lehet, ekkor a mágneses mező objektumunk felületén nem kidudorodásként jelenik meg, hanem horpadásként.
Itt kell megjegyeznem, hogy egy objektum több összetevőből is állhat, melyeket szabadon torzíthatunk és átméretezhetünk.
A *blob* objektumot gömbökből és hengerekből építhetjük fel; ezen építőelemek segítségével olyan változatos formákat alkothatunk, melyek összetettségének csupán alkotóerőnk szabhat határt. Ezekből a formákból akár egészen bonyolult élőlényt is teremthetünk: a Jurassic Park közkedvelt dinoszaurusza, a Tyrannosaurus Rex szintén metaball-modellezéssel készült, noha nem PoV-Rayjel.
A *lathe* objektum (magyar elnevezése a matematikaórákról ismerősen csengő „forgástest”), melynek segítségével egyszerűen hozhatunk létre vázákat, kelyheket, oszlopokat és hasonló tárgyakat. Az objektum megadásának módját az alábbi lista mutatja:

```
lathe {
     [ linear_spline |
     ➥quadratic_spline |
     ➥cubic_spline ]
     PONTOK_SZ`MA,
     <PONT_1>, <PONT_2>, ...,
     ➥<PONT_PONTOK_SZ`MA>
  }
```

A PONTOK_SZÁMA érték mutatja meg, hány ponttal kívánjuk a forgástestet leíró görbét megadni. Ha a pontok koordinátáit kétdimenziós vektorokként írjuk be, a program a pontokat *lineáris* (linear_spline), *négyzetes* (quadratic_spline) vagy *köbös* (cubic_spline) spline-görbékkel köti össze, és az y tengely mentén forgatva létrehozza a testet. A fentebb említett görbék részletesebb leírását nem kívánom ismertetni, akit bővebben érdekel matematikai előállításuk módja, a könyvtárakban könnyen utánanézhet. Itt elegendő annyit tudnunk, hogy lineáris spline-görbék előállításánál a két pont közti görbeszakasz csak ettől a két ponttól függ, míg a négyzetes és a köbös görbék esetében a pontok közti szakasz alakját a szomszédos pontok is befolyásolják, módosításukkal tehát finomabb formákat hozhatunk létre.



A negatív erő hatása a blob objektumra



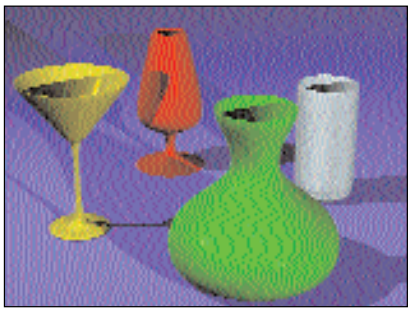
Egyszerű forgástest keresztmetszete

Azt is érdemes megjegyeznünk, hogy a PoV-Ray nem zárja le önműködően a görbét, ha ezt meg akarjuk tenni, az első és az utolsó pontot – azonos koordinátákkal – nekünk kell megadnunk.
A következő objektum, amit tárgyalni fogunk, a *Surface of revolution* – a kifejezést magyar fordításban szintén „forgástestként” adhatjuk vissza. PoV-Ray-beli neve *sor*, és a következő módon hozható létre:

```
sor {
     PONTOK_SZ`MA
     <PONT0>, <PONT1>, ...,
     ➥<POINTPONTOK_SZ`MA-1>
     [ open ]
  }
```

Az értékek megegyeznek az előbbi objektumnál látottakkal, különbséget csupán a testek matematikai előállításának módjában fedezhetünk fel. Ugyanebből a különbségből származik az a tény is, hogy ezek az objektumok kevésbé rugalmasak, mint a *lathe*-objektumok. Ennek oka, hogy min-

den y értékhez csak egyetlen x érték tartozhat. A PoV-Rayt az `open` kulcsszóval utasíthatjuk, hogy a létrehozott test végeit ne zárja le. Ilyenkor a CSG-műveletekben ne használjuk a létrehozott objektumot, mert – a PoV-Ray leírása szerint – az eredmény nem lesz a várakozásainknak megfelelő. Ezzel a megadási móddal zárt görbéket nem határozhatunk meg.



Változatok egy témára

Felmerülhet a kérdés, hogy miért szükséges ez az utasítás? A választ szintén a matematika adja meg: a *lathe*-objektumok kiszámítása során a fénysugár és a test metszéspontjának meghatározásához egy hatodrendű polinomot kell megoldanunk, míg ugyanennek a műveletnek az elvégzéséhez – egy *sor* (Surface of revolution) objektum esetén – egy harmadrendű polinom megoldása is elegendő. A sebességbeli különbség pedig magáért beszél.
A bevezetőben nem említettem ugyan, de szeretnék bemutatni egy nagyon hasznos objektumot, melynek segítségével egyszerűen állíthatunk elő látványos hatásokat, például logókat saját honlapjaink díszítésére. A szövegről, avagy PoV-Ray-es szóhasználattal élve a *text*-ről van szó. Létrehozása igen egyszerű:

```
text {
     ttf "BET K SZLET.TTF",
     "A SZ VEG",
     VASTAGS`G, <ELTOL`S_VEKTOR>
   }
```

A BETŰKÉSZLET.TTF érték határozza meg a használni kívánt betűkészletet, ami jelenleg csak TrueType formátumú lehet. A szöveget ezt követően a C nyelv karakterlánc meghatározási szabályai szerint adhatjuk meg : macskakörmök között. Ez például azt jelenti, hogy ha a kiszámolt képen ' karaktert szeretnénk viszontlátni, akkor az objektum létrehozásakor a szöveg értékben az idézőjelet a következő módon kell meghatároznunk: "\"".
Így a PoV-Rayt arra utasítottuk, hogy a '\' utáni karakter a szöveghez tartozik, és nem annak lezárására szolgál.

A szöveg origója az első betű bal alsó sarkában lesz, a további betűk pedig az x tengely pozitív irányának megfelelően következnek. Az egyes betűk vastagságát a VASTAGSÁG valós szám értéke határozza meg. A betűk alapértelmezésben 1 egység magasak lesznek azért, hogy a .TTF fájlban meghatározott betűközök valószerűek legyenek. Ezt a betűközt felülbírálhatjuk az ELTOLÁS_VEKTOR-ral.
A *text* objektumok kiszámításakor csak a nyomtatható karakterek fognak megjelenni, tehát a *soremelés*-nek, a *tabulátor*-nak és a hozzájuk hasonló vezérlőkarakntereknek a megjelenő szövegre semmiféle hatásuk nem lesz.
A fenti kitérő után visszatérnék a cikk fő témájához, most a domborzati képek megjelenítéséhez elengedhetetlen *height_field* objektumot mutatom be. Ha rendelkezésünkre áll egy olyan domborzati térkép, amelyen a magassági szinteket különböző, egyre világosabb színekkel jelölik, akkor a *height_field* segítségével könnyen és gyorsan alkothatunk tájképeket. Amint a kép elkészült, alakítsuk át a következő képformátumok valamelyikére: gif, pgm, ppm, png, tga, pot. A gif formátum sajátossága, hogy csak 256 színárnyalat tárolására alkalmas. Ebben az esetben domborzatunk nem a színek alapján fog elkészülni, és a domborzati képen létrejövő kiemelkedés az adott képpont (pixel) palettaindexének megfelelő magasságú lesz.
Ezt jól szemléltethetjük egy példával: tegyük fel, hogy a felülnézetből készített gif-képen egy tó látható, amely sötétkék színű, körülötte a hegyek és az erdők világoszöldek. Ekkor joggal feltételezhetjük, hogy a PoV-Ray az elkészített domborzaton a tavat mély területként, az erdőket, hegyeket magasabb területként fogja ábrázolni. Meglepetés érhet minket, ugyanis a gif-állományban a sötétkék színhez magasabb index tartozhat, mint a világoszöldhöz, így domborzatunk végeredményképpen teljesen valószerűtlen tájat mutathat. Ez a hiba nem fordulhat elő, ha a gif formátum 16-bites változatát használjuk (a .POT formátumot), ennek előállítására a FractInt program alkalmas. (A program lemezmellékletünkön is megtalálható, de ezzel a változattal nem lehet pot formátumú képeket létrehozni.) Ilyen gondok a nem indexelt képekkel nem fordulhatnak elő, mert ott a program a magassági értékeket az RGB-összetevők alapján számítja ki.
Ennyi bevezetés után lássuk, hogyan hozhatjuk létre magát a domborzatot:

```
height_field {
  F`JL_T´PUS "F`JLN V"
     [ smooth ]
     [ water_level VAL S ]
  }
```

Itt a FÁJL_TÍPUS a fent említettek valamelyike lehet. A *smooth* kulcsszóval adhatjuk meg, hogy a PoV-Ray alkalmazzon-e simítást a számolás során, míg a *water_level* a tengerszint magasságát határozza meg. A program tengerszint alatti területekkel nem számol.
A megjelenő kép domborzatunktól függetlenül 1x1x1 egység méretű lesz, tehát nagyobb felbontású képek használatával nem hozhatunk létre nagyobb objektumot, viszont növelhetjük a részletességét.
A következő sorok begépelése után a *hf_gray_16* kulcsszó használata a globális beállításoknál szintén részletesebb képeket eredményez:

```
global_settings {
       hf_gray_16 on
}
```

Az objektumok létrehozása során a képek 16-bites színmélységűek lehetnek és a program is eszerint fogja használni ezeket. A PoV-Ray legújabb (3.1g) változata sajnos e ponton nem adott élvezhető eredményt, a kép színei egyáltalán nem feleltek meg a leírófájl alapján várhatóknak.
A most következő objektum teszi lehetővé, hogy a PoV-Rayben régebben elkészített háromdimenziós modelljeinket is felhasználhassuk. Nem másról van szó, mint a háromszögek által meghatároott, a PoV-Rayben *mesh*-nek nevezett tárgyról, mely a következő formában adható meg:

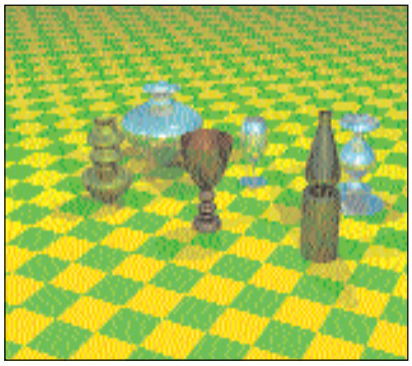
```
mesh {
  triangle {
    <CS CSPONT1>, <CS CSPONT2>,
    <CS CSPONT3>
    [ texture { MINT`ZAT_NEVE } ]
  }
  smooth_triangle {
    <CS CSPONT1>, <NORM`L1>,
    <CS CSPONT2>, <NORM`L2>,
    <CS CSPONT3>, <NORM`L3>
    [ texture { MINT`ZAT_NEVE } ]
  }
```

A fenti listából látható, hogy tárgyunkat kétféle háromszög alkothatja. Összetett objektumok esetén a görbefelületeket szintén háromszögekből építjük fel, de ahhoz, hogy az eredmény sima görbefelület legyen, nagyon sok kisméretű háromszög szükséges. Ezen segíthetünk a *smooth_triangle* alkalmazásával, ahol minden csúcspontnak megadhatjuk a normálvektorát is (ezt egyébként a program számítaná ki). Mivel egy háromszög vala-

melyik csúcspontja más háromszögekhez is tartozhat, nem egyértelmű, hogy az adott pontnak melyik lesz a megfelelő normálvektora, ezáltal a kiszámított képen nem kapunk sima görbületeket. Ha viszont mi határozzuk meg a csúcspont normálvektorát, akkor az eredmény is ránk van bízva. Természetesen a *mesh* objektumok felépítése során nem kell bonyolult számításokat végeznünk, hiszen ezeket az objektumokat általában valamilyen segédprogrammal állítjuk elő, a meglévő modellek átalakításával. E programok az egyes csúcspontok normálvektorát rendszerint a csúcsponthoz tartozó háromszögek normálvektorainak átlagolásával állítják elő. Ilyen átalakítóprogramot találhatunk CD-mellékletünkön is, melynek segítségével LightWave-objektumokat alakíthatunk át a PoV-Ray által ismert formába. Más operációs rendszerekhez több ilyen program is a rendelkezésünkre állhat.
Minden 3D modellezőprogram képes dxf formátumban menteni a modelleket, így ha valaki például a Blender-modellezés képességeit szeretné ötvözni a PoV-Ray sugárkövetéssel számított képeinek valószerűségével, kedvenc modelljeit a megfelelő program birtokában könnyedén átalakíthatja.
Utolsó témánk a *patch*-objektumok létrehozásának és használatának taglalása. A PoV-Ray szóhasználatával élve ezt az objektumtípust *bicubic_patch*-nek nevezzük, amit magyarul talán *folt*-objektumként adhatunk vissza. Ezek a tárgyak 3D-görbék által meghatározott felületek, ahol úgynevezett *vezérlőpontok* (controll point) alakítják ki a test felületét. Hasonlóan a Bezier-spline görbékhez, amelyekről a forgástesteknél olvashattunk, a felületet nem csupán a szomszédos pontok alakítják ki, hanem azok szomszédai is. Ilyen tárgyat a következő utasításokkal hozhatunk létre:

```
bicubic_patch {
     type PATCH_T˝PUSA
     flatness FLATNESS_ RT K
     u_steps U_L P SEK_SZ`MA
     v_steps V_L P SEK_SZ`MA
   <CP1>, <CP2>, <CP3>, <CP4>,
   <CP5>, <CP6>, <CP7>, <CP8>,
   <CP9>, <CP10>,<CP11>,<CP12>,
   <CP13>,<CP14>, <CP15>, <CP16>
 }
```

A PATCH_TÍPUSA jelenleg kétféle értéket vehet fel. Ha az értéke 0, a PoV-Ray csak a *vezérlőpont*-okat tárolja a memóriában, így sokkal kevesebb memória szükséges, viszont a tárgy ábrázolásakor több számítást kell végezni. Ha az érték 1, a program előzetes feldolgozásként több apró foltra bontja a tárgyat, ezáltal csökken az ábrázolás számításigénye, de a memóriaigénye növekszik.
A következő értékeknek a típus meghatározása után a listán látható sorrendben kell követniük egymást. Ezekután kell megadnunk azt a 16 *vezérlőpontot* (háromdimenziós vektorként), amelyek végső soron a test alakját adják. Elkészült tárgyunk felülete mindenképpen érinteni fogja a <CP1>, a <CP4>, a <CP13> és a <CP16> pontokat.
A V_LÉPÉSEK_SZÁMA és az U_LÉPÉSEK_SZÁMA értékekkel adhatjuk meg, hogy a PoV-Ray a testet legkevesebb mennyi sorra és oszlopra ossza fel, miközben a háromszög helyzetét próbálja kiszámítani. A háromszögek legnagyobb számát a következő képlet alapján számíthatjuk ki: darabkák_száma=(2^U_LÉPÉSEK_SZÁMA)*(2^V_LÉPÉSEK_SZÁMA).
Amint a képletből látható, ez az objektumtípus meglehetősen sok memóriát emészthet fel, cserébe valóban szép görbefelületeket eredményez. Ezeket az értékeket célszerű négynél kisebbre állítani, hiszen már hármasnál – amikor a felületet 64 részre osztjuk fel (ez 128 háromszöget jelent) – is megfelelő eredményt mutat.
Amikor a PoV-Ray feldolgozza ezt az objektumot, ellenőrzést végez a pillanatnyilag használt részegységen. Ennek során meghatározza, hogy az eléggé sima-e egy síkbeli téglalaphoz képest. Ennek a próbának az eredményét befolyásolhatjuk a FLATNESS_ÉRTÉK-kel, mely 0 és 1 között lehet. Amikor 0, akkor a PoV-Ray minden esetben felosztja a felületet a V_LÉPÉSEK_SZÁMA és az U_LÉPÉSEK_SZÁMA által meghatározott részre, míg ha a FLATNESS_ÉRTÉK 0-nál nagyobb, a program minden felosztás után megvizsgálja, hogy szükséges-e további darabolás. A nulla értéknél nagyobb FLATNESS_ÉRTÉK használatának előnye és hátránya egyaránt akadhat. Előnye, hogy ha az objektum nem túlságosan görbült, de tartalmaz néhány görbe felületet, a PoV-Ray meghatározza helyüket, és a részegységekre bontást ezeken a helyeken végzi el – vagyis a simább felületeket nem bontja fel feleslegesen. Legnagyobb hátránya, hogyha a program a tárgy egy részét nem osztja további részegységekre, de a velük határos részt viszont igen, ekkor előfordulhat, hogy tárgyunk kicsit összetörten kerül elő a számítások tengeréből. Rendszerint nagy lyukat fogunk rajta találni, amelyen át is lehet látni. E lyuk észlelhetősége nagymértékben attól a szögtől függ, amelyből az objektumot nézzük, tehát előfordulhat, hogy első ránézésre nem feltétlenül pillantjuk meg.
Ahhoz, hogy az elképzeléseinknek megfelelő objektumokat kapjuk, be kell szereznünk az előállításukhoz szükséges segédprogramokat. Olyan programot, ami kifejezetten Linux alá készült volna, mindeddig nem leltem, de Windows alá elérhető a Hamapatch. Aki szívesen kipróbálná, letöltheti a
➲ http://www.hamapatch.t2u.com címről.
Jelenleg készül a Nurbana nevű NURBS modellező, de még nincs használható állapotban. Kísérletező kedvű olvasóink a
➲ http://www.nurbana.cx címen találhatják meg.



Buli után…

A PoV-Ray által létrehozott *folt*-objektumok önműködően simított felülettel készülnek, a felhasználók viszont olyan segédprogramokat is alkalmazhatnak, amelyek a foltok csoportjait finomabban alakítják ki. Miután elsajátítottuk az összetett objektumok előállításához szükséges ismereteket, mindenkinek ajánlom a lemezmellékletünkön található példák és programok megtekintését. A programok között néhány átalakítóprogram is fellelhető, így a *Blender*-objektumok PoV-Ray-formátumra való átalakítását segítő Python parancsfájlok, és egy DOS(emu) alatt használható metaball-modellező program. A példák a PoV-Rayhez adott minták közül valók.
A szemet gyönyörködtető látvány megtekintése után fogjon mindenki bátran a PoV-Ray használathoz!
Mindenkinek kellemes alkotást kívánok!

