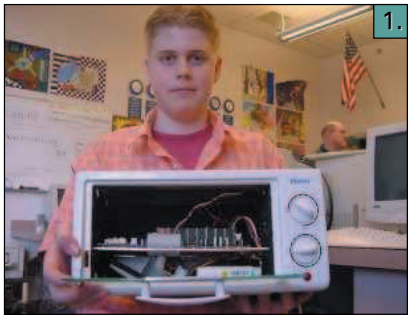
A szerelemnek múlnia kell...

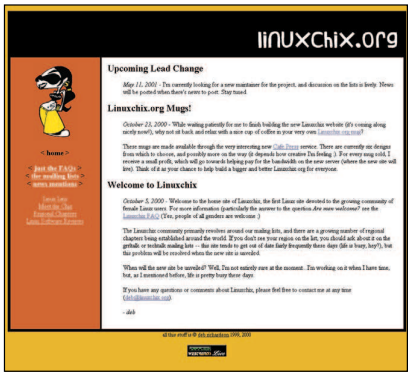
Minden terméknek megvan a maga életciklusa. Előbb-utóbb minden felett eljár az idő. Még a szerelemnek is múlnia kell, ahogy Zorán mondta. Hogy mi köze van a szerelemnek a számítástechnikához? Nagyon is sok!
Mi alapján döntünk, amikor két számunkra ismeretlen termék között kell választanunk? Egyéni érzéseink döntik el a kérdést. És amikor az operációs rendszerek között kell választanunk?
Itt már összetettebb a helyzet, hiszen fontos, hogy használni tudjuk, ehhez pedig két fontos feltételnek kell teljesülnie: egyrészt meg kell ismernünk, másrészt képesnek kell lennie azon feladatok ellátására, amelyekre szánjuk. Természetesen ha már ismerjük, akkor sokkal valószínűbb, hogy választásunk rá esik. Ezért is fontos, hogy odafigyeljünk az oktatásra. Minél több rendszert, irányt mutatunk meg a tanulóknak, annál könnyebben találnak a céljainak megfelelő eszközt. Akit érdekel, hogyan látja egy tanár a jelenlegi hazai helyzetet a Linux és a közoktatás szemszögéből, lapozzon a 40. oldalra!
Rendben, nincs több Kaland-Játék-Kockázat. A minap hallottam, hogy még a tavaknak is megvan a maguk életciklusa. Imádom a Balatont, még most is, amikor egy-egy hétvége alkalmával tízezrek támadják meg a siófoki nagystrandot, de azt mondják, haldoklik a mi drága tengerünk. Még küzdünk érte, de már csak ideig-óráig él.
Én azért reménykedem, nem szeretném hiányolni azt a kellemes érzést, amikor az autópályán előttem és utánam is több kilométeren keresztül csak boldog emberek szoronganak kis bádogdobozokban. Bár ez is nézőpont kérdése: a szembejövők hülyére röhögik magukat a sokakon. Könnyű nekik, hűti őket a menetszél. Nem baj, fagy még meg az ő tevéjük is!
Visszakanyarodva az újsághoz, kezdünk kialakulni. A csapat nevében is köszönöm mindenki véleményét, és igyekszünk is igazítani a kínálaton. Sokan kérték, hogy foglalkozzunk egy kicsit többet a webfejlesztéssel kapcsolatos kérdésekkel, nekik szól négy cikkünk is, főleg a PHP és a Java háza tájáról. A PHP annyira érdekes téma, hogy még egy könyvet is ajánlunk hozzá! Igyekszünk a hálózatfelügyelők kedvében is járni, tovább foglalkozunk a csomagszűréssel, az egyszerűen kezelhető hálózati munkaállomásokkal (linuxos terminálokkal), sőt, két levelezőrendszerrel, a PostFixszel, és a SuSE EMail Server II-vel. A rendszergazdák nagy hasznát vehetik a Nessusnak, mindenképpen ajánlom nekik is az 52. oldalon található cikk elolvasását! Az otthoni felhasználókat szintén megörvendeztetjük a multimédiához kapcsolódó cikkeinkkel, egy játékleírással, valamint pár oldal erejéig a Wine rövid ismertetőjével és így tovább.

Talán csak a beágyazott rendszerek szorultak egy kicsit háttérbe. Sokan írták, hogy nem tudják, mi az a beágyazott Linux. Nos, egy lelkes fiatal készített egy háziasszonyok számára is érthető linuxos eszközt (lásd az 1. és 2. képet)! A pontos leírás megtalálható a ➲ http://www.riverdale.k12.or.us/linux/toaster/ címen.
De a lap életében más változások is történnek. Például megváltozott az újság papírja. Először nagyon féltem, hogy milyen visszhangot kelt majd olvasóinkban, amiért nem „csillogunk” annyira, de szinte csak elégedett véleményeket kaptunk.
Igyekszünk egyre több hazai cikket is bezsúfolni a címlap alá. Kérek mindenkit, ha van olyan téma, amely érdekli, írja meg nekünk! És ami talán a legfontosabb, személyi változások is történtek a lapnál! Gyula végre elvállalta a lap szakmai szerkesztését, és én tudom a legjobban, hogy komoly feladatot vett a nyakába; valamint örömmel üdvözölhetjük csapatunkban Dér Csilla olvasószerkesztőt. (Ha valaki ráér, dobja meg őket egy „Szia” témájú levéllel a Csontos.Gyula@linuxvilag.hu, valamint a Der.Csilla@linuxvilag.hu címen.)
Jut eszembe, most már nőuralom kezd kialakulni, úgyhogy ezennel meghirdetem a legjobb szőke linuxos viccpályázatot, nevezni bármilyen viccel lehet, amiben a Linux is szerepel! Külön kategóriát indítunk a szőke nőket tartalmazó vicceknek! A pályázatokat a vicc@linuxvilag.hu címre kérjük, eredményhirdetés és a legjobb viccek gyűjteménye a következő lapszámban! De ne olyan vicceket küldjetek, hogy „Mit csinál egy szőke nő két Linux feliratú vasgolyóval…”!

> Bemegy az ürge egy irodába, hát látja, hogy egy nagy szőke lobonc lóg csak ki a számítógépből.
> – Elnézést, asszonyom, tudna segíteni?
> – Egy pillanat, csak átültetem a rendszermagot...


1.


2.

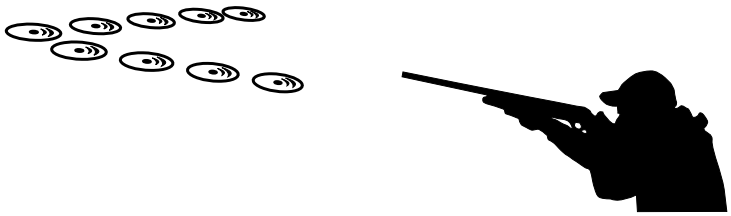


➲ http://www.linuxchix.org

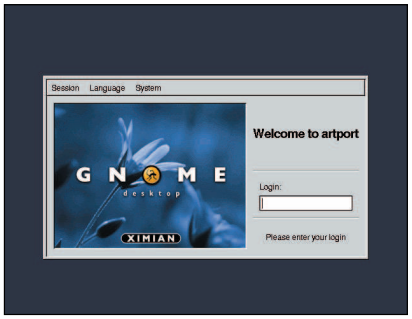
Szy György a Linuxvilág főszerkesztője, a Kiskapu Kiadó vezetője. Mindenki véleményét és levelét örömmel várja az alábbi levélcímen: Szy.Gyorgy@linuxvilag.hu

Programvadászat

A Gnome és a világuralom



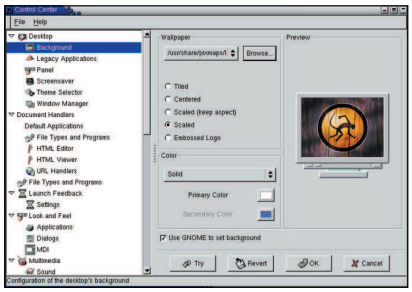
A Ximian Gnome megegyezik a Helix Gnome-mal, csak a nevét változtatták meg. Ez a Gnome kezelőfelület kiegészített változata, teljesen egységes felületet ad a különféle Linux- és Unix-változatoknak.