Fábián Zoltán
(dzooli@freemail.hu, dzooli@yahoo.com)
23 éves, jelenleg programozóként dolgozik. Szabadidejében szívesen kirándul, túrázik. Emellett szeret rajzolni, érdekli a 3D grafika és a Linuxszal kapcsolatban minden olyan program és programnyelv, amit még nem ismer vagy nem próbált ki.

Gumimatraccal vagy légykukaccal indítsuk a Linuxot?

A címben olvasható magyarításért egyszer talán nívódíjat kapok, de addig is az eredeti angol elnevezéseket használom írásomban, melyben a GRUB rendszerindítót mutatom be.

Legtöbbünk találkozott már a LILO-val, amely minden rendszerindításkor arról gondoskodik, hogy a Linux elinduljon. Neve nem az angol lilo (felfújható gumimatrac) szóból ered, ahogyan azt a magyar olvasó első pillantásra gondolhatná, hanem a LInux LOader szavak összevonásával a Unixban megszokott módon kiagyalt mozaikszó, melynek jelentése: *Linux-betöltő*. A GRUB a másik számos jó tulajdonsággal bíró rendszerindító, amelyet *Erich Boleyn*, *Gordon Matzigkeit* és *Okuji Yoshinori* fejleszt.

A GRUB egy mozaikszót takar, melyet a GRand Unified Bootloader szavakból raktak össze és ekként magyarítható: Nagy Egyesített Rendszerindító. A szerzőket nyilván nem zavarta, hogy a született angol nyelvű felhasználó a szó hallatán lárva, légykukac vagy egyéb dögféregcsali típusú képzettársításokra ragadtathatja magát. Ha rendszerindítóként a GNU GRUB-ot választjuk, a számítógép bekapcsolása után átveszi a vezérlést a BIOS-tól, majd betölti és felélesztí a rendszermagot. Ez utóbbi felel a rendszer többi részének indításáért. A GRUB egyelőre szorosan kötődik a PC-hez, de tervezik, hogy később másféle számítógép-kiépítésre is átviszik. PC-n viszont számos operációs rendszert tud indítani, például az OS/2-t vagy a Windowst. Három BIOS-hívás használatával képes megtalálni a számítógépben lévő összes memóriát, és az eredményt jelenti a rendszermagnak. A továbbiakban tehát szükségtelen, hogy a felhasználók a `mem=`*`xxxm`* értéket kézzel írják be, a memória mennyiségének felderítése ugyanis önműködően zajlik. Amennyiben az adott gépen támogatott, a GRUB használja a logikai blokkcímzést (LBA), ezzel sikerült például az „1024 cilinder”-bonyodalmat megoldania, mellyel szemben a LILO tehetetlennek bizonyult.

A felhasználók szempontjából ez azt jelenti, hogy a GRUB az egész merevlemez elérésére képes, azaz bárhonnan tud operációs rendszert indítani. Nem tudom kellően hangsúlyozni, hogy e két utóbbi GRUB-tulajdonság milyen fontos a számomra! S valóban, *Gordon Matzigkeit* GRUB-fejlesztő némi szakmai elfogultsággal két részre osztja a PC-felhasználókat, így LILO- vagy GRUB-rendszerek használóiként jellemzi őket. Szerinte a számtalan program között az indítóprogramok a legfontosabbak, ezért róluk kellene elnevezni az operációs rendszereket. Lehet, hogy szélsőséges a véleménye, de mindenképpen elgondolkodhatunk rajta, hogy miért „Linux” a Linux, hiszen a szó csak a rendszermagot takarja, és nem az egész rendszert. Talán igazságosabbak lennénk a GNU-fejlesztőkkel, ha „GNU/Linux”– vagy „GNU/Hurd”-rendszerekről beszélnénk, együtt említve a rendszermagot és a körülvevő rendszert.

A GRUB telepítése

Én a GRUB-bal a *Linuxvilág* májusi számának Progeny Debian CD-mellékletén találkoztam először, amikor a rendszer telepítése során dönthettem, hogy a GRUB-ot vagy valami mást választok-e rendszerindító programként. Némi tétovázás után a GRUB-ot jelöltem be. Később olvastam el az angol nyelvű leírást, és elsőre mindent meg tudtam csinálni vele. Ezt a nem túl terjedelmes segédletet az `info grub` paranccsal hívhatjuk elő, amit HTML-formában is olvashatunk a /usr/share/doc/grub könyvtárban.

Az írásomban említett összes fájlhoz a megadott útvonal a Progeny Debian faszerkezetében elfoglalt helyekre utal, ugyanezek a fájlok azonban más változatokban esetleg máshol helyezkedhetnek el.

A Progeny Debian-rendszerben a GRUB alapértelmezett, és a `grub_0.5.96.1progeny5.deb` csomagban található. Az újabb változatra vágyók nézzenek körül az ➲ ftp://alpha.gnu.org/gnu/grub FTP-kiszolgálón. A letöltendő fájl neve mindig a következő formátumú: `grub-`*`változat`*`.tar.gz`, tehát az általam használt `grub-0.5.96.1.tar.gz` helyett nagyobb változatszámot kell keresned! Azt is látjuk, hogy a GRUB alfaszakaszban formálódó program, még messze nem éri el az 1.0-s változatot. Nyilván ez a magyarázata annak, hogy csak most kezd felbukkanni a különböző terjesztésekben.

A `tar` programmal becsomagolt, majd a `gzip` által tömörített `tar.gz` kiterjesztésű csomagokat többféleképpen is kibonthatjuk. A művelet két lépésben tehető meg:

```
gunzip grub-0.5.96.1.tar.gz
```

Ez kicsomagolja a tömörített fájlt, és egyetlen tar kiterjesztésű fájlt hagy maga után, amit második lépésben a `tar` programmal kell feldarabolnunk:

```
tar -xvf grub-0.5.96.1.tar
```

A `gunzip` parancs egyébként egyenértékű a `gzip -d grub-0.5.96.1.tar.gz` alakkal. A `gzip` a tömörítő program, amit azonban a `-d` vagy a `--decompress` vagy az `--uncompress` kapcsolókkal kicsomagolásra is használhatunk. A fenti két lépést össze is vonhatjuk, mivel a tar program alkalmas rá:

```
tar -xzvf grub-0.5.96.1.tar.gz
```

Itt a `-z` kapcsoló jelzi a `tar` programnak, hogy a `-f` (azaz fájl) kapcsoló által kijelölt fájlt a `gzip`-pel kell kitömörítenie. Használhattuk volna még a `-z` kapcsoló helyett a hosszabb `--gzip` vagy `--ungzip` kapcsolókat is. Például:

```
tar --extract --verbose --ungzip
➥--file grub-0.5.96.1.tar.gz
```

(Az `extract` kapcsoló az `x`, a `verbose` kapcsoló pedig a `v` helyett áll.) Egzotikusabb forma a `zcat`. Ez egyenértékű a `-c` vagy `--stdout` vagy `--to-stdout` kapcsolóval ellátott `gzip` paranccsal. Ezek a kapcsolók arra kényszerítik a `gzip` programot, hogy kimenetét a parancssorra irányítsa:

```
zcat grub-0.5.96.1.tar.gz
```

Ezzel nem sokra megyünk, főként nagy fájlok esetén, ezért célszerű a `zcat` kimenetét a `|` [szűrő (pipe) karakter] használatával egy csővezetékben a tar programhoz átirányítani:

```
zcat grub-0.5.96.1.tar.gz | tar
➥-xv
```

A `-` kapcsoló jelzi a `tar`-nak, hogy ne keressen fájlt. Egy másik változat:

```
gzip -d < grub-0.5.96.1.tar.gz
➥| tar -xp
```

A tar `-p` kapcsolója megőrzi a hozzáférési jogokat. A `-p` helyett a `--same-permissions` vagy `--preserve-permission` szavakat is írhattuk volna.

A kibontás után létrejön a grub-0.5.96.1 nevű könyvtár, amely a forrásfájlokat tartalmazza. Át kell lépnünk ebbe a könyvtárba, és itt kell kiadnunk a szokásos fordítási utasításokat:

```
$ cd grub-0.5.96.1
$ ./configure
$ make
$ su
Password:
# make install
```

Ha ez elsőre nem történik meg, akkor még az INSTALL fájlt is elolvashatjuk, de ha nagyon nem megy, célszerűbb lesz DEB- vagy RPM-csomagokat keresnünk a Világhálón. Ezek az utasítások a megfelelő helyekre rakják szét a fájlokat, de a GRUB-ot telepíteni kell. Unix-típusú rendszerekben írtak ehhez egy segédeszközt is , a `/usr/sbin/grub-install` programot. Használata igen egyszerű:

```
# grub-install /dev/hda
```

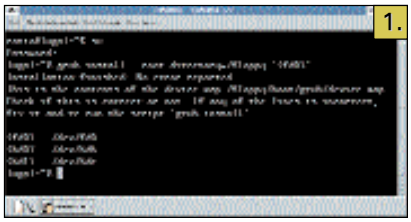
Ez az első IDE-merevlemez rendszerindító területébe (Master Boot Record – MBR) fogja tenni a GRUB-ot. A sor elején található `#` karakter mutatja, hogy mindezt rendszergazdai jogosultsággal tehetjük meg. Nem feltétlenül szükségszerű, de szokásos, hogy a Bash-héjban a rendszergazdai (root) jogosultságot a parancssoron a `#` karakter jelzi, míg a felhasználók a `$` készenléti jelet kapják. Fentebb láthattuk, hogy a beállítást és a fordítást egyszerű felhasználóként végeztem, de a `make install` parancsot már rendszergazdai jogosultsággal kellett kiadnom, hiszen a rendszer területére írtam. Amennyiben csak a saját HOME könyvtáramba telepítettem volna, természetesen nem lett volna szükségem a rendszergazdai jogokra. Ha eredetileg nem rendszergazdaként jelentkezünk be, akkor ezt a jogot később is elnyerhetjük a `su` parancs kiadásával, miután a következő sorban beírjuk a rendszergazdai jelszót.
Ha telepítéskor külön indító lemezterületet hoztunk létre, ennek helyét meg kell mondanunk a GRUB-nak, máskülönben nem fogja megtalálni az indítókönyvtárat:

```
# grub-install --root-
➥directory=/boot /dev/hda
```

Ugyanez a helyzet, ha hajlékonylemezen hozunk létre fájlrendszert:

```
# superformat /dev/fd0 hd
# mkfs -t minix /dev/fd0 1440
# mount /dev/fd0 /floppy
# grub-install --root-
➥directory=/floppy '(fd0)'
# umount /floppy
```

A fenti utasítássor először megformázza az üres hajlékonylemezt, majd minix fájlrendszert készít rajta. (Használjuk a `# mke2fs ➥/dev/fd0` parancsot, ha ext2 fájlrendszert szeretnénk!) A következő sor `mount` parancsa csatolja a hajlékonylemezt a korábban, a rendszer telepítésekor alapértelmezetten létrehozott /floppy könyvtárba, majd a `grub-install` parancs átmásolja a /floppy/boot/grub könyvtárba a szükséges fájlokat, melyek alapértelmezetten a /usr/share/grub/i386-pc könyvtárban találhatók. Ezeket a fájlokat a grub-install az ekkor létrehozott `device-map` fájllal egészíti ki. Az utolsó sorban az `umount` paranccsal lecsatoljuk a hajlékonylemez-meghajtót.


1.

Az *1. képen* látható `grub-install` parancs még nem hoz létre valódi indítólemezt. A GRUB leírása szerint ezt a következőképpen kell megtenni:

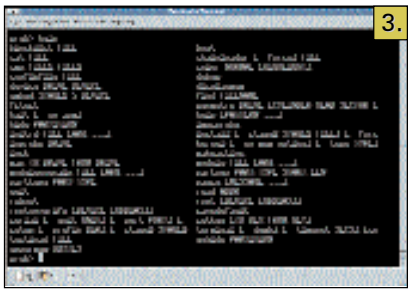
```
# cd /boot/grub
# dd if=stage1 of=/dev/fd0
➥bs=512 count=1
1+0 records in
1+0 records out
# dd if=stage2 of=/dev/fd0
➥bs=512 seek=1
153+1 records in
153+1 records out
#
```


2.

A `dd` parancs a bemeneti állományt (`if=xxx`) a kimeneti állományba (`of=xxx`) másolja át, de a `cp` másolóparanccsal ellentétben a `dd` esetében megadhatjuk a másolandó blokkméretet (`bs=xxx`), illetve másolás közben további változtatásokat is végrehajthatunk. Rendszergazdai jogon lefuttattam a fenti parancsokat, de a második lépésben a *2. képen* látható hibaüzenetet kaptam: Végül a fenti hiba miatt inkább a GRUB parancssort használtam.

A parancssor

Ha rendszergazdai jogosultsággal meghívjuk a `/usr/sbin/grub` programot, a GRUB parancssorba jutunk. Ennek formai követelménye hasonló a Bash parancssorhoz, de annyira azért nem kimunkált, mint nagynevű elődjéé. Ismerkedés céljából elsőként írjuk be a `help` szót. A *3. képen* láthatjuk, hogy jó néhány ismerősnek tűnő parancs közt pár


3.

egzotikusabb is kiírodik. A parancsok rövid leírása a /usr/share/doc/grub/grub_13.html fájlban olvasható. Érdemes rögtön megjegyeznünk a `quit` parancsot, mert ezzel léphetünk ki a GRUB-héjból. A `reboot` például újraindítja a rendszert, a `root` paranccsal pedig azt az eszközt és lemezterületet adhatjuk meg, ahová a gyökérkönyvtárat feltelepítettük. Ha például a gyökérkönyvtár a hajlékonylemezen található, írjuk ezt:

```
grub> root (fd0)
 Filesystem type is ext2fs,
  using whole disk
```

Ha a második lemez harmadik lemezterületén:

```
grub> root (hd1,2)
 Filesystem type is reiserfs,
 partition type 0x83
```

Az `'(fd0)'` kifejezés egyenértékű a Linux-rendszerben használatos /dev/fd0 írásmóddal; olyan, mintha a korábbi példában nem `grub-install /dev/hda`-t írtunk volna, hanem `grub-install '(hd0)'`-t.
Ezek a leképezett értékek kiolvashatók a /boot/grub/device.map fájlból, és inkább a meghajtók BIOS-beli helyzetére, mint a rendszerben alkalmazott számozásra vonatkoznak. A device.map fájl nálam a következőképpen néz ki:

```
(fd0)   /dev/fd0
(hd0)   /dev/hdb
(hd1)   /dev/hde
```

Az első oszlop tartalmazza a BIOS-eszközöket. Ezek az általam adott példában az A, C és Ext meghajtóknak felelnek meg, azaz a hajlékonylemeznek, az IDE- és az Ultra ATA-meghajtónak. A vizsgált gépen ezek mindegyikéről történhet rendszerindítás. A második oszlop a fájlneveket foglalja magába. Tudjuk, hogy a Unixban minden fájl, így a rendszer, a fenti eszközöket különleges állományokként kezeli. Azért szükséges ez a hozzárendelés, mert a GRUB nem mindig ismeri fel megfelelően az eszközöket, és ilyenkor szerkesztenünk kell a device.map fájlt. Látjuk, hogy a GRUB formai követelményei szerint az eszközök neveit `( … )` zárójelek közé kell tennünk. A hajlékonylemez azonosítója esetében az fd magát a meghajtót jelenti, a mögötte lévő szám pedig azt, hogy hányadik meghajtó. A hd merevlemezt jelent, tekintet nélkül arra, hogy IDE-, Ultra DMA- vagy SCSI-lemezről van-e szó. A mögötte álló szám pedig azt jelzi, hogy az első, második vagy harmadik stb. meghajtóval van-e dolgunk. A számozás nullával kezdődik, tehát az első meghajtó száma 0, a másodiké 1 és így tovább. A merevlemezeken található lemezterületeket (partition) újabb számmal különböztetjük meg. Az első lemezen lévő második lemezterületet például így jelöljük: `(hd0,1)`. A lemezterületek számozása ugyancsak nullával kezdődik. Az első kibővített (extended) lemezterületen

létrehozott logikai meghajtó száma 4, azaz a következőképpen jelöljük: `(hd1,4)`.
A GRUB a fájlokat is e jelölés segítségével találja meg. Teszem azt, ha a GRUB az első merevlemez harmadik lemezterületére lett telepítve, akkor az ott található device.map fájlra így hivatkozhatunk:

```
(hd0,2)/boot/grub/device.map
```

Ez az abszolút címzési mód. Ha a `root` paranccsal (lásd fentebb) korábban már tudtára adtuk a GRUB-héjnak, hogy a / gyökérkönyvtár melyik eszközön helyezkedik el, akkor használhatunk relatív címzést is, mivel a GRUB számára már ismert a példában szereplő `(hd0,2)` eszköz. Ilyenkor elég ennyit írni:

```
/boot/grub/device.map
```

A Unix-változatokban elvárható, hogy egy magára valamennyire is adó héjban Ctrl+I vagy a Tab billentyű megnyomásakor parancskiegészítést kapjunk. Nincs ez másképp a GRUB-héjban sem. Ha például csak annyit gépelünk, hogy `ker`, majd leütjük a tabulátort, akkor a parancs `kernel`-re egészül ki. Ha több lehetséges parancs közül választhatunk, természetesen listát kapunk:

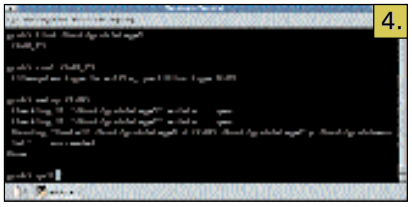
```
grub> h
 Possible commands are: halt
 help hide
```

Ha mondjuk nem tudjuk, milyen meghajtónevek találhatók a gépben, a `find (` parancs begépelése után megnyomhatjuk a Tab billentyűt, hogy megjelentessük a lehetséges lemezek listáját:

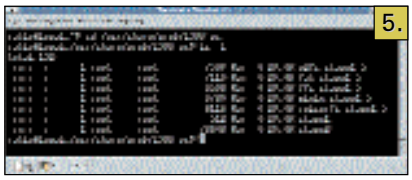
```
grub> find (
 Possible disks are:  fd0 hd0
 hd1
```

GRUB-parancsok futtatásakor is követhetünk el hibákat, ilyenkor ehhez hasonló hibaüzeneteket kapunk:

```
grub> root (fd0,)

Error 12: Invalid device
 requested
```


4.