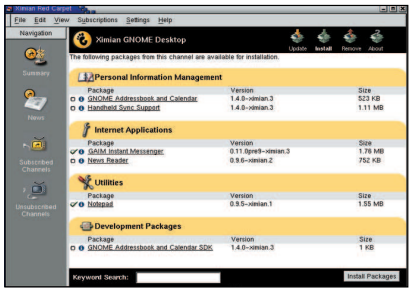
Rengeteg programot tartalmaz, amelyek alkalmassá teszik mind az irodai, mind a grafikai és egyéb feladatok megoldására. A Sun és a Hewlet-Packard is a Gnome kezelőfelületet választotta Unixa „arcául”, a fejlesztés egyelőre gőzerővel folyik, a Sun Solarishoz azonban már letölthető egy próbaváltozat.

Ennek az egységesítésnek óriási előnye, hogy a felhasználók szempontjából nézve teljesen mindegy, milyen Unix elé ültetik le őket, ha dolgoztak már ilyen környezetben, nem lesz számukra idegen, és mindent a megszokott helyén fognak találni. Ezzel el is érkeztünk a felhasználóbarát Unix-rendszerekhez, amelyektől többé már az átlagos felhasználókat sem kell félteni, mindössze azt kell elérni, hogy ők se féljenek a változásoktól és az újdonságoktól – bár az emberek nagy többsége beidegződött szokásainak a rabja és nem mer változtatni, váltani.

A mostani szám második korongját teljes egészében a Ximian Gnome foglalja el, mivel több száz fájlról van szó minden egyes Linux-változathoz. A lemezmelléklet a Debian Potato i386, a RedHat 7.0 i386, a Mandrake 7.0 i586 és a SuSE 6.4 i386 Linux-változatok telepítőkészleteit tartalmazza. (Helyszűke miatt a RedHathoz tartozó változat a harmadik korongon kapott helyet.)

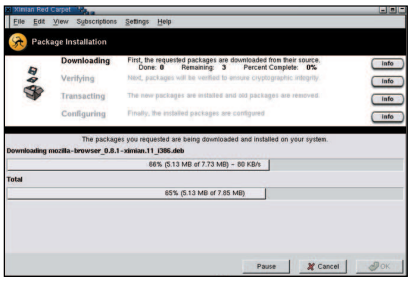


Számos hasznos programot kapunk a csomaggal, ezek között megtalálható a Gimp képszerkesztő, az AbiWord szövegszerkesztő, a Mozilla böngésző, a Red Carpet software manager program (erről még lesz szó a későbbiek során), és a Nautilus fájlkezelő-tehát minden együtt van ahhoz, hogy egy bárki által Linuxszal megálmodott és létrehozott irodai munkaállomást kényelmesen használhassanak. A Control-Center segítségével felületünk beállítását, testre szabását könnyedén elintézhetjük.



Red Carpet

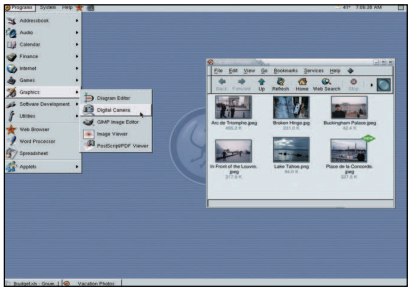
A Ximian Gnome részét képezi a Red Carpet program, melynek segítségével könnyedén és gyorsan frissíthetjük csomagjainkat vagy távolíthatunk el programokat. A telepített csomagok közül önműködően megkeresi a legújabb változatot, és segítségével végre is hajthatjuk a frissítést. Az időközben megjelent egyéb programok szintén szerepelnek a letöltött listában, ezeket is innen telepíthetjük.



A Red Carpet program minden olyan tulajdonsággal rendelkezik, amire egy fejlett csomagkezelőnek szüksége lehet, ilyen például a teljes körű függőségkezelés.

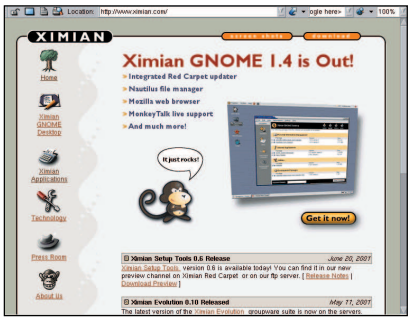
Nautilus

A Nautilus a Ximian Gnome fájlkezelője. Természetesen az egész felületen másolhatunk, mozgathatunk a *húzd és ejtsd* (drag and drop) módszerrel. Rendelkezik képnézegetővel is, ami a képfájlokat kicsinyítve



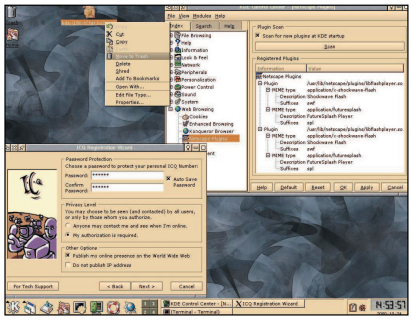
jeleníti meg, így könnyebben választhatjuk ki a szükséges fájlokat.

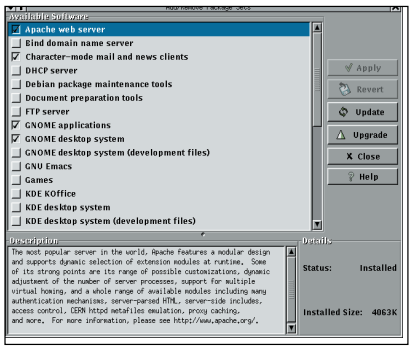
További leírások a ➲ www.ximian.com és a ➲ www.ximian.com/apps/redcarpet.php3 címen olvashatók.



Progeny 1.0 Newton 2-es CD

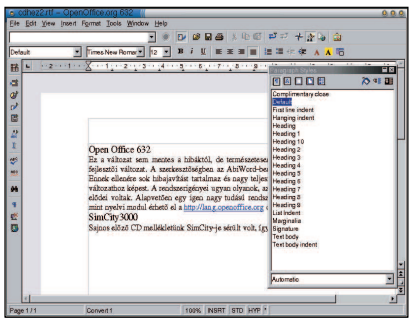
Számos hasznos programot tartalmaz a Progeny második CD-je. Egyszerűen adjuk hozzá ezt a CD-t is az apt listánkhoz az `apt-cdrom add` paranccsal! Ezt követően a programokat máris telepíthetjük az `apt-get install` *programnév* utasítással, vagy használhatjuk a dselect programot és a grafikus csomagkezelőt is. Aki a Gnome-mal szemben a KDE felületet részesíti előnyben, most áttérhet kedvencére, mivel ez a CD azt is tartalmazza. Jó böngészést az új programokhoz!





OpenOffice 632

Ez a változat sem mentes a hibáktól, de ez természetesen így van jól, hiszen még csak a fejlesztői változatot tartjuk a kezünkben. Szerkesztőségünkben az AbiWordben írt .rtf fájlokat ugyan rosszul jelenítette meg, ennek ellenére az előző változathoz képest már számos hibajavítást tartalmaz, és nagy teljesítménynövekedést tapasztaltunk. Rendszerigényei: 64 MB RAM és körülbelül 200 MB merevlemez-terület. Érezhetően gyorsabb lett, mint elődei voltak. Ha elkészül, igen nagy tudású rendszert üdvözölhetünk majd benne. Ha nem angol nyelven szeretnénk használni, célszerű lesz az egész csomagot lefordítani, ehhez pedig a ➲ http://tools.openoffice.org/openoffice_lang.html címen kaphatunk segítséget. Teljesen új ps-nyomtatást használ, így a támogatott rendszereken telepítés után minden további beállítás nélkül nyomtathatunk.



Magazin

Új könyvtárat nyitottunk *Magazin* néven. Ebben a könyvtárban a különféle cikkekhez szervesen kapcsolódó forráskódok, programocskák és egyéb kiegészítők lelhetők fel. Most a következőket rejti: Maya-animációkat MPEG formátumban a *Magazin/Maya* könyvtárban, ezenkívül a Postgresql és a PHP, illetve a PoV-Ray ismeretek című írásunkhoz tartozó forráskódok is itt találhatók meg.

bug1 – készítője Mike Miller

Rendszermag

A jelenlegi korongon a 2.4.5-ös rendszermag található meg. A változások közül néhány:

- tms380tr meghajtófrissítés,
- USB-javítások és -frissítés,
- RAID1/5 hibajavítás,
- ARM- és PPC-javítások.

A teljes listát a ➲ http://www.kernel.org/pub/linux/kernel/v2.4/ChangeLog-2.4.5 címen olvashatjuk el.

Mozilla 0.9.1

CD-inken nyomon követjük a nyílt forráskódú Mozilla böngésző fejlődését. A CD összeállításának pillanatában a 0.9.1-es változat a legfrissebb. Sebessége már kielégítő és memóriahasználata sem olyan nagyfokú, mint azt eddig megszokhattuk. Többnapos, szinte folyamatos használat alatt sem fagyott le vagy szállt el, így már nyugodtan ajánlom mindazok számára, akiket nem riaszt el a változat számában a nullás kezdet.

A futtatásához szükséges vas- és rendszerigények:

- glibc 2.1 (vagy magasabb) könyvtár,
- Intel Pentium-megfelelő 233 MHz-es processzor,
- 64 MB RAM és
- 26 MB szabad hely a merevlemezen.

SimCity3000

Előző CD-mellékletünk SimCityje sajnálatos módon sérült volt, így most orvosoljuk ezt a hibát.

Heroes of Might and Magic III

A mostani játékleíráshoz tartozó Heroes III bemutató változata a CD Jatekok/Heroes3 könyvtárában található meg.

Rendszerigénye:

- Linux 2.2.x rendszermag,
- Pentium osztályú processzor,
- legalább 800x600-as felbontás – 16 bites színmélységgel és XFree86 3.2 vagy újabb grafikus környezet,
- négyszeres CD-ROM-meghajtó,
- 32 MB RAM,
- hangkártya és
- 150 MB merevlemezterület a telepítéshez.

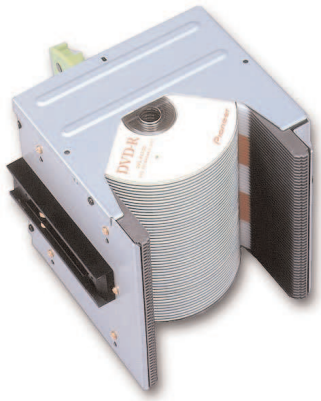
Csontos Gyula
(Csontos.Gyula@linuxvilag.hu)
a Linuxvilág hír- és CD-szerkesztője, valamint a www.linuxvilag.hu tartalomfelelőse.

Az DVD-mentés nagymesterei

A svájci központú BaSys cég bemutatta a legújabb Pioneer DRM-7000-est és kisebb testvérét, a DRM-1004V40-est. A DRM-7000-es típusú raktáregység akár 720 CD tárolására és kezelésére is alkalmas, egyszerre pedig négy lemez anyagát képes feldolgozni. A nála kisebb DRM-1004V40-es már csak száz lemezt képes tárolni. Mindkét típus megfelelő a CD-ROM, CD-R és DVD-ROM adathordozók feldolgozására, illetve a DRM-7000-sen még a CD-RW is használható. A berendezés természetesen NT, Unix és Linux alatt is működik.

A cég referenciái közé sorolható, többek között a BBC tévécsatorna, a BMW és az EBU. Felhasználási területük széles körű: CAD, videó-, fotó-, hanganyagok és szöveges dokumentumok mentésére egyaránt alkalmas.

➲ http://www.basys.hu



Linux összehasonlító

Az alábbi címen a főbb Linux-változatokról tudhatunk meg hasznos adatokat táblázatba foglalva. Ennek köszönhetően könnyebben tudjuk kiválasztani a számunkra megfelelő Linux-változatot. Ezek mindegyikének van saját oldala is, ahol a programcsomagok frissességét, változatszámát ismerhetjük meg. Összehasonlíthatjuk például a Debian Slink, Potato, Woody és Sid csomagjait.

➲ http://home.kimo.com.tw/ladislav/distro_compare.htm

BeIA operációs rendszer

A Be inc. beágyazott rendszerekhez készítette a BeIA operációs rendszerét, mely kiváló felületet nyújt az otthoni és az internetes eszközökhöz. Ilyenek például

a set-top boxok, a kézi gépek, a mobiltelefonok és a házi szórakoztató központok. A Sony is ezt választotta a Sony e Villa(TM) Network Entertainment Center nevű készülékének operációs rendszeréül. Támogatja a HP DeskJet nyomtatókat, legkisebb vasigénye x86-os Pentium osztályú vagy nagyobb, illetőleg PowerPC processzor, 32 MB memória, 8 MB tárhely, utóbbi flashlemez is lehet.

➲ http://www.beia.com
➲ http://www.be.com

Linuxos nyomtatás

Biztosan sokan próbálták már meg nyomtatójukat szóra bírni kedvenc operációs rendszerünk, a Linux alatt. Bizony ez nem könnyű feladat, de szerencsére az Easy Software Products cég

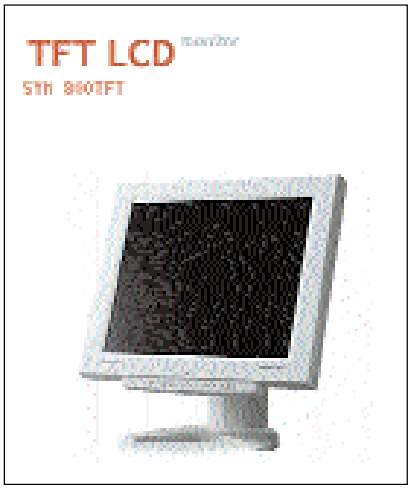
(➲ http://www.easysw.com) elkészítette az általános, szinte minden Unixon használható nyomtatókezelést: a CUPS-ot. Ennek segítségével webes felületen állíthatjuk be nyomtatónkat. Szinte minden nyomtatótípust ismer, mielőtt azonban nekilátnánk a telepítésnek, érdemes körülnézni a

➲ http://www.linuxprinting.org oldalon az adatbázisban, hogy az általunk kiválasztott nyomtatótípus valóban támogatott-e.

➲ http://www.linuxprinting.org
➲ http://www.cups.org

Monitor-nap

Június közepén a Tech Data Intel monitornapot rendezett. A Tech Data a tízéves Computer 2000 utódja. A névvel ellentétben a szakemberek személye és a minőség, valamint a cég alapelvei nem változtak. A bemutatón világviszonylatban is elismert cégek képviseltették magukat, mint például a Philips, a Samsung, a Sony és az Intel. Elsőként a Samsung mutatta be legújabb monitorjait és a legutóbb kifejlesztett Dynaflat X típusú képcsövet. A Samsung a sík képcsőben, a minőség csiszolásában és az energiatakarékosságban látja a monitorgyártás jövőjét. Az Dynaflat X kifejlesztése új korszakot nyitott a képcsövek világában. Ígéretük szerint az új foszforréteg, az új bevonatok, a szebb kontrasztok és a képpontméret további csökkenése



meghozza a várt minőségjavulást. A Samsung kínálatában szereplő 27-féle típusból a LCD folyadékkristályos TFT-t is kiemelték. Ezek a modellek 15, 17 és 18 hüvelykes méretben jelentek meg a piacon, a 800TFT és a 770TFT típusú LCD-k 1280x1024 képfelbontásúak. Természetesen a Sony szintén számos újdonsággal lepett meg bennünket, többek között TFT L181-es típusú 19 hüvelykes (1280x1024/85-as üzemmód) legújabb csodájával. A bemutató végén pedig árverésre bocsátott egy 15 hüvelykes M51 multimédiás TFT monitort. A kikiáltási ára 40 ezer forintról indult és 155 ezer forintnál ütött le a kalapács, ez pedig majdnem megegyezik éppen akciós árával. A Philips is tartotta a szintet, és újdonságai közül a Light Frame program érdemel nagyobb figyelmet, mely nagyobb fényerőt és élesebb képet nyújt a felhasználónak. A CD-ről ugyanúgy, mint bármilyen más programot, fel lehet telepíteni a számítógépre, segítségével hatékonyabb lesz a monitor beállítása.