A lehetséges hibaüzenetetek listája a /usr/share/doc/grub/grub_14.html fájlban található.
Miután a `dd` paranccsal nem tudtam indítólemezt készíteni, a parancssoron próbáltam meg a kézikönyvben leírtak szerint. Először a GRUB-héj `find` parancsával megnéztem, hogy a stage1 fájl hol található. A kapott `(hd0,2)` eredmény mutatja, hogy az első merevlemez harmadik lemezterületén volt. A másodszor kiadott `root` parancs tudatja a GRUB-bal, hol helyezkedik el a gyökérkönyvtár. Ezt követően már lefuttathattam a `setup` parancsot, amely az `install` parancs segítségével telepíti a GRUB-ot a kapcsolójában megadott `(fd0)` lemezre. Ha a `setup (hd0)` parancsot adtam volna ki, akkor a GRUB az MBR-be került volna. Ha az első lemezrész indítószektorába szerettem volna tenni, a parancs a következőképpen nézett volna ki: `setup (hd0,0)`. Amennyiben a GRUB-ot nem az első meghajtó első lemezterületére tesszük, azt több lépcsőben (`chainload`) kell egy másik rendszerindító programmal betölteni. Azért beszélünk ilyenkor chainloaderről (azaz láncolt betöltőről), mert az egyik rendszerindító első láncszemként tölti be az utána következő második láncszemet, a másik operációs rendszer indítóprogramját, esetleg egy másik köztes indítót.
A GRUB telepítésekor először az 512 bájt nagyságú első szakasz, azaz a stage1 fájl másolódik be az MBR-be vagy az indítóterületre a /usr/share/grub/i386-pc könyvtárból. Azért pontosan 512 bájt a mérete, mert ennyit és nem többet lehet ezekre a területekre átmásolni. Ennek a programnak mind-


5.

össze az a feladata, hogy a másfeledik (stage1_5) vagy a második (stage2) szakaszt betöltse. A másfeledik szakaszból több is akad, és mindegyikük felismeri a rá jellemző fájlrendszert, amit a nevükből is rögtön ki lehet találni.
Ezeknek a fájloknak szintén viszonylag kicsi a méretük, ezért rögzített helyre írhatók, közvetlenül a stage1 fájl után. Az adott fájlrendszerre jellemző stage1_5 programszakasz fogja betölteni az immár terjedelmesebb, 78848 bájt méretű stage2 fájlt, amely a GRUB magját tartalmazza.

A menüfelület

Rendszerindításkor arra is lehetőség nyílik, hogy rövid menü jelenjen meg, amely megmutatja a felhasználóknak, hogy a le és fel nyilak segítségével milyen operációs rendszerek közül választhatnak. A kiválasztott sor kivilágosodik, és az Enter leütésekor megkezdődik a rendszerindítás. Ez a lehetőség a LILO esetében is adott. Ami ehhez képest új a GRUB-ban, az az, hogy ezt a menüt menet közben is lehet szerkeszteni, azaz még az indítás szakaszában. Előnyös tulajdonság, hiszen ha a lilo.conf fájlban elírtam valamit, annak komoly büntetése volt, mivel csak a rendszer teljes újraindítása után nyílt lehetőségem a javításra. A GRUB menüben a szerkeszteni kívánt sor a le és fel nyilakkal választható ki, amit az `e` betű leütése követ (feltételezem, az „e” az edit („szerkeszt”) szó első betűjéből ered).
A megjelenő sorokból szintén választani kell, majd ismét az `e` következik, ha a kiválasztott sort szeretnénk szerkeszteni. Ha a szerkesztett sort jónak találjuk, használjuk az Enter, ha nem, az Esc billentyűt. A GRUB leírása szerint az Esc megnyomása egyben visszavonásként (`redo`) is működik, mivel a menüben rögtön az előző állapothoz kerülünk vissza. A kis `o` betű leütésével az éppen szerkesztett sor mögé szúrhatunk be egy sort, a nagy `O`-val pedig a sor elé. Ha a `d` betűt használjuk, az egész sor törlődik. Természetesen egyszerűbb a menüt egy szövegszerkesztőben megírni és javítani. A menü megváltoztatásához a /boot/grub/menu.lst fájlt kell átírni. A vizsgált gépen ez a fájl a következőképpen néz ki:

```
timeout 10
default 0

# --> PROGENY START (1.0) <--

title Progeny Debian (kernel
➥2.4.2)
root (hd0,2)
kernel /boot/vmlinuz-2.4.2
➥root=/dev/hdb3 ro
initrd /boot/initrd-2.4.2.gz

title SuSE 7.0 hdb2 (kernel
➥2.2.16)
root (hd0,1)
kernel /boot/vmlinuz
➥root=/dev/hdb2 ro
initrd /boot/initrd

title Windows Millenium
root (hd0,0)
makeactive
chainloader +1

title Progeny Debian (kernel
➥2.2.18)
root (hd0,2)
kernel /boot/vmlinuz-2.2.18
➥root=/dev/hdb3 ro
initrd /boot/initrd-2.2.18.gz

title Progeny Debian - single-
➥user (kernel 2.4.2)
root (hd0,2)
kernel /boot/vmlinuz-2.4.2
➥root=/dev/hdb3 ro single
initrd /boot/initrd-2.4.2.gz

# --> PROGENY END <--
```

Az első sor timeout 10 kifejezése mondja meg a rendszerindító GRUB-nak, hogy a menüt 10 másodpercig jelenítse meg, utána önműködően indítsa el a `default` parancs értékében meghatározott menüpontot, hacsak a felhasználó másként nem rendelkezik. A 0 érték az első menüpontot jelenti, így példánkban ez az alapértelmezett, de más számot is választhattunk volna. A # jel mögött találhatók a megjegyzések. A `title` parancs mögötti szöveg lesz olvasható a menüben. A `root (hd0,2)` paranccsal megadjuk, hogy melyik meghajtó melyik lemezterületén található a gyökérkönyvtár, a `kernel` parancs betölti a kívánt rendszermagot, az `initrd` pedig a meghatározott initrd fájlt használja indításkor. A `kernel` parancs második részében a rendszermag parancssori (kernel command line) ismert értékeit adhatjuk meg, amelyeket a LILO-nál az `append` fűzött a többihez.

A Windows indítása ettől annyiban tér el, hogy a GRUB ezt az operációs rendszert nem közvetlenül indítja. Először a `root` vagy a `rootnoverify` paranccsal meghatározza a Windows helyét, majd a `makeactive` paranccsal kijelöltí azt a lemezterületet, és végül elindítja az ottani rendszerindító programot. Itt feltételezzük, hogy az indítóprogram (boot loader) ugyanannak a lemezterületnek az indítóterületén (boot sector) található, ahová magát az operációs rendszert is telepítettük. Mivel a Windows a láncolt indítás szempontjából nem támogatott operációs rendszer, esetében használhatjuk a `rootnoverify (hd0,0)` parancsot, amely a `root` paranccsal ellentétben nem próbálja meg befűzni a lemezterületet azért, hogy meghatározza annak méretét és a meghajtó típusát.

A /usr/share/doc/grub/menu.lst egy mintafájl, amely jól használható példa annak megmutatására, miképpen kell a GNU/Hurd, a Linux, a Mach, a FreeBSD, az OS/2, a DOS és a Windows operációs rendszereket indítani. Az ugyanitt lévő grub_5.html fájlban olvashatunk róla egy nem túl bőbeszédű ismertetőt.

Játékosabb kedvű GRUB-felhasználók a `color` parancs segítségével megváltoztathatják a menü színeit a menu.lst fájlban, vagy a GRUB parancssoron:

```
title Villogok
color yellow/green blink-light-
➥cyan/magenta
```

A színek párban állnak. Az elöl lévő szín az előtér, a hátulsó a háttér. A `color` parancs utáni első színpár a normál szín, a második pedig a kiemelt szín. Az előtér és a háttér színei a következők lehetnek: black (fekete), blue (söétkék), green (zöld), cyan (égszínkék), red (vörös), magenta (bíbor), brown (barna), light-gray (világosszürke). A most következő színek viszont csak az előtér színeiként használhatók: dark-gray (sötétszürke), light-blue (világoskék), light-green (világoszöld), light-cyan (kék), light-red (világos piros), light-magenta (világos bíborvörös), yellow (yellow), white (fehér). Ha a szín elé a blink szót írjuk, az előtérszín villogni fog.

A menu.lst fájlba beírt változtatások érvényesítéséhez semmilyen parancsot nem kell lefuttatni, azok a következő indításkor életbe fognak lépni.

A begépelt parancsok és kapcsolók a következő billentyűkombinációkkal szerkeszthetők

A Ctrl-P vagy a felfelé nyíl visszahozza a korábban beírt parancsokat.
A Ctrl-F vagy a jobbra nyíl egy karakterrel előremozgat.
A Ctrl-B vagy a balra nyíl egy karaktert hátraugrik.
A Ctrl-A vagy a Home a sor elejére ugrik.
A Ctrl-E vagy az End a sor végére mozgat.
A Ctrl-D vagy Delete törli a kurzor alatti karaktert.
A Ctrl-H vagy backspace a kurzortól balra lévő karaktert törli.
A Ctrl-K törli a szöveget a kurzor jelenlegi helyzetétől a sor végéig.
A Ctrl-U törli a szöveget a kurzortól a sor elejéig.
A Ctrl-Y beszúrja a törölt szöveget a kurzor pozíciójába.

Kipróbálás

A `grub` héjparancsainak igazi haszna nem a rendszer futása során, hanem rendszerindításkor válik nyilvánvalóvá. Ha a `grub`-ot telepítettük, annak jó ismerői a rendszer indulásakor sokfélét megtudhatnak a gépről, és a kipróbálás céljából különböző parancsokat hajthatunk végre. A `find` paranccsal például kiírattathatják az összes meghajtót:

```
grub>  find (
       Possible disks are:
        fd0   fd1   hd0   hd1
```

Fájlokat listázhatunk ki:

```
grub>  cat
    ➥(hd0,2)/boot/grub/menu.lst
```

vagy tetszőleges kapcsolókkal indíthatjuk a rendszert, mint például az alábbi esetben `grub` parancssorból indítottam a rendszert:

```
grub> kernel
➥(hdb0,2)/boot/vmlinuz-2.4.2
➥root=/dev/hdb3 ro mem=128M
grub> initrd
➥(hdb0,2)/boot/initrd-2.4.2.gz
grub> boot
```

Bizonyos esetekben a rendszermag betöltése után szükséges lehet a `module` vagy `modulenounzip` parancsok kiadása is, hogy modulokat töltsünk be. Az utóbbi nem csomagolja ki a modulokat önműködően. Figyeljük meg, hogy a `mem=128M` rendszermag parancssori értékkel meghatároztam a rendszer számára látható memória mennyiségét, a valós értéktől függetlenül. Ez olyankor lehet hasznos, amikor a programírás során arra vagyunk kíváncsiak, miként viselkedik egy nagy teljesítményű gépen fejlesztett, memóriaigényes program, ha negyedannyi memória jut neki.

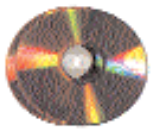
Segédletek

A /usr/share/grub könyvtárban található még egy *Jeff Licquia* által írt lilo2grub nevű Python parancsfájl, ami a LILO lilo.conf beállítófájlt fájlt a GRUB számára is fogyasztható menu.lst alakra hozza.

A nuni

Nem állíthatom, hogy sikerült mindent elmondanom a GRUB rendszerindító programról. Például nem érintettem a biztonsági kérdéseket vagy a hálózati indítási lehetőségeket. Azt sem tanácsolom senkinek, hogy eddig jól működő LILO-ját most törölje, és helyébe a GRUB-ot tegye, hiszen ennek nem sok értelme lenne. Ugyancsak nem állítom, hogy csak a LILO és a GRUB versenyez a rendszerindító programok között. Ott van például a nuni, ami egy kicsi, Linuxra írt rendszerindító az ext2 fájlrendszerre és IDE-meghajtókra. A nuni is túl akar lépni az 1024 cilinderes határon, és 128 GB-ig bárhonnan indíthat a meghajtóról. Kissé terjedelmesre sikerült, ezért pillanatnyilag csak hajlékonylemezekre telepíthető. Fejlesztője az amerikai Oregonban élő *Neil Koozer* (nkoozer@rosenet.net).

Szaló István (ratiosoft@freemail.hu) tanár, immár több mint másfél évtizede foglalkozik programozással, de csak a Java és a Linux megismerése után tudta meg, hogy mi is az igazi programozás. Azóta hátat fordított a feketedoboz módszereknek, és most szabadidejében a nyílt forráskódú Java és a GNU C vagy C++ programokat tanulmányozza. Több írása megjelent már a hazai számítástechnikai lapokban. Ha néha feláll számítógépe mellől, rendszerint művészettörténész feleségével és kisiskolás lányával „találja szemben" magát.



Bemutatkozik az Enhydra

Ügyes segédlet arról, hogyan írjunk servleteket és alapalkalmazásokat az Enhydrával érkező eszközök segítségével.

Cikkünkben azt fogjuk megvizsgálni, hogyan kell servleteket és alapalkalmazásokat készíteni az Enhydrával érkező eszközökkel. Bár az Enhydra webalkalmazásai annyira sem szabványosak, mint a Jakarta-Tomcat servletei, azért jókora erőt képviselnek, és lehetőséget adnak arra, hogy teljesen nyílt forráskódú környezetben használjunk Enterprise JavaBeanst.

Történet és háttér

Az Enhydra Javában írt nyílt forráskódú alkalmazáskiszolgáló, melynek legfőbb célja a Sun J2EE meghatározásainak minél pontosabb betartása. Az Enhydra-fejlesztés éllovasa, a Lutris cég BSD-szerű szabadalom alá helyezte az alkalmazáskiszolgálót. A jelenlegi üzembiztos Enhydra-változat (a 3.5-ös) nagy mennyiségű szabványt támogat, beleértve a servleteket és a JSP-ket is. Az Enhydra Enterprise, melynek tervezett kiadási ideje 2001 nyara volt (lapunk kiadásakor még nem érhető el – a szerk.), további J2EE-képességekkel rendelkezik majd, beleértve az Enterprise Java Beans (EJB) támogatását is. A teljes Enhydra nyílt forráskódú termékként hozzáférhető, azaz letölthetjük a Hálóról, és nyugodtan telepíthetjük. Az olyan ügyfelek számára, akik gyanakvással tekintenek a nyílt forráskódú programokra – tehát szívesebben használnának előrecsomagolt terméket, amelynek minőségét ellenőrizték, továbbá támogatást szeretnének kapni a Lutristól –, az Enhydra kereskedelmi változatai is megvásárolhatók. Természetesen gyanítom, hogy a legtöbb embernek, aki e sorokat olvassa, nemigen van szüksége a Lutris támogatására, de azért jó tudni, hogy készek és hajlandók segíteni másokat a termék használatában.

A Lutris nemcsak a J2EE-piacot célozta meg, hanem a vezeték nélküli internetalkalmazásokét is. Még nem találom túl elterjedtnek a mobil internettechnológia használatát – a mobiltelefonom WAP-szolgáltatása jóindulattal is csak szánalmasnak nevezhető –, de a Lutris ennek az idővel elkerülhetetlenül fontos piacnak egyik játékosaként tekint az Enhydrára.

Bár az Enhydrának van még mit fejlődnie a Zope vagy az ArsDigita Community Systemhez képest a mindshare és a közösség terén, már így is meglehetősen nagyszámú piaci sikert könyvelhet el magának. Például a Hewlett-Packard mostanában jelentette be, hogy a továbbiakban együttműködik a Lutrisszal, és számos alkalmazásban Enhydrát fog használni. Ez ékesen bizonyítja, hogy a nyílt forrású alkalmazáskiszolgálók igenis fontos helyet töltenek be a webes alkalmazásfejlesztés világában, és még olyan cégek esetében is szóba jöhetnek, amelyek egyébként sokkal többet is képesek lennének fizetni értük.

Kezdjünk hozzá!

Most, hogy egy kicsit feltártuk a háttérben zajló folyamatokat, próbáljunk meg az Enhydra segítségével egyszerű alkalmazásokat készíteni. Első feladatunk természetesen a termék letöltése és telepítése lesz. A magam részéről az Enhydra Enterprise próbaváltozatával dolgoztam, mivel EJB-képességeit szerettem volna kipróbálni. A próbaváltozat az Enhydra Enterprise JDK 1.3-ra támaszkodik, ami kellemes újdonságnak tekinthető az előző változatokhoz képest, amelyek a JDK 1.2-vel működtek annak ellenére, hogy az 1.3 már jó ideje elkészült.

Letöltöttem és kicsomagoltam az Enhydra tarfájlt, amely számos fájlt és könyvtárat hozott létre az enhydra4.0 könyvtárban. A leírást szokás szerint a doc könyvtár tartalmazza, a lib könyvtárban találhatók azok a .JAR-fájlok, amelyek az Enhydra számára szükségesek, végül pedig a conf könyvtár tartalmazza az Enhydra általános beállítófájljait. Itt helyezkedik el továbbá egy bin könyvtár is, amelyben csaknem kizárólag parancsfájlok rejlenek (a windowsos .BAT megfelelőjükkel), ezek hajtják végre az Enhydrát beállító különféle Java-programokat.

Futtassuk le a `configure` parancsfájlt az Enhydra-terjesztés főkönyvtárában (melyre a következőkben $ENHYDRA-ként fogok hivatkozni), hogy biztosak lehessünk benne: az Enhydra-parancsfájlok és Javaprogramok figyelembe veszik azt a könyvtárat, ahová az Enhydrát telepítettük. A `configure` egyetlen kötelező értéket vár: a JDK-telepítésünk könyvtárát. A `configure` számos Makefile-t és más beállításfájlt módosít, de semmilyen kód újrafordítását nem kényszeríti ki. A `configure` lefuttatása után (amely semmilyen látható kimenetet nem fog adni) az $ENHYDRA könyvtárban a `bash` parancsfájlt (`setup.bash`) ugyancsak futtassuk le, ez a JDK végrehajtható állományokat tartalmazó könyvtárát adja hozzá a PATH-hoz.

Az Enhydra számos különféle, egymással összekapcsolódó programcsomagot tartalmaz. Maga az alkalmazáskiszolgáló (más néven multiserver) képes közvetlenül a szemközti HTTP-ügyfelekkel dolgozni, vagy akár egy olyan proxyként működő webkiszolgálóval együttműködni, mint az Apache. Ha a `loadOrder` tulajdonságot az $ENHYDRA/conf/bootstrap.conf fájlban átírjuk, csökkenthetjük az alkalmazáskiszolgáló által elindított szolgáltatások számát, vagy megváltoztathatjuk az indítási sorrendjüket.