➲ http://www.techdata.hu

IBM – Nyárköszöntő party

Az IBM júniusi Nyárköszöntő összejövetelén mutatta be az IBM eServer pSeries 620-as és 660-as sorozatát. Az új termékekkel tulajdonképpen a tavaly megjelent RS/6000 F80/H80 típusokat váltották fel. A változást a nagyobb teljesítményű RS64-IV processzornak, a kétszeres memóriának és a hetven százalékkal nagyobb OLTP teljesítménynek köszönhetően hozta.

A bemutatón szó esett az AIXL-ről, melynek már a neve mutatja (L), hogy linuxos alkalmazások is futtathatók rajta. Az AIX új nemzedéke az 5L-változat, mely a legújabb unixos módszerek ötvözete, s ebből következik linuxos irányultsága is. A rendszert a SuSE és a RedHat is támogatja. AIX 5L jellemzői a következők:

- 64-bit Unix operációs rendszer (POWER és IA-64 processzorokra),
- TCP/IP és webkiszolgálási újdonságok,
- új RAS képesség,
- 24/32-processzoros támogatás egészen 96 GB memóriáig,
- Java V1.3.0 JDK és futtató.

Irány a Terasz!

Egy fiatal csapat elképzelése alapján készült el a Terasz.hu internetes portál. Ez azonban csak első pillantásra hasonlít a többire, mert megtalálhatók rajta a szokásos témakörök: hírvilág, magazin, enciklopédia, linkajánló, chat és fórum, sőt még internetes áruházban is vásárolgathatunk. A terasz viszont túllépve a ma már alapszolgáltatásnak számító tárgykörökön, számos fiatalos hangvételű újdonsággal lep meg bennünket. A művészeti oldalra kattintva híres színészek előadásában hallgathatunk verseket vagy szemezgethetünk az érdekes válogatásból. Játszhatunk az álomtőzsdén, ahol a részvények helyett termékekkel, szolgáltatásokkal, élőlényekkel, személyekkel vagy fogalmakkal kereskedhetünk. Érdemes ellátogatni ide, és üzleti érzékünk vizsgájának helyszínéül nem a BUX-ot választani. Próbára tehetjük tippelő képességünket az ÁlomPC játékban, melyben a Terasz irodájában használt monitorok típusát találhatjuk ki. Az ingyenes játékokon túl fizetős változatot is terveznek, ahol már értékes nyereményekre számíthatunk. Az irodában elsősorban a fejlesztők gépein és a kiszolgálókon használnak Linuxot, ezek közül is a Debian- és a SuSE-változatokat futtatják.
➲ http://www.terasz.hu

Linuxos irodai gép

Kezdő vállalkozásoknak éppen annyira nagy anyagi áldozatot jelent a számítógép beszerzésén túl a jogtiszta programok és az operációs rendszer megvásárlása, mint a már meglévő vállalkozások bővítése. Erre kínál megoldást a TOP Computer Kft. a DTK Abacus számítógépcsaláddal, melynek tagjait előre telepített Linuxszal és ingyenes

StarOffice irodai csomaggal értékesíti. Az így előkészített számítógép beállítás után azonnal alkalmas internetezésre, levelezésre, szövegszerkesztésre, táblázatkezelésre, tehát mindenre, amire egy irodában csak szükség lehet. A Linuxnak köszönhetően könnyedén alakítható ki belőlük biztonságos nagyteljesítményű helyi hálózat. Nagyon változatosak a számítógép-összeállítások: a Celeron 700-as processzortól a Pentium III-as processzorokon keresztül akár AMD processzorral is kérhetjük a gépeket, memóriaméretük 64–128 MB-ig, merevlemezük 10–20 GB-ig választható, kérésre természetesen egyéni igényeket is kielégítenek. A számítógépek három év teljes körű jótállással rendelkeznek.
➲ http://www.topcomputer.hu

SuSE 7.2 magyar változat

A SuSE Linux hazai képviselete június végén piacra dobta a SuSE Linux 7.2-es magyarított változatát. Ez számos új kialakítási, teljesítménybeli, biztonsági és multimédiás újítást tartalmaz. A KDE 2.1.2 grafikus munkakörnyezetre épül, amelynek központi eleme a Konqueror böngésző – ez tulajdonképpen böngésző és fájlkezelő egyben. A program támogatja az összes ismert webformátumot, többek között a Flash, a RealAudio és a RealVideo médiaformátumokat is. A biztonságos internetes banki ügyintézés (tranzakciók lebonyolítása) érdekében az SSL (Secure Socket Layer) biztosítja a védelmet. A grafikai felületet kedvelők számára az Xfree86 4.0.3-as változat számos kényelmi szolgáltatást nyújt. Az újdonságok között még megemlíthetjük az MP3-lejátszókat, a dobgépet, a MIDI billentyűzetet-programot és a sokrétű audio-video szerkesztőstúdiót, melyek szintén helyet kaptak a csomagban. A program ingyenes frissítésével sem lehet sok gondunk, mivel a SuSE YOU állandó elérésű frissítő néhány egérkattintás után önműködően gondoskodik róla. A 7.2-es telepítésekor a SuSE lehetőséget kínál személyes tűzfal kialakítására, sőt még a vírusoktól sem kell tartanunk az AMaViS (A Mail Virus Scanner) használata mellett. A fájlként csatolt titkosított fájlrendszer (loop mounted crypto file system) módszer segítségével bizalmas adatokat is tárolhatunk. A következő változat előreláthatólag 2001 őszén jelenik meg a boltokban.
➲ www.linuxvilag.hu

Körösztös Anita (Korosztos.Anita@linuxvilag.hu) A Linuxvilág marketingfelelőse. Nem olyan rég csöppent bele a Linux világába, azóta lelkesen foglalkozik vele. A barátaival töltött időt tartja a legfontosabbnak és lelkesedik a vízi- és téli sportok iránt is.

Csontos Gyula (Csontos.Gyula@linuxvilag.hu) a Linuxvilág hír- és CD-szerkesztője, valamint a www.linuxvilag.hu tartalomfelelőse.

Gondolj rá úgy, mint az árverésre

Szóval, milyen gyakran is pötyögik be a nevedet a Google keresőbe? Hogy ezt megtudjuk, létrehoztunk egy hirdetést a Google AdWords oldalán, majd megkértük a Google üzemeltetőit, hogy becsüljék meg, mennyibe kerülne havonta a hirdetés, ha akkor jelenne meg, amikor valaki a következő nevekre keres:

Larry Augustin: 0
Chris DiBona: 0
Phil Hughes: 0
Rob Malda: 0
Don Marti: 4000
Rick Moen: 0
Bruce Perens: 0
Eric Raymond: 4000
Doc Searls: 0
Richard Stallman: 0
Linus Torvalds: 1300
Richard Vernon: 0
Bob Young: 0

Természetesen kipróbáltuk operációs rendszerekre is.
Linux: 4 284 200
Windows: 5 653 800
Unix: 872 900
A becslések nem kerülnek pénzbe. Próbáld ki te is a ➲ http://adwords.google.com/ weblapon.

A Microsoft és a PHP erősödik

A Netcraft cég 1995 szeptembere óta figyeli világszerte a kiszolgálók piacát. Felmérései tanúsága szerint akkoriban még az NCSA kiszolgálója vezetett óriási fölénnyel, 1999-re azonban ez a termék végleg eltűnt a felmérésekből.
Az adatok tanúsága szerint világszerte az Apache a piacvezető webkiszolgáló, ugyanis 1996 óta e terméké a legnagyobb a piaci részesedés (tavaly meghaladta a hatvan százalékot). A Netcraft jelentése szerint idén januárban azonban 1,29 százalékot veszített részesedéséből az Apache. A Microsoft IIS 1,82 százalékkal erősödött, a harmadik helyezett Sun Iplanet-kiszolgáló pedig 0,29 százalékkal visszaesett. Az Apache és a Microsoft egyaránt növekedést mutatott (abszolút értékben). A Netcraft 16 207 982 tartomány esetében talált Apache-t, és 5 903 512 esetben Microsoft-kiszolgálót. A Netcraft más fejleményekre is felhívja a figyelmünket. Elemzése alapján a Microsoft webkiszolgálópiacon mutatott viszonylagosan állandó részesedésével ellentétben az SSL-t alkalmazó kiszolgálók körében állandó és töretlen növekedés tapasztalható, ide számíthatjuk már az Interneten titkosított tranzakciókat bonyolító webhelyek 49 százalékát. Nem tagadható, hogy ebben nagy szerepet játszanak a Microsoft alkalmazásai, mivel nincs a Microsoft Commerce Servernek egyértelműen megfeleltethető termék a unixos világban.
A Netcraft felmérése egyúttal fényt derít arra is, hogy a PHP-t rendkívül gyors ütemben helyezik üzembe: világszerte több mint ötmillió Apache kiszolgálón található PHP-modul. Ez az Apache-rendszerek majdnem 40 százalékát jelenti. Igaz, hogy valójában jóval kevesebb helyen használják a fent említett modult, de az általánosan elterjedt nézet szerint a PHP-t könnyebb programozni, mint a Perlt vagy a JSP-t, és feltűnően rövid idő alatt fejlesztők széles rétegének tetszését nyerte meg. A PHP, a MySQL adatbázis-kiszolgálóval és az Apache webkiszolgálóval magától értetődő alkalmazásfejlesztési környezetet alkot, és a Microsoft IIS/ASP/SQL-kiszolgáló hármas méltó linuxos megfelelője.

Éhen halok!
A tárolás olyan, mint az evés. Takarékoskodhatsz rajta, de enned azért muszáj.

Colin Ferenbach, az EMC tárolókkal foglalkozó cég jövőjéről

Ők mondták

A végső próbatétel – Szinte minden férfi jól viseli a versenyhelyzetet. Ha igazán próbára akarod tenni a személyiségét, ruházd fel hatalommal. *(Abraham Lincoln)*

Az ár számít – Az egyetlen dolog, amit nem tehetsz meg a nyílt forráskódú programokkal, az az egyeduralomra alapozott pénzszerzés. *(Jeremy Allison)*

Haiku-Foo – Minden apt-get minket erősít. Dist-upgrade-elj a nagy dicsőségért! *(Debian Haiku, írta Marc Merlin)*

Zárt forrású vita – Hosszú távú célkitűzésünk, hogy a lehető legtöbb eszközre Windows CE kerüljön, és szabvánnyá váljon. Nem muszáj, hogy nyereséges legyen az elkövetkezendő néhány évben. Az MS-DOS első kiadásán sem kerestünk pénzt. Ha tíz dolláros áron be tudsz törni erre a piacra, akkor rajta! *(Bill Gates)*

Miután megnyitottuk az InterBase forráskódját, az ügyfelek első kérdése az volt, hogy mennyibe kerül. A program leglényegesebb tulajdonsága a nyílt forráskód után az árcédula volt. *(Ted Shelton)*

Végre a nyílt forráskódnak hála, a cég fejlesztései nem halnak meg a céggel együtt. *(Deirdre Saoirse)*

A linuxos webkiszolgálókat működtetőknek lehetőségük nyílik-e arra, hogy építhessenek a .NET-re? Természetesen igen. Ez azonban nem azt jelenti, hogy a hosszú távú terveink nem arról szólnak, hogy ezek a kiszolgálók is Windowst futtassanak, de biztosítjuk a .NET használatának lehetőségét. *(Steve Ballmer)*

Én nem tartozom azok közé, akik szerint Bill Gates maga az ördög. Csupán az a gyanúm, hogy ha a Microsoft összeállna az ördöggel, nem lenne szüksége tolmácsra. *(Nick Petreley)*

Az igazság és az őszinteség kifizetődő? Ha az igazság és az igazság keresése között kell választanom, az utóbbit választom. *(Bernard Berenson)*

Ha elárulod az igazat, nem kell semmire sem emlékezned. (Mark Twain)

Mindig az őszinteség a legjobb megoldás. Ha színlelni tudod, nyerő vagy. *(George Burns)*

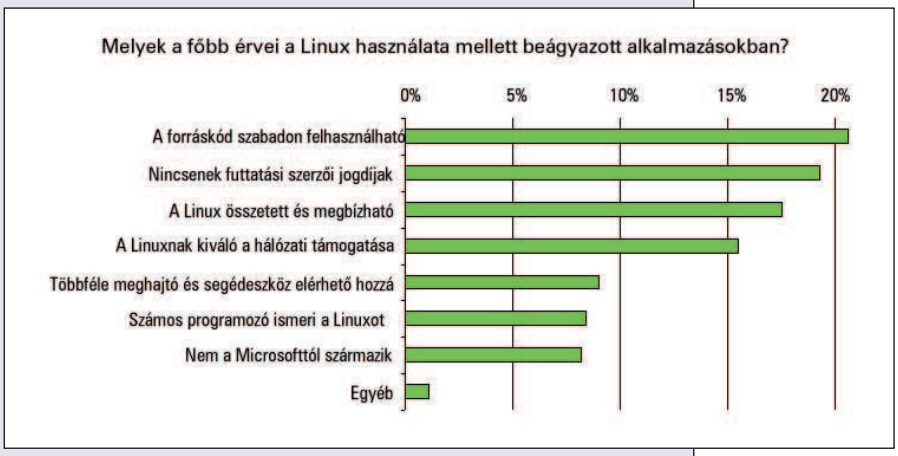
A csukott száj nem kel szárnyra. *(Simon Murcott)*

Szólásszabadság vagy ingyen sör

Mindjárt vágjunk is bele a közepébe. Az első kérdésre adott válaszok meglepő eredményt hoztak.
Miért akarnak a fejlesztők nyílt forráskódú programot és Linuxot használni a beágyazott rendszerekben?
Azt hihetnénk, a kérdésre adott kézenfekvő válasz: „mert szabadon használható". Nem így van!
Mit értékel a legtöbbre a nyílt forráskódban?
A fejlesztőktől itt azt kértük, hogy az alábbi lehetőségek közül válasszák ki az első-, a másod- és a harmadrangú érvet a nyílt forrású programoknak a beágyazott alkalmazásokban történő felhasználása mellett.

1. ábra A nyílt forráskóddal kapcsolatos fejlesztői lekérdezés eredménye

- Így az operációs rendszeren belül közvetlenül tudom növelni a szolgáltatásokat.
- Ez még akkor is biztonságot jelent, ha soha sincs rá szükség.
- Könnyebbé teszi az alkalmazás beállítását és a hibakeresést.
- Lehetővé teszi, hogy megértsem, mi zajlik az OS belsejében.
- Megteremti a lehetőséget az OS hibáinak azonnali kijavítására.
- Megszünteti az OS-kereskedőktől való függőséget.
- Az együttműködésen alapuló, nyílt forrású fejlesztési folyamat kiváló programot eredményez.
- Nincs szükségem rá, nem akarok nyílt forráskódot.
- Egyéb.