A kiszolgáló indításához egyszerűen csak futtassuk le az `$ENHYDRA/bin/multiserver` parancsot. A program indulásakor a szolgáltatásokat egymás után indítja, míg végül a következő üzenetet láthatjuk: „Bootstrapper initialized normally”, azaz a betöltő parancsfájl hiba nélkül lefutott. Ezen a ponton kipróbálhatjuk a kiszolgálót: ha a böngészőt a 8001-es kapura irányítjuk, egy irányítópultot hoz fel, amelyen megfigyelhetjük a multiserver pillanatnyi állapotát.

Az indításra tett első kísérleteim kudarcot vallottak, mivel a program mindig arról panaszkodott, hogy nem találja az enhydra.jar fájlt a CLASSPATH-ban. Ez azért volt különösen zavaró, mert az egész Enhydra-terjesztésben semmiféle enhydra.jar nevű fájlnak nem akadtam a nyomára.

A megoldás, mint kiderült, meglehetősen egyszerű, ha nem is nyilvánvaló: az Enhydra tudja, hogy milyen CLASSPATH szükséges az egyes programok futtatásához, de ezeket a beállításokat figyelmen kívül hagyja, ha a CLASSPATH-t már eleve beállítottuk. Ezért a CLASSPATH-t még a kiszolgáló futtatása előtt töröljük, eltávolítva ezáltal a környezeti változót. Ha megtettük, a multiserver a várakozásnak megfelelően fog elindulni.

Egyszerű webalkalmazás készítése

Az elmúlt pár hónapban bepillantást nyerhettünk a Java-servletek és a JavaServer Pages világába. Az Enhydra mint J2EE-megfelelő

1. lista Foo.java

```
/*
 * Egyszerű servlet, amely azt mutatja be,
 * hogyan lehet azokat az Enhydrába illeszteni.
 * Ez a servlet a /foo URL alatt lesz elérhető.
 */

package atf.presentation;

// Servlet import
import javax.servlet.*;
import javax.servlet.http.*;

// Standard import
import java.io.*;
import java.text.*;
import java.util.*;

public class Foo extends HttpServlet {


   public void doGet(HttpServletRequest request,
     HttpServletResponse response)
       throws ServletException, IOException
   {
       // kimeneti folyam a STDOUT-ra
       PrintWriter out = response.getWriter();

       // kiírunk némi HTML-t
       out.println("<HTML><Head><Title>");
       out.println("Foo</Title></Head>");
       out.println("<Body>");
       out.println("<P>Foo</P>");
       out.println("</Body>");
       out.println("</HTML>");
   }
}
```

alkalmazáskiszolgáló ezeket a módszereket teljes mértékben támogatja. Ráadásul nyílt forrású kiszolgáló lévén az Enhydra a servlet és a JSP-motor alapjául a Jakarta-Tomcat-motort használja. Amint azt nemsokára látni fogjuk, lehetőség nyílik arra (sőt, gyakran előnyösebb), hogy az Enhydra saját fejlett webalkalmazásait használjuk.
Az Enhydrával érkező eszközökkel viszonylag könnyű servleteket készíteni. Az Enhydra alkotói minden bizonnyal sok időt szenteltek egy olyan rendszer megalkotásának, amely nemcsak telepítve erős, hanem a fejlesztés alatt is viszonylag könnyen kezelhető.
Ha valaki általában csak servleteket szokott írni, fordítani és könyvtárba helyezni, tapasztalni fogja, hogy az eljárást megnehezítvén az Enhydra csak útban van. Ez nem kis részben annak a módnak „köszönhető”, amely szerint az Enhydra-alkalmazásokat telepíteni kell: külső kiszolgáló igénybevétele nélkül, ugyanis az Enhydra arra számít, hogy a legtöbb webalkalmazást a többitől függetlenül szeretnénk kipróbálni (és futtatni).
Egyszerű servlethez az Enhydra részét képező alkalmazáskészítő varázslót fogjuk használni (`appwizard`), mely az `$ENHYDRA/bin/appwizard` paranccsal hívható meg. Az `appwizard` ugyan nem IDE, de elég kifinomult fájlmásoló program, amely olyan egyszerű csontvázalkalmazásokat képes készíteni, melyek már önmagukban működnek.
Az `appwizard` első futtatásakor megkérdezi, hogy webalkalmazásokat (például egyszerű servlet) vagy Enhydra-szuperservleteket szeretnénk-e fejleszteni. Válasszuk az egyszerű (standard) webalkalmazást. (A szuperservletekről később még szót ejtünk.) A következő képernyő arra kérdez rá, hogy HTML-kimenetet szeretnénk-e vagy WML-t, ez utóbbi lassanként a mobil internetalkalmazások XML-alapú szabványává válik. A HTML-t fogjuk használni, és a projektkönyvtárat, valamint a csomagot is „atf”-nek nevezzük. Alapértelmezés szerint az Enhydra-alkalmazások a saját könyvtárunkba, az enhydraApps alkönytár alá kerülnek. Válasszunk a kiadott kódhoz szabadalomtípust, és az `appwizard` máris elkészíti új alkalmazásunkhoz a fájlokat.
Az `appwizard` nagyszámú önműködően készített fájlt és könyvtárat hoz létre, többek között:

- Egy általános Makefile-t, amely lehetővé teszi, hogy az alkalmazást lefordítsuk. Az egyes alkalmazások alkönyvtáraiban szintén egyedi Makefile-okat találunk.
- A config.mk-t, ez számos környezeti változót állít be, amelyek a Makefile futását befolyásolják. Olyan adatokat találhatunk itt, mint az Enhydra-változat, a JDK-telepítés és az Enhydra-telepítés könyvtárát.
- Az src könyvtár a Java-servletekhez és a HTML-fájlokhoz tartozó kódot tartalmazza. Az src-n belül egy általános WEB-INF könyvtárra bukkanhatunk, amelyben web.xml fájl nevez meg minden egyes telepítendő servletet.
- Az atf könyvtárat, melynek neve a készülő projekt nevétől függ. Négy alkönyvtárat tartalmaz: business, data, presentation és resources. Minket most leginkább a presentation és a resources könyvtár érdekel, hiszen az első tartalmazza a servleteket, a második pedig a HTML-fájlokat és a JSP-ket.

Az alkalmazás felépítéséhez futassuk le a `make`-et a projekt gyökérkönyvtárában. (Az Enhydra Enterprise leírásában kiemelték, hogy a `make` helyett immár Java-alapú Ant-építőeszközt alkalmaztak, de úgy tűnik, az alkalmazáskészítés azért továbbra is a `make`-en alapul.)
Miután a `make` befejezte munkáját, alkalmazásunk legfelső szintjén, az src és az input mellett új kimeneti alkönyvtárat kell találnunk. A kimenő könyvtár mindent tartalmaz, ami az alkalmazás elindításához szükséges lehet, beleértve a .CLASS fájljainkat tartalmazó normál Java .WAR (web archivum) fájlt, XML-meghatározókat, JSP-ket és képeket:

```
WEB-INF/classes/atf/presentation/WelcomeHTML.class
WEB-INF/classes/atf/presentation/WelcomeServlet.class
WEB-INF/classes/atf/presentation/RedirectServlet.class
media/Enhydra.gif
index.jsp
WEB-INF/web.xml
```

Figyeljük meg, hogy itt már három .CLASS fájlunk létezik, holott az src/atf/presentation/ csak kettőt tartalmazott. Ez azért történik így, mert az XMLC a src/resources könyvtár HTML-fájlját Java-forrásfájllá alakította, mely végezetül Java .CLASS fájllá alakult át.
Így mindössze két parancs (az `appwizard` és a `make`) segítségével képesek voltunk egy teljes, működőképes Enhydra-alkalmazás elkészítésére. Az alkalmazás jelen állapotában semmi különöset vagy érdekeset nem végez, de kitűnő vázat ad, amely módosítható és kibővíthető.
Alkalmazásunkat az `output/start4` segítségével futtathatjuk.

Ez a 9000-es kapun indítja el az alkalmazást (a kapu száma az input/conf/servlet/servlet.conf.in fájlban adható meg). Ha a böngészőnket ekkor a http://localhost:9000-re irányítjuk, a servlet kimenetét fogjuk látni, azaz az Enhydra-logót, alkalmazásunk nevét (atf), a pillanatnyi időt és a dátumot, valamint egy hivatkozást, amely az alkalmazásra visz bennünket vissza.
A HTML-lap az XMLC segítségével készült, és jól szemlélteti, miként épül be az XMLC az Enhydra legtöbb alkalmazásába.
Az XMLC a src/atf/resources/Welcome.html fájlt Java-servletté fordítja, mely ezután .CLASS fájllá alakul. Az XMLC által létrehozott Java-osztály minden egyes `<span>`-taghoz egy horgot (hook) készít, így a `<span>`-tagok között bármit módosíthatunk, amelynek ID-tulajdonságot mutat fel.
A WelcomeServlet, az alkalmazásunk elején végrehajtott servlet a pillanatnyi időt és a dátumot úgy jeleníti meg, hogy egy példányt állít elő az XMLC által készített osztályokból:

```
now = DateFormat.getTimeInstance(DateFormat.MEDIUM).
                ➥format(new Date());
welcome = new WelcomeHTML();
welcome.getElementTime().getFirstChild().setNodeV
                ➥alue(now);
```

Másképpen fogalmazva: a `<span>`-tagok közti ID-vel azonosított szöveget azáltal helyettesítjük az idővel, hogy a HTML-fájlt DOM által elérhető fává alakítottuk, majd egy adott csomópont értékét megváltoztattuk.
Alkalmazásunkhoz további servleteket adhatunk, ha a src/atf/presentation könyvtárba másoljuk, vagy már eleve itt hozzuk létre őket. Fontos tudni, hogy a csomag neve nem egyszerűen atf, hanem atf.presentation lesz.
Ezekután egy nagyon egyszerű osztályt fogunk írni, melynek a Foo nevet adtuk, és az *1. listában* látható. A csomag nevét kivéve semmiféle különbség nem látható a hagyományos servletek és a Foo.java közt.
A servletmotornak meg kell mondanunk, hogy servletünkhöz egy URL-t rendeljen, amit legegyszerűbben az src/WEB-INF/web.xml fájlban tehetünk meg. Ez az XML-állomány két részre osztható: az első rész servletosztályokat rendel servletnevekhez, a második servletneveket URL-ekhez. A *2. listán* a Foo-servletünk kezeléséhez igazított web.xml fájlt láthatjuk.
Végül az új osztályunkat a Makefile-ba be kell vinnünk, amit úgy tehetünk meg, hogy osztályunk nevét hozzáadjuk a CLASSES változóhoz:

```
CLASSES = WelcomeServlet \
          ➥RedirectServlet Foo
```

Futtassuk a `make`-et és ellenőrizzük, hogy a Foo.class valóban hozzáadódott-e az alkalmazáshoz:

```
jar tvf output/archive/atf.war
```

Ha működik, futassuk az `output/start4-et`, és a böngészővel mutassunk http://localhost:9000/foo címre. Most az új Foo servletünk által kiküldött HTML-kimenetet kell látnunk.

Enhydra-alkalmazás készítése

Tapasztalt webfejlesztők nyelvtől vagy környezettől függetlenül rendszerint minden egyes honlaphoz külön programot szoktak írni. Ha öt különböző dinamikusan létrehozott lapot szeretnénk megjeleníteni, akkor öt külön CGI-programot, mod_perl-kezelőt, servletet vagy JSP-lapot kell készítenünk.

2. lista A web.xml módosított változata

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<web-app>
    <servlet>
        <servlet-name>
            welcome
        </servlet-name>
        <servlet-class>
            atf.presentation.WelcomeServlet
        </servlet-class>
     </servlet>
     <servlet>
        <servlet-name>
            redirect
        </servlet-name>
        <servlet-class>
            atf.presentation.RedirectServlet
        </servlet-class>
    </servlet>
     <servlet>
        <servlet-name>
            foo
        </servlet-name>
        <servlet-class>
            atf.presentation.Foo
        </servlet-class>
    </servlet>


    <servlet-mapping>
        <servlet-name>
            welcome
        </servlet-name>
        <url-pattern>
            /welcome
        </url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>
            redirect
        </servlet-name>
        <url-pattern>
            /redirect
        </url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>
            foo
        </servlet-name>
        <url-pattern>
            /foo
        </url-pattern>
    </servlet-mapping>
</web-app>
```

Az Enhydra lehetővé teszi, hogy eltérjünk ettől a modelltől egyedi lapok helyett alkalmazásszinten gondolkodva. Ehhez nyújtanak segítséget a szuperservlet néven ismert alkalmazások, ahol egyetlen alkalmazásobjektumot rendelünk számos megjelenítő felülethez.

A megjelenítő objektumokat az Enhydra URL-ben könnyen azonosíthatjuk: a .po utótag árulja el az Enhydrának, hogy az URL-ben megnevezett objektumot kell meghívnia. Így az Abc.po kérelem az Abc nevű megjelenítő objektumhoz tartozó `run()` eljárást indítja el.
A hagyományos Java servletekkel ellentétben a megjelenítő objektumok minden HTTP-kérelem esetén újraértelmeződnek. Lehet, hogy ez valamivel kevésbé hatékony, mintha egyetlen servletpéldányhoz több szálat használnánk, viszont megkímél minket a szálbiztos kód írásának fáradalmaitól.
Egy egyszerű Enhydra-alkalmazás tehát legalább egy alkalmazásobjektumot vagy megjelenítő objektumot tartalmaz. Ezek a PO-k kapcsolják össze az Enhydra másik két fő objektumtípusát: az üzleti (business) objektumokat (ezek tartalmazzák a gyakran használt szolgáltatásokat) és az adatobjektumokat (ezek állandó tárolók, például relációs adatbázist rendelnek Java-osztályokhoz). Amint már korábban is említettük, az alkalmazás src könyvtárában mindhárom objektumtípus – presentation, business és data – saját könyvtárral rendelkezik. Továbbá az összes ilyen objektum a háromrétegű webalkalmazás egy-egy alapvető rétegét képezi. Ezért – bár beletelhet egy kis idő, amíg megszokjuk a három objektumtípus elkülönítését – ez a modell egyre inkább terjed a webalkalmazások körében.
Akárcsak az imént, az alkalmazás vázának elkészítéséhez az Enhydra `appwizard`-ját használjuk, amit azután megváltoztatunk. Futassuk le újra az `appwizard`-ot és az első képernyő listájából az egyszerű webalkalmazás helyett válasszuk a *super servlet* lehetőséget! Én a projektet egyszerűen myprojectnek neveztem el és az il.co.lerner csomagba helyeztem – cégünknél általában ezt használom a belső projektekhez. Az `appwizard` ezután az alkalmazás vázát az ~/enhydraApps/myproject könyvtárban hozza létre. Az alkalmazás hasonló szerkezetű, mint a servletünk, és hasonló könyvtárszerkezettel rendelkezik. Az src/il/co/lerner alatt egy presentation, egy data és egy business könyvtárunk lesz. Akárcsak az imént, most is létezik egy legfelsőbb szintű Makefile, amely lefordítja és elkészíti szuperservletünket.
Nézzünk bele a presentation/WelcomePresentation.java fájlba, ahol a tulajdonképpeni megjelenítést végző megjelenítő objektum forráskódját találjuk. Amennyiben a legfelsőbb szintű könyvtárba begépeljük a `make` parancsot, az alkalmazás indításához lefuttatjuk az `output/start4` parancsot, majd a webböngészőt a

3. lista WelcomePresentation.run()

```
public void run(HttpPresentationComms comms)
   throws HttpPresentationException, IOException {

   WelcomeHTML welcome;
   String now;

   welcome = (WelcomeHTML)comms.xmlcFactory.create
             ➥(WelcomeHTML.class);
   now = DateFormat.getTimeInstance
         ➥(DateFormat.MEDIUM).format(new Date());
   welcome.setTextTime(now);
   comms.response.writeDOM(welcome);
}
```

➲ http://localhost:9000 címre irányítjuk, azt látjuk, hogy böngészőnk átirányítódik a http://localhost:9000/WelcomePresentation.po címre. Ez a lap ugyanazt a kimenetet adja, mint amit a vázservletünk írt ki, azaz az Enhydra-logót, a pillanatnyi időt és a dátumot láthatjuk.
A po utótag – mint már ismerhetjük – utasítja az Enhydrát, hogy hívja meg a `WelcomePresentation` `run()` eljárást.
Az önműködően létrejövő vázalkalmazás, a `WelcomePresentation.run()` valahogy úgy fog kinézni, ahogyan azt a *3. listában* megfigyelhetjük.
A szuperservlet csatolófelülete hasonló a hagyományos servletéhez, így a servleteket ismerő programozónak nem kerül sok fáradságba az elsajátítása. A `run()` eljárás egyetlen HttpPresentationComms típusú értéket fogad, amely megjelenítő objektumunkat teszi elérhetővé mindazzal a kapcsolattartási lehetőséggel, amire a külső világ eléréséhez szükség van, ideértve a HTTP-kérelmeket és válaszobjektumokat is.
A `run()` eljárás a kimenet megjelenítéséhez elkészíti a WelcomeHTML egy példányát. Ez az a Java-osztály, amit az XMLC a Welcome.HTML-ből készített. Ezt követően a `run()`a „time” ID-vel jelölt `<span>`-tagok tartalmát a pillanatnyi dátummal és idővel helyettesíti. Ezekután írjuk a HTTP-válaszobjektumba az üdvözlés tartalmát, amelyet a DOM-fa tartalmaz.
Elkészíthetjük saját megjelenítő objektumunkat, a FooPresentationst, ahogy a *4. listában* láthatjuk. Ne felejtsük el az új objektumot a presentation könyvtár Makefile-jának CLASSES sorához hozzáadni! Amikor a legfelsőbb szintű alkalmazáskönyvtárból a `make`-et újrafuttatjuk, a FooPresentation lefordítódik és bekerül Enhydra-alkalmazásunkba.
Nagyon szép, hogy saját megjelenítő objektumot tudunk írni, de hol marad az alkalmazásobjektum, amely irányítja? A fő forráskönyvtárban ugyanazon a szinten, ahol a presentation, a data és a business könyvtárakat találhatjuk, létezik egy, a projekt nevével azonos nevű Java-osztályfájl; a mi esetünkben tehát egy src/il/co/lerner/myproject.java fájlt kell keresnünk.

Szuperservlet telepítése

Szuperservletünket önmagában mostanáig az átfogó alkalmazáskiszolgálón kívül futtattuk. Ez a képesség azon fejlesztők számára nagyszerű, akik a fő (termelő) weblap zavarása nélkül szeretnék a programokat kipróbálni, de alkalmazásukat végül a kiszolgálóba építik.
Az Enhydra segítségével viszonylag egyszerűen elvégezhető ez a feladat is: az alkalmazásunkkal kapcsolatos adatokat a kiszolgáló beállítófájljához adjuk, így az megtalálja az alkalmazást. Ezután alkalmazásunkat az irányítópanel segítségével az alkalmazáskiszolgálón egy tetszés szerinti URL alá helyezzük, ezáltal mindenki

4. lista FooPresentation.java