2. ábra Érvek a Linux használata mellett beágyazott alkalmazásokban

Minden kiválasztott állítást aszerint súlyoztunk, hogy a legfontosabb (5 pont), a második legfontosabb (3 pont), vagy a harmadik legfontosabb (1 pont) volt-e. Ezután az eredményeket összeszedtük és úgy egységesítettük, hogy a legtöbb pontot elért állítás egyes értékkel szerepeljen. Az eredmények az *1. ábrán* láthatók.
Ezek az eredmények több tekintetben elgondolkodtatók. Először is a felmérésben nem jut kifejezésre a programozók abbéli hajlandósága, hogy a forráskódba belenyúlva egyéni Linux-változatokat hozzanak létre. A fejlesztők inkább a forráskód birtoklását értékelik, ez ugyanis lehetővé teszi, hogy a szerzői jog által védett operációs rendszerek szállítóinak rabságából kiszabaduljanak. A forráskód birtoklása sokkal könnyebbé teszi azt is, hogy kitaláljuk, milyen folyamatok zajlanak a rendszer belsejében. Az „így módosítani tudom a programot" vagy az „így ki tudom javítani a hibákat" típusú választások tekintélyes számú szavazatot kaptak, de a dolgok áttekintő ábrájában ezek a sor végére kerültek.
Érdekes, hogy a sorrendben ez az állítás szerepel elsőként: „Az együttműködésen alapuló, nyílt forrású fejlesztési folyamat kiváló programot eredményez." Ebben az különösen jelentős, hogy a levédett programok készítői ügyviteli modelljeik alapvető módosítása nélkül ezt nem tudják utánozni, nagyon valószínűtlen lépés lenne a részükről.

Tavaly a ➲ LinuxDevices.com felmérést készített a fejlesztőkkel, hogy megpróbálja megérteni, mi készteti őket a Linux használatára a beágyazott rendszerekben és az értelmes készülékekben.

A Linux-rendszer választásának egyéb okai

A felmérés során arra kérték a fejlesztőket, hogy mondják meg, miért akarnak Linuxot használni a beágyazott alkalmazásokban. A válaszadókat itt megkérték, nevezzék meg az összes indokot, melyeket fontosnak tartanak az alábbiak közül:

- Nem kell fizetni a futtatásért.
- A forráskód rendelkezésre áll (szabad).
- Nem a Microsofttól származik.
- A Linuxnak kitűnő a hálózati támogatása.
- Több meghajtó és segédeszköz kapható.
- Sok programozó ismeri a Linuxot.
- A Linux összetett, illetve megbízható.
- Egyéb.

A *2. ábra* mutatja az eredményeket.

Szólásszabadság vagy ingyen sör

Az egyik különösen meglepő tanulság az, hogy a beépített eszközök magától értetődő költségérzékenysége ellenére, a Linux szólásszabadsága avagy a „szabad beleszólás” szempontja (vagyis a forráskód rendelkezésre állása) lassan háttérbe szorította az ingyen sört (vagyis az ingyenes használatot), mint a Linux beágyazásánál figyelembe veendő elsődleges szempontot.

A költségkövetkezményekbe jobban belegondolva feltettünk néhány kérdést a kiadásokkal kapcsolatban is.

Fizetne-e a linuxos fejlesztői, illetve támogatói szolgáltatásokért? Fizetne-e egységenkénti szerzői jogdíjat?

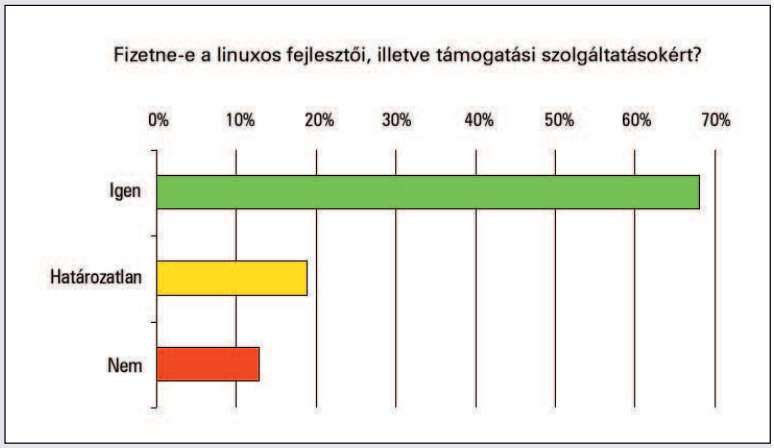
Az eredmények a *3.* és *4. ábrán* láthatók.

(Megjegyzés: a második kérdést csak utóbb tettük hozzá, így az itt látható eredmények egy viszonylag kicsinek tekinthető adatmintán alapulnak.)

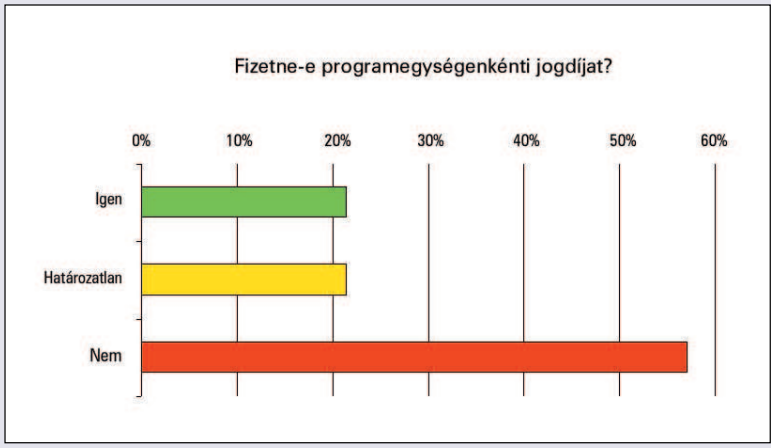
Az eredményekből leszűrhetjük, hogy míg a beágyazott programok fejlesztői valóban készek arra, hogy pénzt áldozzanak a külső szolgáltatásokra és a támogatásra, a Linux beágyazása esetén (68 százalék igent mondott és csak 13 százalék nemet) a számok átbillentek az ellentétes oldalra, amikor a kérdés az volt, hajlandók-e szerzői jogdíjat fizetni (58 százalék nem és csak 21 százalék igen).

Gyanítom, hogy ezeket az eredményeket néhány beágyazott Linuxot szállító és szolgáltató legalábbis „érdekesnek” fogja találni!

A beágyazott Linux piaci áttekintése a ➲ www.linuxdevices.com/cgi-bin/survey/survey.cgi webhelyen olvasható.



3. ábra Kedvező eredmény született!



4. ábra A számok átbillentek az ellentétes oldalra

Forrás: LinuxDevices.com áttekintés, 2000. december
➲ http://www.linuxdevices.com/polls/

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.

OpenOffice és KDE: a Linux arca

Polló László urat, a Portocom Számítástechnikai Részvénytársaság igazgatóját kerestük meg, hogy a Linuxhoz fűződő kapcsolatáról faggassuk.

Mészáros Ferenc: Úgy hírlik, hogy Önök gépeiket SuSE Linux operációs rendszerrel forgalmazzák.

***Polló László*:** Kétféle számítógépet forgalmazunk, a Portocom azonban alapvetően hordozható gépeiről híres, jelenleg is ezek teszik ki forgalmunk 98-99 százalékát. Ha linuxos gépet kérnek tőlünk, akkor a SuSE Linuxot ajánljuk a hordozható gépekre, sajnos azonban nem túl sokat adtunk el belőle. Készítettem erről egy gyors kimutatást: negyven kelt el, ebből egy Mandrake-kel és 39 SuSE Linuxszal. Azt azonban tudni kell, hogy a hordozható gépeinken Linuxot futtatók száma azért lényegesen több, mint ez a negyven. Az emberek ugyanis szeretik maguk föltelepíteni a saját Linux-rendszerüket. Akik jelenleg Linux-kedvelők, azok általában nem olyanok, akik egyszerűen csak használják a számítógépet, hanem szakemberek, akik mélyen érdeklődnek az operációs rendszerek iránt, ezért inkább maguk szeretik testre szabni azt. Becslésem szerint egyébként az eladott 15 ezer gépből több száz – azaz nagyjából két-három százalék – üzemel Linux alatt. Jelenleg talán az a legjobb hír, hogy működnek!

***M. F.:** A Linux egyre inkább kezd előtérbe kerülni több felületen is. Az Önök által forgalmazott egyedi kártyákhoz tudnak-e linuxos meghajtókat biztosítani?*

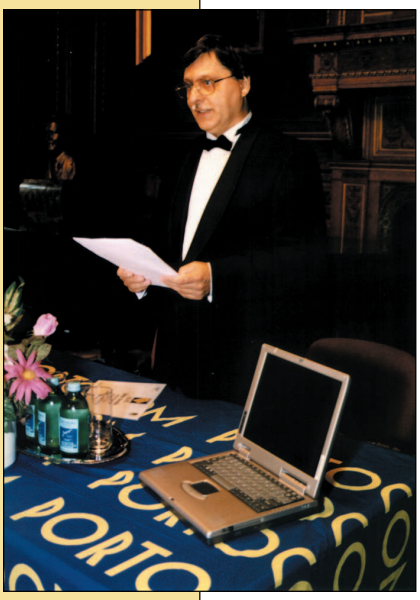
P. L.: Jelenleg a kártyagyártók elég ritkán adnak linuxos meghajtókat, leggyakrabban Microsoft-meghajtókat mellékelnek. Ugyanakkor kártyáink – miután mi Tajvanról vásárolunk – olyan lapkakészletekre épülnek, amelyek általánosan használatosak a világ összes részén, ebből kifolyólag minden gond nélkül futtathatók a legváltozatosabb linuxos környezetekben is. Tehát ilyen típusú összeférhetetlenségi hibákkal eddig még nem kellett megküzdenünk.

***M. F.:** Akkor a forgalmazóik, eladóik meg tudják mondani, hogy melyik az a kártya, amelyik megbízhatóan működik Linux alatt is?*

P. L.: Jelenleg ilyen vizsgálatokat nem végzünk, ugyanis nincs rá számottevő igény – a kereslet egyelőre még nem indokolja, hogy fejlesztőmérnökeink ezzel foglalkozzanak. Természetesen nagyobb megrendelés esetén megtesszük, ilyen volt például Portonet-rendszerünk kiépítése, mely asztali PC-kből áll; és saját magunk fejlesztettünk hozzá programokat, sőt meghajtókat is beszereztünk, illetve megfelelőségi vizsgálatokat is végeztünk. Egyik fejlesztőmérnökünk kimondottan ilyen típusú kérdésekkel foglalkozik. Meglévő kártyáink – ezek a szokásos faxmodem-, ethernetkártyák – gond nélkül működnek linuxos környezetben, tehát nem kellett ezzel még külön foglalkoznunk.

***M. F.:** Több munkagépet tartalmazó hálózatra egyéni megoldásuk van.*

P. L.: A Portonet hálózati számítógéprendszer létrehozása során az alapgondolatunk az volt, hogy nagyon olcsó és mégis jól üzemeltethető irodai környezetet készítünk Linux alá. Mi a következő megoldást választottuk: legyenek olyan munkaállomások, amelyekben a processzor, a memória nagyteljesítményű – tehát gyors gépek –, ugyanakkor tároljunk mindent egy központi gépen. Ez arra jó, hogy minden javítás, frissítés azonnal végrehajtódik az összes munkaállomáson, továbbá nagy sebességű 100 Mbit/mp-es ethernethálózattal összekötöttük ezeket a gépeket, hogy be lehessen tölteni egy boot-EPROM segítségével magát az operációs rendszert is, így arra tetszőleges programokat lehet letölteni. Nagymértékben kihasználjuk a munkaállomások erőforrásait, miközben a rendszergazdák „álmát" valósítjuk meg azzal, hogy mindent központilag tudunk felügyelni.



***M. F.:** Ez a megoldás mennyiben hasonlít a régi nagyszámítógépes rendszerekhez, ahol a központi géphez terminálokkal csatlakoztak a felhasználók?*

P. L.: A terminálok csak a megjelenítésre szolgáló eszközök voltak, a jelenlegi munkaállomások programok futtatására is alkalmasak, és ezeket a központi géphez kapcsoljuk. A régi hálózatokban először volt a terminál, amely csak karakterek és képpontok megjelenítésére volt alkalmas, aztán később jött az X-terminál, ez már grafikus felületek használatára is alkalmas volt. A mostani megoldás jóval többet tud elődeinél, mert processzorral és memóriával rendelkező gép, így programok futtatását is lehetővé teszi, ezáltal tehermentesíti a központi kiszolgálót. Ebben az elgondolásban már a beruházási költségeket is mérsékelni lehetett. Bár a munkaállomásból legfeljebb egy merevlemezt tudtunk kihagyni, ami nem olyan nagy megtakarítás. Ezzel szemben az operációs rendszeren nagyon sokat nyertünk, hiszen szabad Linux-változatokat használtunk, illetve nagyon olcsó kiszolgálóhoz való Linux-változattal is el tudtunk indulni. A munkaállomásokra gyakorlatilag csak a Linux-rendszermag került föl – ez ugyanis szabad felhasználású –, és erre telepítettük a StarOffice-t, illetve az OpenOffice-t. Az elején még az 5.1B-vel indultunk, majd az 5.2-es StarOffice-szal folytattuk. Az OpenOffice-nak jelenleg is zajlik a magyarítása, ezért még nem használjuk. A StarOffice nagyon olcsó rendszer és az irodai feladatokat önmagában is képes ellátni: a szövegszerkesztést, a táblázatkezelést, a rajzolást, a webböngészést, a különleges matematikai képletszerkesztést, az adatbázis-kezelést, és még ki tudja hány szolgáltatással bír – tehát a StarOffice rendkívül széles körben használható irodai csomag. Tudjuk, hogy a programcsomag úgy keletkezett, hogy a Sun megvásárolta a német StarDivision céget (a StarOffice fejlesztőjét), és hogy borsot törjön a Microsoft orra alá, szabaddá tette a programot. Én úgy gondoltam, hogy

a Portonet a Linuxszal és a StarOffice-szal hazánkban jó megoldás lesz, pontosan azért, mert a tőkeszegény, induló vadkeleti kapitalizmusban dolgozó ország számára kitörést jelenthet. Majd egy évig erőltettük ezt a rendszert, de sajnos nem könyvelhettünk el igazi sikereket.

M. F.: *A StarOffice-ról megjelentettek egy könyvet is, amit a cég munkatársai írtak. Kérem, nagy vonalakban ismertesse olvasóinkkal.*

P. L.: Megjelentettük a StarOffice kezelési kézikönyvet is, mely önmagában sikeressé vált. Már a második utánnyomás is fogyóban van, és hamarosan jön a harmadik kiadás. A sikerre való tekintettel a StarOffice alapkézikönyvhöz most a haladók számára is megjelentetünk egy kézikönyvet. Mindkettőt *Kenczler Mihály* írta. Ha a StarOffice-nak – OpenOffice-nak – elkészül a magyarítása, a kézikönyv már nem lesz annyira fontos, hiszen az első kötet azt a felhasználói réteget célozta meg, akik nem tudnak annyira angolul (gyakorlatilag a feliratok fordítását tartalmazta).
Már elkészült a második kézikönyv – egyelőre még csak – nyomdai előkészítés előtti változata. A főbb fejezetcímek: Fejlett dokumentumkezelés, Sceduler – feladatütemezések, Grafikák a StarOffice-ban, Internetkezelés – levelezés, böngészés, Adatkezelés, Programozás – hiszen a StarOffice-nak saját, önálló programozási nyelve van.