```
/*
 *  myproject
 *
 *  Enhydra szuperservlet megjelenítő objektum
 *
 */

package il.co.lerner.presentation;

// Enhydra Szuperservlet importok
import com.lutris.appserver.server.httpPresentation
➥.HttpPresentation;
import com.lutris.appserver.server.httpPresentation
➥.HttpPresentationComms;
import com.lutris.appserver.server.httpPresentation
➥.HttpPresentationException;


// Standard imports
import java.io.IOException;
import java.util.Date;
import java.text.DateFormat;

public class FooPresentation implements
➥HttpPresentation {


   public void run(HttpPresentationComms comms)
     throws HttpPresentationException,
     ➥IOException {

     // A tartalomtípus célja a felhasználó
     // böngészője

comms.response.setContentType("text/html");


     // Írjunk ki némi HTML-t
     comms.response.writeHTML("<HTML>");
     comms.response.writeHTML("<Head>
     ➥<Title>Foo</Title></Head>");
     comms.response.writeHTML("<Body>");
     comms.response.writeHTML("<P>Foo</P>");
     comms.response.writeHTML("</Body>");
     comms.response.writeHTML("</HTML>");
   }

}
```

számára elérhetővé válik.
Ennek kivitelezéséhez az alkalmazáskiszolgálót még egyszer újra kell indítanunk az `$ENHYDRA/bin/multiserver` parancsfájl segítségével. Ezután a `multiserver` a 8001-es kapun érhető el. Töltsük be a rendszergazdai képernyőt a böngészőbe, és nézzük meg a rendelkezésre álló alkalmazások listáját a bal felső sarokban. Másoljuk az output/conf/myproject.conf nevű alkalmazásbeállító fájlt – de ne a teljes alkalmazást – az $ENHYDRA/apps alá. Ha ezt megtettük, szerkesszük át az $ENHYDRA/apps/myproject.conf fájlt, megváltoztatva a `Server.ClassPath[]` értékét. Két lehetséges `Server.ClassPath[]` érték létezik eleve a myproject.conf fájlban: egy az alkalmazás önálló módú futtatásához, egy másik pedig a `multiserver` alatti működéshez. Tegyük megjegyzésbe az önálló futásértéket (az első), és szedjük ki a megjegyzésből a második értéket.
Most vissza kell térnünk a böngészőhöz, és az irányítópanelén kattintsunk az *Add* gombra (amelyiken egy nagy + jel látható). Egy új alkalmazást szeretnénk beilleszteni, nevének (myproject) már a listában kell lennie. Válasszunk alkalmazásunkhoz egy kezdő-URL-t és gépeljük be azt a szöveget, amit ehhez az alkalmazáscsoporthoz szeretnénk rendelni. Az alkalmazás beillesztéséhez kattintsunk az *OK* gombra, majd frissítsük az irányítópanelt. A bal felső sarokban a „myproject” szót kell látnunk a többi betöltött alkalmazással egyetemben. Ha a myproject névre kattintunk, a képernyő jobb alsó részén adatokat láthatunk a projektről.
Az alkalmazás futtatásához egy vagy több kapcsolatot hozzá kell rendelnünk, majd el kell indítanunk. Alkalmazásunk alapértelmezetten a 8002-es és a 8003-as számú kapun fut, amelyhez – ha akarjuk – egy vagy több kapcsolatot adhatunk hozzá. Ha a kapcsolatokat már meghatároztuk, kattintsunk a képernyő bal szélén található *run* gombra. A kapcsolat URL-ek hivatkozássá válnak, és rájuk kattintva a webalkalmazásunk egyik vagy másik kapcsolatát nyithatjuk meg (azaz az Enhydra-logót és a pillanatnyi időt), egy új ablakban megjelenítve. (A JavaScriptet és az új ablakokat általában bosszantónak találom, de az Enhydra fejlesztői – ami a szép webfelületet illeti – meglehetősen jól egyensúlyoznak ízlésesség és használhatóság között.)
FooPresentation objektumunkat az URL megváltoztatásával ki is

További érdekességek

Az Enhydrával való munka előfeltételezi a Java ismeretét, de természetesen nem árt némi servletekkel kapcsolatos háttértudás sem.

Az Enhydra viszonylag jól megalapozott termék, de a leírás megértése nem kevés időt vesz igénybe. Ráadásul az elérhető leírások jó része az Enhydra 2.x és 3.x változataihoz készült, így némileg már idejétmúlt. Mindenképpen javaslom az Enhydra-kézikönyv elolvasását: ➲ enterprise.enhydra.org/software/documentation/ee4b1/index.html (lásd a képen)

Az Enhydrát általánosságban tárgyaló, néhány XMLC-példával is megtűzdelt kitűnő cikk olvasható *Roger Metcalf* tollából, mely az ArsDigita Systems Journalban jelent meg, és a ➲ http://www.arsdigita.com/asj/enhydra címen érhető el.

Két egymást kiegészítő O'Reilly-könyv elég adatot szolgáltathat ahhoz, hogy elindulhassunk az XMLC témakörében. *Brett McLaughlin*, a Lutris munkatársának „Java and XML” című könyve nagy részletességgel szól a DOM-ról, összehasonlítva a SAX- és más XML-értelmező lehetőségekkel.

A Java Servlet Programming második kiadásában pedig *Jason Hunter* és *William Crawford* szentel egy teljes fejezetet az XMLC-nek, melyet kiszolgálóoldali Javát használó, hasonló dinamikus laplétrehozó rendszerek fejezetei követnek.

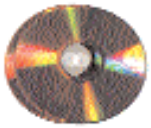
próbálhatjuk. Helyettesítsük a WelcomePresentation.po-t FooPresentation.po-ra, ekkor nagy valószínűséggel a Foo HTML-kimenetet találjuk a képernyőn.
Alkalmazásunkat egy vagy több kapuról kizárólag a webalapú irányítópanel használatával is eltávolíthatjuk, esetleg magából a kiszolgálóból. Végezetül a rendszert akár az irányítópanelen leállíthatjuk, vagy üssük le a CTRL+C-t abban a terminálablakban, ahol a multiservert elindítottuk.

Összegzés

Servleteket nem különösebben nehéz írni, de az Enhydra sokkal többet kínál ennél. Különlegessége, hogy olyan környezetet nyújt, ahol a servletek egyszerűen létrehozhatók, valamint teljes, összetett webkiszolgáló futtatása nélkül is kipróbálhatók. Ráadásul az Enhydra által nyújtott szuperservletekkel sokkal könnyebb dolgozni, mint a hagyományos servletekkel, különösképpen azért, mert sem szálproblémákkal, sem pedig az egyes lapokhoz új vezérlő írásával nem szükséges bajlódnunk.
Természetesen néhány hátulütője is akad a dolognak. Akárcsak a többi Java-program, az Enhydra CLASSPATH-beállításához is némi türelem szükséges. (Bár a Lutris javára kell írni, hogy a saját CLASSPATH-változóm eltávolítása az összes gondot megoldotta.) Igaz, az Enhydra önműködően létrehozott makefájljai óriási mértékben képesek csökkenteni a gondolkodási időt, amit egy kész alkalmazás létrehozásába bele kell fektetni, ám a Java-programok azonban még így is legalább tízszer annyi fájlból állnak, mint Perl- vagy Python-testvéreik.
Bár a szuperservletek nyilvánvalóan fejlettebbek „nem szuper” társaiknál, azért én még mindig habozom egy kicsit, mielőtt fejest ugranék egy olyan módszerbe, amely annyira eltér a régi, jól bevált és kitűnően leírt szabványtól – különösképpen most, amikor a nyílt forrás közössége egyre inkább az Enhydrára összpontosít. Végül, mivel az Enhydra Enterprise még nem került forgalomba, a telepítés és a leírás némi kívánnivalót hagy maga után.
Mindenezek ellenére könnyen megeshet, hogy a jövőben az Enhydra segítségével fogok Javában fejleszteni az egyszerű Jakarta-Tomcat helyett, amelyet ez idáig használtam. Az XMLC és az összevont fejlesztőkörnyezet kettőse több szempontból is meglehetősen vonzó.
Mint korábban említettem, az egyik ok, ami miatt az Enhydra Enterprise a leginkább felkeltette a figyelmemet, az, hogy képes kapcsolódni a Sun Enterprise JavaBeanshez. A következő két hónapban közelebbről is megvizsgáljuk az Enhydrát, elsőként az XMLC-vel, majd a DODS eszközzel ismerkedünk, amely relációs adatbázisokat rendel Java-objektumokhoz. Ezt követően egy kicsit az EJB világába is bekukkantunk, hogy szemléltessük: bár nyílt forráskódú programokra alapozunk, eszközeink semmivel sem lesznek kevésbé hatékonyak, mint az üzleti programozókéi.

Reuven M. Lerner (reuven@lerner.co.il) kisebb webes és internetes módszerekkel foglalkozó tanácsadó cég tulajdonosa és vezetője. A cikk megjelenésének időpontjában valószínűleg már végleg elkészült Core Perl című könyvével, melyet idén jelentet meg a Prentice-Hall. Az ATF honlapon érhető el (➲ http://www.lerner.co.il/atf/).

Naplófájl színezése



Gaelyne bemutatja, hogy az Apache httpd.conf fájljának megbűvölésével miként színezhetjük ki webes naplófájljainkat.

A weboldalak tárolásával foglalkozó cégeknek a legtöbbször életbevágóan fontos, hogy Apache-kiszolgálóik mit tesznek elérhetővé a világ számára, és ezt a lehető leggyorsabban szeretnék megtudni.

A rendszergazdának állandóan figyelnie kell a rendszer naplófájljait, ugyanígy a webes rendszergazdának is ellenőriznie kell a webes naplófájlokat. A valós idejű rendszerelemző programok nagy számát tekintve biztos voltam benne, hogy webes naplófájlok figyelésére több ilyen programot is találok majd. A ➲ http://freshmeat.net és más internetes lelőhelyek átbogarászása után megállapítottam, hogy igényeimet egyik program sem elégíti ki. Néhány ugyan éppen elérte a megfelelő színvonalat, a legtöbb azonban csak egy fájl figyelésére alkalmas; az a néhány pedig, amelyik több fájlt is elemez, olyan reménytelenül kezelhetetlen volt, hogy inkább más megoldás után néztem.

Végül nem programot használtam a feladatra, csupán végrehajtottam néhány módosítást az Apache httpd.conf fájljában. Az az ötletem támadt, hogy a gépekről származó adatokat egy „eldobható” naplófájlban összegyűjtöm, amit a `colortail` segítségével egy külső monitoron jelenítek meg. Így azonnal láthatom, melyik gép végez webes tevékenységet, honnan jön a forgalom, illetve éppen mely oldalakat hívják le. Sőt, ezáltal a parancsfájlokkal ügyeskedő srácokat és a nagy keresőmotorokat is megfelelően „kezelhetjük”. A rendszer olyan jól bevált, hogy később a rendszernaplózást is ennek használatával oldottuk meg.

A httpd.conf módosításai

Az általános naplózás formátuma (LogFormat) mellett egy „webmonitor” nevűt is létrehoztam:

1. lista A colortail.conf

```
# Az elérhető színek listája. Ezek bármelyikét
# használhatjuk, ha az alábbi formátumot betartjuk.
# COLOR magenta
# COLOR cyan
# COLOR green
# COLOR yellow
# COLOR brightred
# COLOR blue
# COLOR brightblue
# COLOR brightwhite

COLOR magenta
{
^.*(\[valami.com\]).*$
^.*(HEAD /).*$
}

COLOR cyan
{
^.*(\[masvalami.com.au\]).*$
^.*(GET /naplok/).*$
^.*(GET /konyvtar/).*$
^.*(GET /masikkonyvtar/).*$
}

COLOR brightyellow
{
# minden IP-címre illeszkedik
^.*([0-9]{3}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*$
^.*([0-9]{2}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*$
^.*([0-9]{1}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*$
# egy sorban két IP-címre illeszkedik
^.*([0-9]{3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]
➥{1,3}).*([0-9]{3}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*$
^.*([0-9]{2}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*([0-9]{2}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*$
^.*([0-9]{1}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*([0-9]{1}\.[0-9]{1,3}\.[0-9]{1,3}\.
➥[0-9]{1,3}).*$
}

COLOR brightred
{
# a root szóra illeszkedik
^.*(root).*$
# a DENY szóra illeszkedik
^.*(ppp-in DENY ppp0).*$
^.*(eth-in DENY eth0).*$
# Rendszernaplózás
^.*(authentication failure).*$
^.*(PAM_pwdb).*$
^.*(ftpd).*$
^.*(ssh).*$
^.*(ipop3d).*$
^.*(\[error\]).*$
^.*(LOGIN).*$
# CGI-BIN és PHP elemek
^.*(cgi-bin).*$
^.*(php).*$
^.*(php3).*$
}
```

2. lista A cron

```
#/bin/sh
#
# Napló-visszaállítás.
# Ez a fájl naponta egyszer fut le a cronból.
# Megforgatja a webmonitor_log fájlt és
# újraindítja a színes megjelenítést.

# PPP-kapcsolatunk a csatlakoztatott számítógép
# nevét "C128.valami.com"-ra állítja.

pty=`w | grep C128 | cut -c 10-15`

# Leállítja a colortailt, megforgatja
# a naplófájlt, újraindítja az Apache-és
# a rendszernaplózó démonokat.

/usr/bin/killall colortail;
cp /var/log/httpd/webmonitor_log
➥/var/log/httpd/webmonitor_log.old;
rm /var/log/httpd/webmonitor_log;
kill -1 `cat /var/run/httpd.pid`;
/usr/bin/killall -HUP syslogd;

# Ha a C128 csatlakoztatott, akkor
# az alábbi sorok a colortail kimenetét
# küldik át.

if [ "$pty" != "" ]
then
colortail -f -k
➥/etc/colortail.conf/var/log/httpd/
➥webmonitor_log> /dev/$pty &
fi
```

```
LogFormat "[%v] %h %u \"%r\" %>s %b\n\"%{Referrer}i
➥\" \"%{User-Agent}i\" %t" webmonitor
```

Ez a naplóadatot a hivatkozóval (Referrer) és a böngésző nevével (User-Agent) együtt egy második sorban jeleníti meg, megkönnyítve az olvasást. A naplófájl bármilyen formátumú lehet, akár a „megszokott” is, amilyet a hagyományos naplózáshoz használunk. Azért döntöttem a megváltoztatása mellett, mert az Apache elég rugalmas ahhoz, hogy megtehessük vele.

Mivel a GIF, JPEG és PNG grafikus fájlok a megjelenítést elronthatják, az alábbi sorokat illesztettem a *httpd.conf* fájl általános naplórészébe, amellyel kizártam e három fájltípust:

```
SetEnvIf Request_URI \.gif$ unwanted
SetEnvIf Request_URI \.jpg$ unwanted
SetEnvIf Request_URI \.png$ unwanted
```

Névalapú virtuális gépeket használtam, és mindegyikhez saját `<VirtualHost>` és `</VirtualHost>` tagok tartoznak. Állandó naplófájljaik mellett „webmonitor” fájlunkhoz minden egyes gépnél egy `CustomLog` parancsot adunk ki, például:

```
<VirtualHost valami.com>
...
CustomLog /var/log/httpd/someisp.com-access_log
➥combined
CustomLog /var/log/httpd/webmonitor_log
➥webmonitor env=!unwanted
...
</VirtualHost>
```

Bővítésem a következőképpen nézett ki:

```
CustomLog /var/log/httpd/webmonitor_log webmonitor
➥env=!unwanted
```

A `/var/log/httpd/webmonitor_log` a naplófájl elérési útvonala, amelyet az Apache azonnal el is készít, ha induláskor még nem létezett. Az egyéni formátumot használó naplófájl neve webmonitor, ezt a fenti `LogFormat` szakaszban határoztuk meg. Az `env=!unwanted` hatására a `SetEnvIf` sorokban megadott elemeket nem naplózza, így a grafikus fájlokra irányuló kérelmekről nem kapunk jelentést.

A fentiekben ábrázolt felügyeleti módszer olyannyira hasznosnak bizonyult, hogy a rendszernaplófájlokra is kiterjesztettük. Ehhez az alábbi sorokat illesztettük a `/etc/syslog.conf` fájlba:

```
kern.*;authpriv.*;*.crit;*.error;*.warning;*.emerg
➥/var/log/httpd/webmonitor_log*
```

A Colortail

A Colortail *Joakim Andersson* (➲ pt98jan@student.hk-r.se) programja, amit a ➲ http://www.student.hk-r.se/~pt98jan/colortail.html címről tölthetünk le, és a GNU felhasználási szerződésének feltételei érvényesek rá. A naplófájlok kiszínezése szebbé varázsolja a megjelenítést, továbbá óriási előnye, hogy a labor másik sarkából egyetlen pillantással meg tudjuk állapítani, éppen melyik gép bonyolít le forgalmat.

A Colortail mellett néhány példajellegű beállításfájlt is találunk, melyek egyike sem igazán felel meg a webes naplózáshoz (talán a *conf.xferlog* jó valamire). Némi bűvészkedés után az alábbiakban ismertetett formátumot alakítottuk ki. Ez egy keresztezés, mely webes és rendszerre vonatkozó események figyelésével egyaránt foglalkozik.

A színezés bekapcsolása

Ha a színezést helyileg szeretnénk bekapcsolni, akkor a

```
colortail -f -k /etc/colortail
➥/var/log/httpd/webmonitor_log &
```

parancsot használjuk. Egyetlen hátránya, hogy nem lehet állandóan a képernyőn, mindig a konzolról vagy az X-ablakról kell rá átváltani. A tevékenységek tökéletesebb megfigyeléséhez a színes kimenetet a rendszerhez kötött Commodore 128D típusú számítógépen jelenítjük meg. Az általunk használt felépítésben a C128 egy belső kiszolgálóra csatlakozik nullmodemmel és PPP-kapcsolattal. Innen lépünk be a naplófájlokat tartalmazó kiszolgálóra. Erre a célra bármilyen régi számítógép megfelel, amely képes ANSI vagy VT100-as megjelenítésre, valamint 80 oszlopos képernyővel rendelkezik. A PPP nem követelmény.

A `colortailt` nem a Commodore-ról indítjuk el, hanem az éjjelente lefutó cronra bízzuk a naplófájlok megforgatásának és a színes kimenet átküldésének feladatát. A *2. listán* az ezt végrehajtó fájl látható.

Összegzés

A naplófájlok figyelésének annyiféle módja létezik, mint égen a csillag, éppen ezért találtam érdekesnek ezt a feladatot. Bár a színezett naplófájlokban nincs semmi forradalmian új, még sehol nem láttam ilyesmit, és ez a megvalósítás tökéletesen megfelelt az igényeimnek. Remnyeim szerint ez a cikk felkeltette a webes események valós idejű megfigyelésére áhítozó rendszergazdák érdeklődését.

Gaelyne R. Gasson (gaelyne@videocam.net.au) webes rendszergazda Dél-Ausztráliában. A cikkben vázolt módszerrel egy szempillantás alatt meg tudja állapítani, honnan nézik webkaméráját ➲ http://gaelyne.com/webcam/.

Alkalmazásindítók

Marcel az ideális programindítókról álmodozik és néhány kedvencével ismertet meg bennünket.

Áh, François, sehogysem boldogulok ezzel a menüvel, de nem tudom levenni róla a szememet. Annyifelé elkalandoznak a gondolataim! E havi témánkul ugyanis az indítófelületeket választottam. François, te biztosan tudod, mit értek alkalmazásindítók alatt, például azt a hatalmas tappancsot a Gnome-munkaasztal bal alsó sarkában a tizenegyes asztalnál, vagy éppen az óriási K betűt a KDE-munkaasztalon a hatos asztalnál. És természetesen ott sorakoznak a kis kezelőgombok a KDE- és Gnome-menün!

Ám a fent említetteken túl az emberek már régóta kísérleteznek az alkalmazások más módon történő indításával. Például… Ó, végre megérkeztek régi ismerőseink! Isten hozott benneteket Chez Marcelnél! François és jómagam éppen az alkalmazásindítókról beszélgettünk… Üljetek le és helyezzétek magatokat kényelembe!

François, légy oly kedves, hozz egy kis bort, de sebtében! Az 1989-es évjáratú Elzászi Rizling kellemes nedű, igazán itt az ideje, hogy vendégeink is megízleljék.

Megmutattam François-nak a programindítókat a KDE-ben és Gnome-ban, de számos további munkaasztal-felület vár ránk a Világhálón is; a könnyebb elérhetőség kedvéért pedig ikonokat helyezhetünk el a munkaasztalon.

Ti, akik hozzám hasonlóan a parányi Window Makert élvezettel használjátok, ismerkedjetek meg a nevét meghazudtolóan kiváló egygombos alkalmazásindítóval. Ezt a WMbad névre hallgató helyes kis programindítót *Alexandre Beraud* és *Emmanuel Sunyer* készítette, és több olyan hasznos szolgáltatást nyújt, mint a tématámogatás vagy a görgetés, illetve az alkalmazások közti színátmenetes áttűnés. Csupán egy kattintás a nyílon és máris láthatjuk, amint az aprócska ábrák legördülnek egymásról.

Vajon mit kezdjünk mindezzel? Saját munkaasztalomról elárulhatom, hogy időnként meglehetősen túlzsúfolt. Így hát nem rossz ötlet egy olyan alkalmazásindító beszerzése, amely alig igényel helyet a munkaasztalon. Vessetek csak egy pillantást az *1. képre*, a munkába állított WMbadre!

Összeépítéséhez első lépésként gyűjtsük be a forráskódot tartalmazó tömörített állományt a ➲ http://perso.mnet.fr/esunyer/wmbadeng.html címről. A program telepítése egyszerű, bár néhány lépéssel tovább tart, mint amihez eddig hozzászokhattunk. A theme és pixmap könyvtárakat saját-könyvtárunkba kell bemásolnunk:

```
tar xzvf wmbad-0.3.0.tar.gz
cd wmbad-0.3.0
make
make install
mkdir $HOME/.bad
cp config.bad $HOME/.bad
cp alpha.xpm $HOME/.bad
cp -R pics $HOME/.bad
cp -R themes $HOME/.bad
```

1. kép Nem is rossz, igaz? A WMbad helyigénye igen kicsi

Most már csak az alkalmazás elindítása maradt hátra. Ehhez gépeljétek be: `wmbad &`. Minden, ami a WMbad testreszabásához szükséges, magából a programból elérhető. Szükséged lenne gyors xtermre? Kattints egyet a képernyő bal oldalán megjelenő tálcára, majd a jobb oldali kerek gombra, így a beállítófájlt (config) a szövegszerkesztőben szerkeszthetjük, majd frissíthetjük.

Remélem, ti is úgy fogjátok találni, hogy a beállítófájl formátumával könnyű dolgozni. Lássuk a példát! Fordítsunk különös figyelmet a „Commands number” (parancsok száma) szöveget tartalmazó sorra! Ha növelitek a meghatározott parancsok számát, ennek megfelelően ezt a számot is meg kell változtatni. Jómagam hét lapozóparancsot adtam meg, a „gyári” beállítófájl azonban csak négyet tartalmaz:

```
[Theme pixmap]
default.xpm
[NONE/SCROLL/STORE]
SCROLL
[Commands number]
7

[Com 1]
emacs
editor.xpm
```

Véletlenül tudomásomra jutott, hogy akadnak köztetek olyanok is, akik az AfterStepet használják. Menünk alábbi fogása pont nekik való. A következő sorok valójában mindenki számára hasznosak lehetnek, hiszen KDE-ben, Window Maker ablakkezelőből futtattam a következő programot, melynek neve `asbutton`, és *Ryan Lathouwers* keze munkáját dicséri. A program legfrissebb változata a ➲ http://home.pacbell.net/rzanlath/asbutton.html címről tölthető le. Csomagoljuk ki a forráskódot, majd végezzük el a program összeépítését!

```
tar xzvf asbutton-0.3.tar.gz
cd asbutton-0.3
make
make install
cp .asbuttonrc $HOME
```

Az utolsó parancs az alapértelmezett beállítófájlt a sajátkönyvtárunkba másolja.

2. kép Az asbutton program

Az állomány formátuma egyszerű és pontosan leírt, úgyhogy egyéni parancsokat hozzáadni sem lehet bonyolult feladat. És ha a kíváncsiság már csaknem megöl benneteket, hogy a programot az eredeti parancsgombokkal lássátok, indítsátok el az `asbutton &` parancs segítségével. Amennyiben – az alapértelmezés szerinti négygombossal szemben – a kilencgombos változatot kívánjátok futtatni, használjátok a `-n 9` kapcsolót.

A programindító által elfoglalt hely minél jobb kihasználása érdekében egy indítógombhoz több alkalmazást is rendelhetünk. Egy kattintás a bal egérgombbal, és máris indul a felső program, egy kattintás a jobb egérgombbal, és azonnal dolgozni kezd például az `xosview`. Ezt kilenc gombra kivetítve azt jelenti, hogy egyetlen egérkattintással 18 programhoz férhetünk hozzá, miközben csak csekély, 64×64 képpontos felületet használunk fel a képernyőből (*2. kép*)!

Ha ragaszkodtok ahhoz, hogy ez az indítási lehetőség megjelenjék az AfterStep ablakkezelőtökben, a következő sort szúrjátok be

a beállításokat leíró állományba:

```
*Wharf asbutton - Swallow
"asbutton" asbutton  &
```

Ha a szóban forgó állományt nem sikerült megtalálni, elárom a helyét:

```
$HOME/GNUstep/Library/AfterStep
/wharf/config
```

3. kép A minibar – bor sajnos nincs a kínálatban

Szeretnék egy másik programindítót is bemutatni nektek, mely a neve miatt különösen vonzó. Hittétek volna, hogy a minibar elnevezés valami ilyesmit takar? *Michael Eddington* minibarját megpillantva – bevallom – én először csalódottnak éreztem magam, ugyanis még egy üveg közönséges asztali bor sem található benne! Programindító létére azonban olyan apró, hogy szinte alig vesz el helyet a munkaasztalból. Ikonjai kisméretűek és egyetlen egérkattintással bármilyen parancsot elindíthatunk. „Előételként” töltsük le a ➲ http://www.phed.org/miniProject címről saját programpéldányukat.

Egy apró figyelmeztetést azért engedjetek meg… bár a világért sem szeretném, ha szerénykednétek a kóstolgatásakor! Vegyétek és fogyasszátok, amíg csak tetszik, barátaim! François, tölts még bort! Nos, amiről szólni akarnék, az, hogy a terjesztésben létezik egy előfordított bináris állomány, viszont nehézségekbe ütköztem a futásidejű könyvtárak kapcsán. A feladat megoldásának legegyszerűbb módja, ha a tömörített állományt kicsomagoljuk, eltávolítjuk a régi futtatható állományt, majd elvégezzük a program újbóli összeépítését:

```
tar xzvf minibar-0.05.tar.gz
cd minibar-0.05
make clean
make
make install
cp minibarrc $HOME/.minibarrc
```

Amint látjátok, az utolsó lépés a beállítófájl sajátkönyvtárunkba történő bemásolását foglalja magába. Mondom, ez egy közönséges állomány, mellyel egyszerű dolgozni. Minden nyomógomb és parancs kialakítása könnyű, ahogyan ez a *3. képen* is látható.

Ha például „billentyűt” szeretnék kialakítani, amelyre az egérrel rábökve elindíthatom a Gimpet, az állományba az alábbi sorokat szükséges beszúrnom:

```
/usr/include/X11/pixmaps/
                ➥mini-palette.xpm
The GIMP
gimp
```

A beszúrt sorok közül az első miniikonjaink elérési útvonalát tartalmazza (a telepítési folyamat jó néhány ikont bemásol a `/usr/include/X11/pixmaps` könyvtárba), de emellett más ikonokat is használhatunk, sőt, sajátokat is készíthetünk. Az alábbi példában egy 48×48-as png képet akartam Gimppel 16×16-os méretűvé átalakítani, végül pedig .XPM állományként elmenteni. Egy kis mágia az ImageMagickkel, és íme:

```
convert -geometry 16x16!
          ➥gnome-gimp.png gimp.xpm
```

Az átalakítóprogram az ImageMagick-eszközkészlet része, és valószínűleg már telepítetted Linux-rendszeredre. További érdekes parancsok várnak még felfedezésre, többek között a `mogrify` és `identify`. Abban az esetben, ha az ImageMagick nincs a rendszeredre telepítve, de ebben nem vagy teljesen biztos, telepítő CD-id segítségével ellenőrizheted, vagy keresd fel a Hálón a ➲ http://www.imagemagick.org helyet.

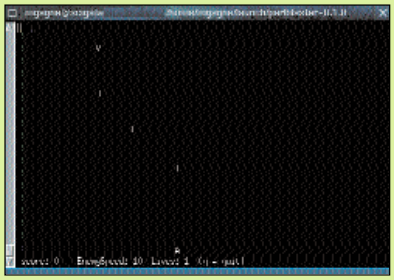
Az igazság az, hogy az ilyen alkalmazásindítók csak közönséges menük, igaz, némi körítéssel. Minthogy lényegük még mindig egyszerűségükben rejlik, a mai menühöz hozzáadnék egy utolsó fogást is, valami olyat, ami egy másfajta indítóval áll közeli rokonságban.

Hadd ajánljam a figyelmetekbe *David Slimp* programját, a Perl Defense Blastert! Ez a szerepjáték teljes egészében Perl nyelven íródott, és ASCII-képernyőn fut, igencsak szerény grafikával. Ennek ellenére meglehetősen nehéz abbahagyni – vagyis könnyen okoz játékfüggőséget! Saját példányotokat a ➲ http://perlblaster.sourceforge.net/ címről tölthetitek le. Néhány modul beszerzése végett szükséges lehet a CPAN-honlap felkeresése is: a Curses.pm és a TermReadKey.pm. FTP használatával pedig az ➲ ftp://ftp.cpan.org/CPAN/modules címről juthatunk hozzá a programokhoz. ➲ ftp://ftp.cpan.org/CPAN/modules/by-module/Curses és ➲ ftp://ftp.cpan.org/CPAN/modules/by-module/Term

Igaz ugyan, hogy befejeztem már a Perl-modulok telepítését, ennek ellenére hadd adjak erről is rövid összefoglalót. Példaként a Curses.pm modult fogjuk használni.

```
cd /usr/local/temp_dir
tar xzvf Curses-1.05.tar.gz
cd Curses-1.05
perl Makefile.PL
make
make install
```

Minthogy a Perl Defense Blaster (*4. kép*)



4. kép A Perl Defense Blaster, a harcos Perl-szerepjáték

nem több puszta héjprogramnál (egyúttal jó alkalmat teremt arra, hogy a Perl-kóddal kísérletezzünk!), nincs más teendőnk, mint a forráskódot kicsomagolni és futtatni:

```
tar xzvf perlblaster-
          ➥0.1.0.tar.gz
cd perlblaster-0.1.0
./perlblaster &
```

A számbillentyűzet négyes és hatos gombjait használhatjuk a balra, illetve a jobbra mozduláshoz, az ötöst pedig tüzeléshez. Sok sikert! Remélem, ínyetekre való volt a mai menü, és legközelebb ismét benéztek hozzánk. Egészségetekre!

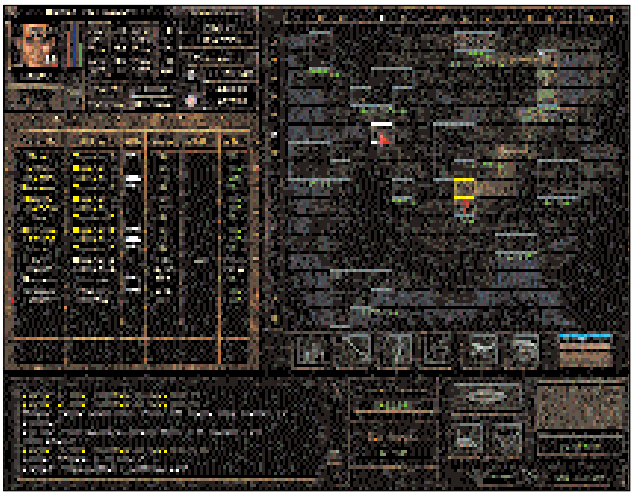
Marcel Gagné (mggagne@salmar.com) Mississaugaban (Ontario, Kanada) él, a Salmar Consulting Inc. rendszerépítéssel és hálózati tanácsadással foglalkozó cég elnöke. Pilóta és sci-fi író egy személyben. A világhálón elérhető honlapján sok hasznos dolgot találhatunk. ➲ http://www.salmar.com/marcel/

Kapcsolódó címek

asbutton ➲ http://home.pacbell.net/ryanlath/asbutton.html
CPAN Modules FTP Repository ➲ ftp://ftp.cpan.org/CPAN/modules/minibar
➲ http://www.phed.org/miniProject/
Perl Blaster ➲ http://perlblaster.sourceforge.net/
WINE Headquarters ➲ http://www.winehq.com/
WMbad ➲ http://perso.mnet.fr/esunyer/wmbadeng.html

Jagged Alliance 2 Linuxra

Szeretetteljes üdvözlet a Tribsoftnak! E bimbózó, linuxos játékokat gyártó vállalat bemutatkozó terméke a Windowsról átültetett Jagged Alliance 2, ami nagyszerű választásnak bizonyult. A JA2 eredetileg Sir-tech játék még 1999-ből. Zsoldos katonák seregét irányíthatod benne közvetlenül, hogy száműzött vezérük hatalmának visszaszerzésében segíts, és szülőföldjét megszabadítsátok a zsarnok uralkodó rabigájától. Ennek érdekében saját pénzügyi forrásaiddal kell gazdálkodnod, a megfelelő időben rátermett zsoldosokat kell fogadnod, illetve az alkalmatlanokat kirúgnod; továbbá a Te dolgod segédkezni a gerillacsapatok kiképzésében, valamint megszerezni a szükséges fegyverzetet és ellátmányt. A továbbiakban még olyan feladatok várnak rád, mint néped egész országának felszabadítása, és természetesen egyszer megküzdesz a rosszfiúkkal is.



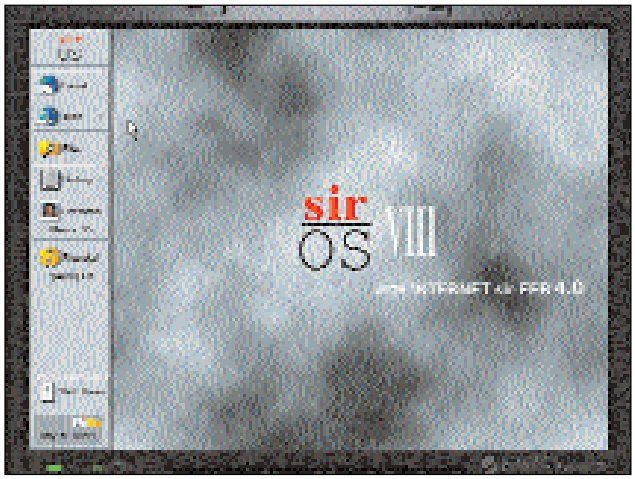
A játékot csupa elérendő céllal tűzdelték tele, ugyanis végigjátszása a számos beállítás miatt huzamosabb ideig eltarthat, és egyetlen tanulókör is olyan bonyolult, hogy még a lábaid is összegabalyodnak – főleg akkor, ha irtózol a kézikönyvek hosszas böngészésétől. A mára már klasszikussá érett játék mellett népszerűségén és összetettségén kívül még egy igen fontos érv szól: tényleg nagyon-nagyon szórakoztató!

A történet

A Jagged Alliance 2 voltaképpen a Jagged Alliance sorozat harmadik része (a második a *Jagged Alliance: Deadly Games*), és nem ez az utolsó darabja. A Tribsoft munkatársai fáradhatatlanul dolgoznak, hogy a közeljövőben a JA2-sorozat következő részével ajándékozzanak meg bennünket, a *Jagged Alliance 2: Unfinished Business*-szel.
A játék alapvetően az irányításod alatt álló zsoldosseregek tevékenysége körül forog, mivel szereped szerint a képzeletbeli arulco nép száműzött vezérének próbálsz segíteni. Deidranna királynő eltörölte a demokratikus kormányzatot, és hadserege erejének felhasználásával saját magát nevezte ki a csöppnyi ország legfőbb uralkodójává. Arulco elűzött vezetője, Enrico Chivaldori titokban felveszi veled a kapcsolatot: a játék kezdetén egy csendes kiskocsmában találkoztok, ahol megosztja veled szomorú történetét, és rengeteg pénzt kínál azért, hogy felszabadító háborújában a segítségére légy. Terve igen egyszerű: behatolsz az országba, visszaszerzed a városait, kiképzed az embereit, hogy meg tudják magukat védeni, végül visszahelyezed őt a hatalomba. Ha beleegyezel abba, hogy segítesz neki, a kapcsolattartáshoz szükséges hálózattal és a kezdéshez elegendő pénzzel is ellát (a beígért többivel majd később). Nos, természetesen beleegyezel (hiszen pénzéhes zsoldos vagy, és pénztárcája bankjegyekkel van kitömve), majd elkezdődik a kaland.
A belső történet voltaképpen nem egyenes vonalú; bár az ország visszaszerzése mint végső cél az egész játékot uralja, az is a feladataid közé tartozik, hogy Arulco népének segítséget nyújts – emiatt pedig gyakorta fognak kisebb-nagyobb, de mindig feladatok teljesítésével is megbízni. Ilyen például, hogy pénzjutalomért terroristát fogj el, titkos ügynököknek adj át üzeneteket, polgárőrséget képezz ki, valamint a felkelők csapatai számára tisztítsd meg az utánpótlási útvonalakat – ilyesmikkel a JA2 minden fordulójában találkozhatsz. A térkép mutatta világ maga annyira összetett és részletes, hogy időről-időre nagyobb célokba rejtett kisebb küldetésekbe is belebotolhatsz. Például az ország egyik körzetének ellenséges erőktől való megtisztítása közben, ha figyelmesen belepillantasz az épületekbe, egy pici, ingeket készítő gyárat fedezhetsz fel. Ha még közelebbről megnézed, mi folyik ott, észreveheted, hogy gyermekeket dolgoztatnak, és nemes szándéktól vezéreltetve kiszabadíthatod őket. A játék alatt kapcsolatba léphetsz az NPC-kkel („nem játékos szereplő” – több mint 150 van belőlük); akik valójában csak adatforrások, saját játékod történetének egyedi fejlődéséhez azonban nagyon is sokat tehetnek hozzá. A JA2 történetének szabadon formálható természetéből fakad a játék talán egyetlen hibája: lehetőség nyílik az elnyújtott, rétestészta-típusú játékidőkre, azaz rövid JA2-játékmenet nem létezik.

A játék menete

Némi fejtörést okozhat a Jagged Alliance 2 besorolása. Szerepjáték? Nem, akarom mondani: mégis, igen. Stratégiai játék? Néha. Valós idejű vagy körökre osztott? Mindkettő, vagyis attól függ… Szemmel láthatóan a JA2-t nehéz beskatulyázni. A Sir-tech a JA2-ben a stratégiai, a szerepjátékok, a valós idejű taktikai, valamint a körökre osztott játékok legjobb tulajdonságait egyesítette, okosan egyensúlyozva az egyes összetevők között, és rendkívül gördülékeny játékmenetet teremtve. Igazán mesterré válni azonban meglehetősen nagy kihívás. A JA2 vezérlőfelületének változatossága és összetettsége a SimCityt juttathatja eszünkbe, mindamellett azt hiszem, a legtöbb játékos egyetért velem a tekintetben, hogy ha egyszer a játék alapelemeinek mesterévé válnak, és a kezelőfelület legapróbb részleteit is kitapasztalják, a vezérlés eme szintje nemcsak szükséges a szerepjáték kombinációihoz, a taktikák és stratégiák olajozott működtetéséhez, de a játék rendszerében voltaképpen ez nyújtja a rugalmasság és alkalmazhatóság kényelmes fokozatát.
A játék a zsoldosoknak járó hordozható számítógépeden kezdődik. A Sir-tech kifejlesztette sirOS VIII-nak nevezett saját, faux operációs rendszerrel bíró felületét (kicsit mintha a Windowsra hajazna), amellyel Arulco mély dzsungeleiből az Interneten keresztül tudsz a Sir-techhez kapcsolódni (nem árt tehát időben műholdas modemet beszerezni). Innen kezdeményezhető számos erőforrás kezelése is. Kérdéses sem lehet: laptopod nélkülözhetetlen ahhoz, hogy zsoldosként sikeres légy. Ennek segítségével elektronikus leveleket küldhetsz és fogadhatsz (sőt, nemkívánt levélszemetekhez (spam) jutsz,



1. kép Íme, Eniac-laptopod!

amelyek az ellenkezőjét állítják magukról!). Üzleti weboldalakat kereshetsz fel, az e-kereskedelemmel foglalkozó oldalakról (ezek közül némelyik fejlesztés alatt állhat) pedig fegyvereket és lőszert vásárolhatsz, valamint költségvetésed alakulását is nyomon követheted. Adatokat ugyancsak tárolhatsz rajta, például Arulco térképét, a kémek által szerzett katonai adatokat, sőt az eseménynaplót is át tudod böngészni, hogy vajon mikor mit is cselekedtél.

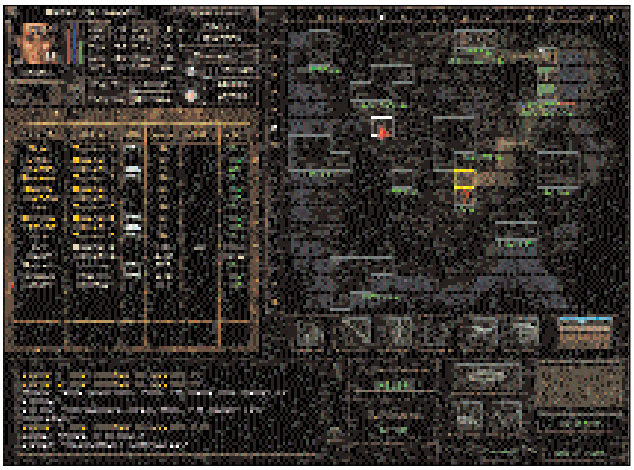
Számos weboldal átbogarászása után zsoldosokból álló kezdőcsapatod megszervezésével folytatódik a játék. A Nemzetközi Zsoldosszövetség (Association of International Mercenaries – A.I.M.) a legnagyobb és legtekintélyesebb munkaközvetítő, ami a rendelkezésedre áll – bár végső soron a saját ízlésed szerint megalkotott katona felhasználása is a Te döntésed. A zsoldosok toborzásában csupán egyetlen korlátozó tényező akad: a pénz. Minden zsoldosnak megvan ugyanis a maga ára.

Mi több, minden katona teljesen egyedi személyiséggel bír, továbbá saját felszerelési tárgyaik lehetnek. Néhányan jól kijönnek másokkal, némelyek viszont nem. Némelyikük igazi profi, hiszen egész életében ezt csinálta, mások pedig csak őrült, rögeszmés korcsok, akik pénzért még a saját anyjukat is eltennék láb alól. A játékban a legnagyobb kihívást nem más jelenti, mint hogy a zsoldosokból igazi csapatot kovácsolj, akik együttműködnek egymással, értékelik egymás képességeit; és ami a legfontosabb: képesek vállvetve harcolni, hogy áttörjenek a gonosz Deidranna királynő hadseregein, továbbá a szűkös költségvetésed adta kereteken belül mindent megtegyenek a kitűzött cél érdekében. A játék folyamán lehetőséged nyílik toborozni vagy bérelni néhány jóval fejlettebb zsoldost is, akik új tapasztalatokat szerezve bölcs veteránokká válnak.

Valóban érdekes folyamat, ahogyan csapatoddal összeszoksz; a játék folyamán szemed előtt bontakoznak ki embereid sajátos adottságai: személyiségük fejlődik és kiemelkedik. Néhányuk eléggé ostoba lesz (itt figyelmeztetném a szülőket: néhány zsoldos úgy káromkodik, akár egy kocsis!), mások bátrak és nagyszájúak, vagy félénkek és befelé fordulók, és majd' minden párbeszédük mosolyt csal az arcodra. Kedvencem a Steroid kódnévre hallgató zsoldos, aki Hans és Franz stílusú Schwarzenegger-hangon, pléhpofával jelenti be sebesülését: „Kilyuggatták a bőröm és most beázik".

Zsoldosaidat Arulco valamelyik városában dobják le, tehát innen mozgathatod őket. Valós időben és bármely irányban haladhatnak a térképed felvázolta lehetőségek szerint, azaz ha egy lápon átmenni két percig tart, akkor az valójában is két percig fog tartani. Arulco nagy térképe parcellákra, illetve körzetekre lett osztva, ezek mutatják, hogy zsoldosaid pillanatnyilag hol tartózkodnak. Annyi osztagba oszthatod őket, amennyibe csak akarod, és annyi különböző körzetbe küldheted szét a katonákat, amennyibe csak igényeid kívánják (ahogy a játék zajlik, zsoldosaid mindenütt ott lesznek, hogy a helyi lakosokból milíciát képezzenek ki). Arra figyelj, hogy egy időben csak egyetlen körzetre közelíthetsz rá. Ha úgy óhajtod, a vezérlőképernyőre is kiléphetsz, ahol felgyorsíthatod az idő múlását. Ezáltal a hosszú határás, az ország egyik feléből a másikba tartó utazások egy szempillantás alatt eltelnek, zsoldosaidat pedig egyenként nyomon követheted. Bármire ráközelíthetsz, ami felkelti az érdeklődésedet. Ha katonáid a közeli, valós idejű nézetből figyelve ellenségbe ütköznek, a játék innen, valamint az áttekintő (*Map Screen*) módból is azonnal körökre osztott küzdelembe vált át. Ekkor választhatod a szomszédos körzetbe történő visszavonulást (ha közel vagy a határhoz), vagy kiseperheted a környékedről a királynő összes emberét. A körökre osztott küzdelemben zsoldosaidnak tucatnyi különböző mozgási lehetőség áll a rendelkezésükre, és a megfelelő harcmódokat használva addig irányíthatod a csapatmozgásokat, ameddig meg nem nyerik a csatát. Kihasználhatod a terepet és a növényzetet, hogy fedezéket nyújtsanak, vagy előnyös helyzetbe kerülhess a lesből való támadások során. Zsoldosaid a térképen új hadállásba tudnak váltani, le tudnak guggolni, képesek kúszni, sőt, lopakodó módban be tudnak osonni azon ellenséges erők mögé, akik még nem fedezték fel őket. Harc közben meg is sérülhetnek, így valakinek egészségügyi ismeretekkel kell rendelkeznie, hogy elsősegélyben részesíthesse őket, esetleg – kegyetlen hajlamaid kiélésével – hagyhatod, hogy sebesülten harcol-

2. kép Kirohanás a Reptéri Kapuknál



3. kép Katonák mozgatása az áttekintőben: szektorról szektorra előre!

janak tovább. Saját katonáid és az ellenség különböző testrészek sérülését tudja okozni (láb meglövése esetén az áldozat átmenetileg összeesik, a fejlövés pedig komoly sérülést okoz); sőt, ha elég közel vannak egymáshoz, az ellenfelek szemtől szemben ugyancsak megküzdhetnek. A játék menete során a csatán kívül is a zsoldosoké a központi szerep, hiszen velük óvhatod meg az egyetlen bevételforrásodat jelentő arulcói arany- és ezüstbányákat. Minél több bányát tudsz az ellenőrzésed alatt tartani, annál több lesz a belőlük származó havi bevételed, ezáltal több zsoldost tudsz felbérelni, illetve eltartani. Ugyancsak a zsoldosok kezelik a fegyverzetet és az ellátmányt. Országszerte ők szerzik be és szállítják a fegyverzetet, a lőszereket és a páncélzatot, továbbá az elsősegélycsomagokat, a kabátokat, a feszítővasakat, a kantinokat, valamint az olyan különleges áruk tömegét, mint a rágógumi vagy sör. A zsoldosok a polgárőrséget is ki tudják képezni, ha pedig egyedül hagyod őket, különféle fogásokat gyakorolhatnak. Néha alvásra is szükségük van, különben kimerülnek, és képtelenek lesznek rendesen harcolni.

Előnyök

- Szuper játékmenet és egyensúly
- Nagyszerű hang, illetve zene
- Nagyon szórakoztató

Hátrányok

- Ócska grafika
- Bonyolult felület
- Nincs többjátékos mód

A JA2 grafikája, nos igen, tényleg 1999-ből való. A felbontás 640×480-ra korlátozódik, és bár nem vagyok biztos benne, hogy a több információ megszerzése érdekében nagyobbra lenne szükséged, a kép valahogy pontozott és darabos, nem felel meg a jelenleg használatos szabványoknak. Az érem másik oldala viszont, hogy ha egyszer igazán belelendülsz a játékba, hamar megfeledkezel a grafikáról, esetleg rájössz, hogy az ilyen típusú játékokban teljesen feleslegesek a 3D-ben megjelenített bokrok. A környezet hihetetlen összetett; sőt, minden körzetben több ezer tárgyat lehet kezelni: kinyitni, becsukni, felvenni, felrobbantani stb. Nagyon szórakoztató a változatos táj felderítése és a zsákmány utáni kutatás az épületekben. A karakterek animációja igazán kitűnő (a harcképek közeli látványa erősen emlékeztetett a Maxis cég The Sims elnevezésű játékára, a szereplők mozgásának részletessége csaknem ugyanolyan jó), a vágóképeket és a videobetéteket a játék motorja maga rajzolja ki. Egyetlen igazi hibát fedeztem fel a grafikai motorban: a terep modellezését összetett helyzetekben csaknem lehetetlen értelmezni, szinte képtelenség megmondani, hogy a közted és az ellenség (aki egyébként időről időre nyilvánvalóan keresztüllát a falakon és a fákon) között elhelyezkedő akadályba vagy azon keresztül lősz-e. A hanghatások bámulatra méltóak, és a zene – miközben a hosszú játékidőnek köszönhetően érzékelhetően ismétlődik –, mindig kifogástalanul illik a cselekmény hangulatához. Sajnálatos módon a Jagged Alliance 2-ben egyszerűen nincs lehetőség a többjátékos módra, mindemellett minden egyes játék kezdetén számos lehetőség áll a rendelkezésedre, hogy fölrázd a társaságot. A nehézségi szintek között nagyok a különbségek; a játékban szerzett tapasztalatod drámai mértékben függ attól, hogy melyik nehézségi szintet választottad. Dönthetsz a Tons of Guns beállítás mellett, ami kíméletlen mértékben növeli a rendelkezésre álló fegyvertípusok számát, vagy állítsd csak be a Sci-Fi lehetőséget, melynek köszönhetően zsoldosaidat az éj leple alatt idegen bolygóról származó rovarok támadják meg!

Linuxos megjegyzések

A Jagged Alliance 2 meglehetősen szerény gépkövetelményt támaszt: a Tribsoft cég csupán egy Pentium II 233 MHz-es processzorral, 32 MB memóriával, 4x-res CD-ROM-mal és 400 MB szabad merevlemez-területtel rendelkező gépet javasol. Mivel a JA2 nem igényel 3D-s gyorsítást, bármilyen videokártya alkalmas a futtatására, a hibamentes működéshez legalább az Xfree86 3.3.x változata szükséges a 640×480-as vagy nagyobb felbontásban. Egy OSS-megfelelő hangkártya is elkel, és természetesen a 2.2 vagy későbbi változatú Linux-rendszermag, valamint a glibc 2.1 vagy frissebb változata ugyancsak erősen javallott. A játék mind a 700 MHz-es, mind az 500 MHz-es rendszeren (Geforce/Debian Woody és G400/VA továbbfejlesztett

4. kép Munkatársakat keresünk

RedHat), amelyeken kipróbáltam, úgy futott, mint az álom. A telepítés méretének nagysága 305 MB-tól (alaptelepítés) 800 megabájtig (ha felraksz minden térképet és beszédfájlt a merevlemezre) variálható. Mindezt vagy héjprogrammal, vagy egy csinos kinézetű grafikus telepítővel teheted meg.

Összefoglaló

A játék valóban nagyszerű. Kezdetben, férfiasan bevallom, kicsit szkeptikus voltam, főleg a játék viszonylag magas korának, a low-tech grafikának és a látszólag túl bonyolult vezérlésnek köszönhetően. De amikor igazán megismertem a JA2-t, kezdtem megérteni, milyen kitűnő darabbal van is dolgom, és hogy miért pont erre esett a Tribsoftnál a választás. Azon játékok közül való, amellyel százszor játszva száz különböző játékmenetet ismerhetsz meg, és még legközelebb is ki akarsz próbálni valamit. Csáberejét nehéz lenne meghatározni: talán a zsoldosok mély gondolatokkal teli személyisége, az NPC-k párbeszédeinek egyéni vonásai, a *válaszd ki a saját kalandodat*-érzés vagy a stratégia, a taktika, a valós idejű és a körökre osztott mód közti *mindig oly sima átmenet* adja… Akár így, akár úgy, a Jagged Alliance 2 az a játék, amellyel órákon keresztül játszhatsz, és amikor befejezed, újra szeretnéd kezdeni. Erősen ajánlom mindenkinek, aki nem idegenkedik az összetettségtől.

J. Neil Doane
(caine@valinux.com) a VA Linux Systems mérnöke. Ha gép- és videojáték-hóbortja engedi, szívesen vezet repülőt, emellett gitározik és újabban hódeszkázik is (ez utóbbiban még igencsak gyengécske).

Adatok

Gyártó: Tribsoft
E-mail: info@tribsoft.com
URL: ➲ http://www.tribsoft.com

Az első pillantás az Apple G4-re

Matthew – a Linux-telepítés és a programfordítás kapcsán – az új Apple G4-én szerzett tapasztalatait osztja meg velünk.

Amikor először olvastam arról, hogy az Apple G4-alapú személyi számítógépet tervez, fogalmam sem volt róla, mi is az a G4. Szuperszámítógéphez fogható teljesítmény? Gigaflopos feldolgozási sebesség? Hogyan lehetséges ez? A G4-ben Motorola 7400-as processzor található AltiVec-egységgel. Az AltiVec az új PowerPC processzorokban található vektorműveleteket feldolgozó egység márkaneve. A Motorola a 7410 és 7450-es modelleket is bejelentette, amelyekben L2-es és hatalmas L3-as gyorstár, sebesebb CPU-mag és mélyebb, hétlépcsős utasítás-csővezeték lesz. Az AltiVec javított egészszámos és lebegőpontos műveletfeldolgozó egység. Új 128-bites feldolgozóegységgel rendelkezik, továbbá 32 vektorregiszterrel, és több mint 160 új utasítással, melyek a csővezetékben várakozó adat feldolgozását segítik elő.

A kiszolgáló gép egy Apple kétprocesszoros G4 450 MHz-es PowerPC volt 512 MB RAM-mal, 30 GB-os Quantum Fireball IDE merevlemezzel, MultiDrive-val (ez DVD-R-író, mely az összes egyéb lemezformátumot írja és olvassa), két IEEE 1394-gyel (Firewire-csatoló), 100 Mb/mp ethernettel, és egy csomó USB-csatlakozással. A billentyűzet és az egér egyaránt USB-n keresztül csatlakozott. Az Apple „Új Világ”-ra keresztelte ezt a gépet, és annak ellenére, hogy az elnevezés meglehetősen „marketingszagú”, az Apple a kifejezést azokra a gépeire használja, amelyekben a boot a ROM-programban található (a „Régi Világ” gépeiben ugyanezt a PROM-ban tárolták).

Nem tudom pontosan, hogy milyen indíttatástól vezérelve, de a Yellow Dog Linux-változat mellett döntöttem. Több lehetőség is kínálkozott volna (SuSE, LinuxPPC), de a YDL RedHat-alapú, aminek köszönhetően neve nem csengett teljesen ismeretlenül.

A YDL a Terra Soft Solutions terméke. A cég egy további terjesztéssel is bír, ami a Black Lab Linux nevet viseli, de én inkább a YDL-t ajánlom. Letöltöttem a YDL Champion Server 1.2.1 két CD-jét a Terra Soft egyik tükörkiszolgálójáról. Az első telepítőlemezként szolgál, a második neve pedig „Tasty Morsels” – ezen találhatjuk a vészrendszer ISO-fájlját, valamint néhány további PPC-s programot. Ezen a ponton merült fel a kérdés: hogyan tovább?

Elolvastam a YDL telepítési útmutatóját, amelyben kiderült, hogy yabootra van szükségem (yaboot – yet another boot loader), amit a HPFS-lemezrésze (a Mac félére) kell telepíteni, tehát létre kell hoznom egyet a maces rendszerprogramokkal.

A Mac OS9 újratelepítéséhez, majd a Linux telepítéséhez a következő lépéseket hajtottam végre:

- HFS-lemezrész (4 GB) létrehozása a yabootnak (és az OS9-nek);
- az OS9 újratelepítése gyári CD-ről;
- a CS (Champion Server) 1.2.1 CD-ről a yaboot, a yaboot.conf és a vmlinux.gz rendszermappába történő másolása.

A yaboot.conf nagyon hasonlít a lilo.conf-ra. Amikor a yaboot elindul, nyomjunk TAB-ot, hogy a rendszer kiírja a választható operációs rendszerek nevét – ezek valamelyikét kell beírnunk a promptnál.

Rendben, telepítsünk Linuxot! Helyezd be a YDL CD-t a DVD-meghajtóba és rendszerindulás közben tartsd nyomva C gombot! Így indíthatod CD-ről a rendszert. Ekkor megjelenik a YDL telepítőjének képernyője. A legtöbb esetben követheted a YDL telepítési útmutatójában leírtakat, de azért adnék egy jó tanácsot a lemezterületekről (hacsak nem volt már Linux a Maceden) – létre kell hoznod őket. Ezúttal már nem *ext2* lemezterületeket létesítesz, hanem *Apple_UNIX_SVR2*-típusúakat. A program neve is módosult egy kicsit: nem `fdsik`, hanem `pdisk`. A `p` paranccsal a lemezterületeket itt is megjelenítheted. Ha követted a tanácsaimat, kilenc lemezrészt kell látnod. Ezeket alapértelmezésben a rendszer hozza létre, és lehetőleg ne nyúljunk hozzájuk.

1. táblázat **Merevlemez felosztása (512 bájtos blokkokkal) a /dev/hda merevlemezen**

#	type name	length	base	(size)
1:	Apple_partition_map Apple	63	1	
2:	Apple_Driver43*Macintosh	54	64	
3:	Apple_Driver43*Macintosh	74	118	
4:	Apple_Driver_ATA*Macintosh	54	192	
5:	Apple_Driver_ATA*Macintosh	74	246	
6:	Apple_FWDriver Macintosh	200	320	
7:	Apple_Driver_IOKit Macintosh	512	520	
8:	Apple_Patches Patch Partition	512	1032	
9:	Apple_HFS untitled	5217260	1544	(2,5 G)
10:	Apple_UNIX_SVR2 root	4194304	5218804	(2,0 G)
11:	Apple_UNIX_SVR2 usr	16777216	9413108	(8,0 G)
12:	Apple_UNIX_SVR2 opt	8388608	26190324	(4,0 G)
13:	Apple_UNIX_SVR2 swap	262144	34578932	(128,0 MB)
14:	Apple_UNIX_SVR2 home	8388608	34841076	(4,0 G)
15:	Apple_Free Extra	15403650	43229684	(7,3 G)

Eljött az ideje, hogy kialakítod a szokványos lemezterületeket. A magam részéről úgy döntöttem, hogy külön lemezrészeket hozok létre a `/`, `/usr/`, `/opt/`, és `/home` könyvtáraknak és a lapozóterületnek. Lehet, hogy az eltérő összeállítás mellett döntesz, az enyémről listát találhatsz az *1. táblázatban*. Miután a változásokat a `w` utasítással kiírtuk a lemezre, lépjünk ki a `q` paranccsal, majd indítsuk újra a gépet, ellenkező esetben a program nem érzékeli a lemezen történt változásokat. Kezdd újra a telepítést, indulás közben ismét nyomva tartva a C gombot. Add meg az újonnan létrehozott csatolási pontokat, majd ezután elkezdheted a csomagok válogatását, ahogyan ezt egy szokványos RedHat-rendszeren is tennéd. A telepítés befejeztével ismét újra kell indítanod a gépet. Ez alkalommal ne nyomj le semmilyen gombot, hiszen most a MacOS-t szeretnénk indítani.

Ismét a MacOS-ben találjuk magunkat. Nyisd meg a yaboot.conf fájlt, amit CD-ről másoltál a rendszermappába, és vess rá egy pillantást! A ➲ http://www.linuxvilag.hu/G4.html címen az én fájlomat a 2. listában láthatod.

Figyeld meg a „Linux” címkéjét! A CD-n lévő yaboot.conf hibás, ezért a yaboot elé egy további `\\`-t kell írnom. Használd most a ALMA-POT-O-F billentyűkombinációt, és lépj be az Open Firmware-be. A „0 >” jelnél írd be a következőt:

Az Open Firmware

Szólnom kell néhány szót az Open Firmware-ről. Az Open Firmware-t az IEEE 1275-os szabvány írja le. Az Apple-gépet illetően ez az első érdekes újdonságok egyike. Nem láttam addog, amíg nem volt rá szükségem. A Mac bekapcsolás után 880 Hz-es sípolással jelzi, hogy sikeresen túljutott az indulás POST részén (alkatrész-ellenőrzés), és felkészül az operációs rendszer betöltésére. Ennél a pontnál az indulás folyamata az Alma-Opt-o-f billentyűk lenyomásával leállítható. Ha minden jól megy, a következő üdvözlőszöveget pillantjuk meg:

```
Apple PowerMac 3,3 3.4f1 BootROM built on
08/08/00 at 22:02:19
Copyright 1994-2000 Apple Computer, Inc.

Welcome to Open Firmware.
To continue booting, type "mac-boot"
and press return
To shut down, type "shut-down"
and press return

 ok
0 > _
```

A „0 >” egy készenléti jel legbelül az Open Firmware Forth-értelmező. A Forth veremalapú nyelv – hogy megértsük a logikáját, vegyük a következő példát:

```
0 > 3  [RETURN]
1 > 4  [RETURN]
2 > +  [RETURN]
1 > .  [RETURN]
```

Megjelenik az eredmény: `0 > 7`
Az első parancs „3”-at helyezett el a veremben. A készenléti jelben megjelenő számból tudhatjuk meg, hány elem található a veremben. Ezek után leraktunk „4”-et a verembe, majd utasítottuk az értelmezőt, hogy adja össze a számot. Most már csak egyetlen elem van a veremben. A „.” művelet a pillanatnyilag veremben lévő elemet adja vissza és megjeleníti értékét, jelen esetben „7”-t.
Az Open Firmware használatával igen sokat láthatsz a gépedről: megtekintheted például alapértelmezett beállításait, ha az alábbit írod be:

```
0 > printenv
```

Az 1. lista mutatja (➲ http://www.linuxvilag.hu/G4.html) a beépített környezeti változókat és alapértelmezett értékeiket. További jó adatforrás a `devalias` parancs. Írd be, majd nyomd le a Return billentyűt. Figyeld meg a hd értékét! Ez az első IDE merevlemezed fizikai címe, a hd pedig álnév a printenv által kijelzett teljes címre.

```
0 > boot hd:,\\yaboot
```

Némi villogás után megjelenik a LILO jele és elindul a Linux. Most már láthatod, mire képes az Open Firmware! A fenti parancs segítségével úgy futtathatsz fájlt a merevlemezről, hogy még nincs is operációs rendszer betöltve a gépen! Az X-et (Xfree86 3.3.6) a telepítés során állítottam be az `Xconfigurator`-ral: 1024x768-as felbontást választottam 24-bites színmélységgel. A yaboot.conf-hoz a következő sort adtam hozzá:

2. táblázat **AltiVec-ismerő gcc: új adattípusok**

Kulcsszó	*Méret*	*Adattípus*
vector unsigned char	16	unsigned char
vector signed char	16	signed char
vector bool char	16	unsigned char
vector unsigned short	8	unsigned short
vector unsigned short int	8	unsigned short
vector signed short	8	signed short
vector signed short int	8	signed short
vector bool short	8	unsigned short
vector bool short int	8	unsigned short
vector unsigned int	4	unsigned int
vector unsigned long	4	unsigned int
vector unsigned long int	4	unsigned int
vector signed int	4	signed int
vector signed long	4	signed int
vector signed long int	4	signed int
vector bool int	4	unsigned int
vector bool long	4	unsigned int
vector bool long int	4	unsigned int
vector float	4	float
vector pixel	8	unsigned short

```
append="video=aty128fb:vmode:17,cmode:24"
```

A rendszermag ezáltal megfelelően felismeri a gépen található ATI videokártyát. Ezt követően megnyitottam a `/etc/X11/XF86Config` fájlt, és a „Screen” részbe beírtam a `DefaultBitsPerPixel 24`-et, így a színmélységet nem szükséges minden X-indításnál kézzel megadnom.

Az AltiVec

Most, hogy a Linux települt, nézzük meg közelebbről az AltiVecet. Az AltiVec lebegőpontos és egészszámos műveleteket végző egység, amely 32 darab 128 bites regiszterben tárolt adattal dolgozik. A vektorokat kezelő egység SIMD-módszerrel (egy utasítás, több adat) dolgozza fel a vektorregiszterekben található adatokat. A processzor egyetlen utasítással egyszerre 4, 8 vagy 16 adategységen tud dolgozni. Lássunk erre egy példát!
A Motorola 162 új assembly-utasítást hozott létre, hogy a programozók kihasználhassák az AltiVec új lehetőségeit. Ezekről az utasításokról részletesen olvashatsz az „AltiVec Technology Programming Environments Manual”-ban (altivec_pem-ben).
A magasabb szintű C-utasítások, amelyek ezeket az új assembly-parancsokat használják, megtalálhatók az „AltiVec Technology Programming Interface Manual”-ban (altivec_pim-ben). Mindkettő letölthető pdf formátumban a Motorola weblapjáról, vagy pedig a ➲ http://www.altivec.org webhelyről.
Következő lépésem az AltiVec RPM-ek letöltése és telepítése volt. Ezekben a csomagokban a gcc (2.95.2) módosított változata rejlik, ami ezeket az új lehetőségeket használja. Telepítése a következőképpen zajlik:

```
rpm -U binutils-2.9.5.0.22-6.vec.ppc.rpm
rpm -i gcc-altivec-2.95.2-1i.ppc.rpm
rpm -i gcc-altivec-c++-2.95.2-1i.ppc.rpm
```

Telepítés után az új `gcc`-t így tudtam használni:

```
gcc-vec program.c -o program
```

A `gcc` a /opt/bin könyvtárba települ, így nem zavarja az alapértelmezett `gcc` programot. Az RPM-csomag létrehoz egy `gcc-vec` hivatkozást a /usr/bin könyvtárban, ami a /opt-ban található altiveces `gcc`-re mutat.
Az új vektorparancsok használatához olyan alkalmazásokat kell írnod, amelyek használják őket, és olyan `gcc`-vel kell fordítanod, ami ismeri az új utasításokat. Hiába fordítod le a szabványos C forráskódot az új `gcc`-vel, nem számíthatsz az AltiVec-egység okozta teljesítménynövekedésre. Az AliVecet ismerő `gcc` tud az új kulcsszavakról és függvényekről. Az első hely, amit tanulmányoznod kell, az altivec_pim, ebből tanulhatod meg a `gcc-vec` által ismert új parancsokat. A vektoradattípusokat a *2. táblázatban* tekintheted meg. Az `altivec_pim` alapján az AltiVec-ismerő fordítóknak a következő makróval kell szolgálniuk:

```
#define __VEC__
```

Ha olyan kódot szándékozol írni, amit többféle rendszeren is le lehet fordítani, és ami mégis kihasználja az AltiVec lehetőségeit, amennyiben ez utóbbi megtalálható a rendszerben, az alábbihoz hasonlót írj:

```
#ifdef __VEC__
        /* Ide jön a kódod */
        /* ... */
#else
        /* ha nincs más lehetőség, a régi
           módszert választjuk */
        /* ... */
#endif
```

Mellékeltem egy egyszerű példaprogramot az AltiVecet ismerő `gcc` használatához, ez a ➲ http://www.linuxvilag.hu/G4.html címen érhető el (3. lista).
Először figyeld meg a `typedef union` meghatározásokat! Mint már korábban említettem, az AltiVec regiszterei 128 bitesek. Ezek a meghatározások egyrészt biztosítják, hogy a fordító az ilyen típusú adatokat 128-bites határokra igazítja, másrészt azt is, hogy a `vektor` adattípus egyes elemeit egyszerűen elérjük. Végül az sem elhanyagolható, hogy a `union` adattípus használatával és a `printf()` függvény segítségével könnyen betekinthetünk a regiszterek tartalmába. Az `altivec_pim` szolgáltat formázott bemenetet és kimenetet a `scanf()/printf()` függvénypár segítségével. Elméletileg a következő C-kóddal egy lebegőpontos számokból álló vektorregisztert nyomtathatnál ki:

```
vector float f32 = (vector float)(1.1, 2.2, 3.3,
4.4);
printf( "%,vf\n", f32 );
```

Ehhez azonban a C programkönyvtárnak (glibc*) ismernie kell a vektorformátum-meghatározásokat. A GNU C programkönyvtár jelenlegi megvalósítása (2.2) nem ismeri ezeket, valószínűleg nem is fogja. Ezért remélem, marad időm és módosítani tudom a GNU libc-t, hogy megfelelő legyen erre a célra. (Ha tanáccsal tudsz szolgálni vagy érdekel a dolog, nyugodtan keress meg!) Figyeld meg továbbá a vektortípusok meghatározásának két különböző módját. Az első módszer állandó vektort határoz meg, az értékeket `cVals`-, `sVals`-, `iVals`- és `fVals`-okban tárolja, ahol a vektoradattípust ugyanabban az állításban adjuk és határozzuk meg. Ez a példa azt mutatja, hogyan tároljunk állandókat (amik futásidőben nem változnak) vektorokban. A másik módszer lényege, hogy megad egy `union`-típust, majd a vektort futásidőben elemenként tölti fel. Ezen eljárás segítségével adatokat olvashatsz be egy átmeneti tárolóból, azokat vektortípusú változóba másolhatod be, majd ezt a változót átadhatod a vektort ismerő függvényeidnek.
Végül figyeld meg a `vec_add( )` függvény alakját! Minden esetben ugyanazt a `vec_add( )` függvényt használtam, és mindig jó eredménnyel tért vissza, függetlenül attól, hogy változói `vector short`, `vector int` vagy `vector float` típusúak voltak-e (a két összeadandó azonos kell legyen). Ebben az esetben a fordító értelmezte az adattípusokat, melyeket a `vec_add( )` függvénynek adtam át, és létrehozta számomra a megfelelő `vadd` assembly-utasítást. Az alábbi példában láthatjuk, hogyan képes a fordító a megfelelő egyeztetést létrehozni:

```
vector float a,b,c;

/* Assign a,b */

/*     ...      */

c = vec_add( a, b);
```

Ezt a következő assembly-utasítás hozza létre:

```
vaddfp c,a,b
```

A dolog egyre egyszerűbbé válik. A program fordításához használd a következő parancsot:

```
gcc-vec -fvec vecdemo1.c -o vecdemo1
```

A fordító `-fvec` kapcsolója a vektorparancsok értelmezésére utasítja. Ha nem adod meg a `-fvec` kapcsolót, a fordító nem ismeri fel a vektoradatokat és programfordulás közben hibát jelez, ez emlékeztet majd arra, hogy legközelebb ne felejtsd el használni a kapcsolót.
A ➲ http://www.linuxvilag.hu/G4.html címen a 4. listán látható a program kimenete.
A fentiekben rövid bevezetőt kíséreltem meg nyújtani a Linux PowerMacen történő használatáról, továbbá a linuxos programozók számára elérhető AltiVec-lehetőségekről.
Köszönetet szeretnék mondani az AltiVec-fórum tagjainak. A levelezési lista segítsége felbecsülhetetlenül értékes volt, e nélkül sokkal nehezebb lett volna elindulnom. Ugyancsak köszönettel tartozom az AltiVec-fejlesztőknek, amiért számtalan fejlesztői eszközt biztosítottak, és lehetővé tették, hogy a programozók könnyen dolgozhassanak egy olyan hatékony rendszeren, mint a G4-es.

Matthew Fite
(mattfite@yahoo.com) feleségével Észak-Virginiában él, és beágyazott programokat fejleszt. Annak ellenére, hogy napközben kereskedelmi RTOS-szel kell foglalkoznia, titkon arról álmodik, hogy egyszer bevezetheti az RTLinuxot. Mindig új tervei születnek, bár a felesége szerint már így is eleget foglalkozik a gépekkel.

Kapcsolódó címek

AltiVec ➲ http://www.altivec.org/
Motorola ➲ http://www.mot.com/AltiVec/
Apple Developer Connection ➲ http://developer.apple.com