M. F.: *A StarOffice lefedi a Linux alatt azokat a lehetőségeket, amelyeket leginkább a Microsoft ural a Windows alatt, azaz az irodai felhasználás piaci területét?*

P. L.: Még egy dologról szót kell ejteni a Linuxszal kapcsolatban. A Windows hazánkban akkor vált igazán sikeressé, amikor a Microsoft – igen korán – magyarította irodai programcsomagját, az MS Office for Windowst. Azt gondolom, hogy ma hazánkban azért van a Microsoftnak ekkora (megközelítőleg 99%-os) piaci részesedése ezen a területen, mert ezt elsőként meglépte, jól tette, és így nem véletlenül érte el ezt a részesedést. Természetes, hogy a Linux a jelenlegi nagyjából egy-két százalékos piaci részesedéséről csak úgy léphet előre, hogyha ugyanezt a lépést megteszi. Mi tavaly már megtettük az első lépést: a KDE-nek (az egyik legkedveltebb kezelői felület) elkészítettük a profi fordítását, és ehhez szintén megjelentettünk egy kézikönyvet *KDE – a Linux arca* címmel. Így a felület legalább „magyarul beszél". A StarOffice-t sajnos nem tudtuk magyarítani, erre ugyanis nem volt energiánk. Azóta természetesen a KDE is továbblépett, és már nem mi fordítjuk, ezt szintén állami segítséggel fogja egy másik cég megtenni. Mostantól kezdve a KDE, illetve az OpenOffice magyarul fog beszélni, így a Linux-OpenOffice együttesnek megszűnik a nyelvi korlátok miatt kialakult versenyhátránya a Microsofttal szemben. Itthon – tudomásul kell venni – a lakosság nagyon kis százaléka beszél idegen nyelvet, és abban is bizonytalan, amit ismer, tehát a fordításra nagy az igény.

M. F.: *Abban, hogy a Linux és párhuzamosan az OpenOffice előnyre tegyen szert a Windowshoz képest, nagymértékben közreműködik a szoftverrendőrség is, hiszen egyre többen félnek a Microsoft termékeinek törvénytelen használatától. Így lehetőségük lesz megkerülni a gondot szabadon használható programmal.*

P. L.: Nem egyszerű folyamat! Azért hadd mondjam el, hogy nálunk, ahol negyvenen vagyunk, és mindenkinek van számítógépe, négy munkaállomáson nem tudtuk egyszerűen „kiirtani" a Microsoft Office-t, ezeken meg kellett hagyni törvényes felhasználási szerződésekkel, és mindössze 36 állomásra tudtuk csak a StarOffice-t telepíteni.

M. F.: *Ez azért nem rossz arány!*

P. L.: Azért váltottunk, mert kiszámoltuk, hogyha negyven munkaállomásra telepítjük a Microsoft Office-t – akár még egy korábbi, olcsóbb változatát –, az is több mint négymillió forintba kerülne a cég számára; a jelenlegi, legújabb változatok pedig még ennél is lényegesen drágábbak lennének. Úgy döntöttünk, hogy amilyen mértékben csak lehet, mérsékeljük az ezzel kapcsolatos kiadásainkat. Ugyanakkor a cégnél működő belső rendszer csak Microsoft-előtéttel rendelkezik, tehát nálunk a munkaállomásokon nincs Linux. Kényszerből Microsoft operációs rendszereket alkalmazunk, mert a Dataflex fejlesztőeszköznek, amit a saját programjaink fejlesztői használnak, jelenleg még nincs Linux alatt futtatható ügyfélváltozata. És addig, amíg nincs ilyen, a cégen belül nem tudunk teljes körűen áttérni.

M. F.: *Ez akár felhívás is lehetne vállalkozó szellemű linuxos fejlesztők számára, ha szabad lenne a Dataflex is!*

P. L.: A Dataflex nagyon hatékony fejlesztőkörnyezet, magam is éveken keresztül dolgoztam benne, ezért is választottam ezt. Azt gondolom, hogy az összes igazán komoly fejlesztőkörnyezet ugyanazzal a gonddal küszködik. Az elmúlt 15-20 év alatt fejlődtek ki a Microsoft világában, és történik ugyan némi vándorlás a Linux irányába, de az első lépések mindig a kiszolgálók felé történnek. A Dataflex alkalmazásfejlesztő környezetnek is létezik Linux alatt futtatható változata, az úgynevezett runtime-on futtatható változatok. Ezek vagy buta terminálokhoz, vagy Linuxot futtató gépekhez készültek, de nincsen olyan szép grafikus felületük, mint ami a Microsoft Windowshoz tartozik. Ez elsősorban a Windows rendkívül jelentős piaci részesedésére vezethető vissza. Addig, amíg a Linux igazán meg nem erősödik, addig nem is fogják elvégezni ezt a munkát, így ez „róka fogta csuka, csuka fogta róka" helyzet. Meg kell erősödnie a Linuxnak. Még egy saját tapasztalattal egészíteném ki a képet: a cégünknél természetesen van linuxos gép – a webkiszolgálónk ➲ http://www.portocom.hu.

M. F.: *Itt bizonyíthatta megbízhatóságát a Linux!*

P. L.: A NetWare nálunk hálózati kiszolgálóként üzemel, tehát fájl-, illetve nyomtatókiszolgáló egyben, és amikor telepítettük a webkiszolgáló alkalmazást is, bizonytalanná vált a működése. Ezért két külön gépre osztottuk szét a feladatokat, és azóta Linux alatt fut a hálózatkezelés, illetve meghagytuk a fájl- és nyomtató-kiszolgálást a NetWare-nek.

M. F.: *Szeretném, ha bemutatná olvasóinknak a Portocom kiszolgálóját.*

P. L.: Beindítottuk a Portonet-rendszert, amihez egy kiszolgáló, illetve ügyfél-munkaállomások tartoztak. A Portocom kiszolgálóra eddig csekély igény mutatkozott, úgyhogy egyelőre felfüggesztettük a gyártását. A Portonet ügyfelet változatlanul áruljuk, ez egy jópofa kinézetű gép, mindenféle Linux meghajtó tartozik hozzá, természetesen az összes alkatrész működik Linux alatt. A kiszolgálót változatlanul elő tudnánk állítani, de nem vagyunk igazán versenyképesek, tudniillik egyre több cég áll rá a Linuxra profi módon, számukra ez a fő irányvonal. Nem úgy, mint nálunk, hiszen ez számunkra csupán kiegészítés a hordozható gépek üzletága mellett.

M. F.: *Gépeik processzorai Intel Celeron 667 MHz-től, illetve Pentium III 733 MHz-től kezdve választhatók. Melyik a legkedveltebb gépük?*

P. L.: A FreeStar a legolcsóbb. Itthon általában az a jellemző, hogy mindig a legolcsóbb a legkelendőbb, ez nyilván összefügg azzal, hogy az embereknek mennyi pénzük jut erre az eszközre. A FreeStarból most tartottunk egy akciót, behozattunk kétszázat és már az akció meghirdetésének második napján elfogyott az összes. Amikor rendes áron kínáljuk, akkor kiegyensúlyozottabb a forgalom. A hordozható gépek honi piacára jellemző, hogy a gépek nyolcvan százaléka alacsony kategóriás, míg tíz-tíz százaléka középkategóriás és csúcsgép. A Portocom egyébként inkább a közép- és felsőkategóriában igazán versenyképes. Típusválasztékunk fele alacsony-, 25-25 százaléka középkategóriás és csúcsgép, ezenkívül egyedi katonai kiviteleket is forgalmazunk. A világ nyilván abba az irányba halad, hogy 1/3- 1/3- 1/3 arányban fogy majd e három kategória. A Föld más tájain a hordozható gépek értékesítésének kilencven százalékát meghatározó három területen – a Távol-Keleten, Észak-Amerikában, illetve Nyugat-Európában ez a jellemző. A hordozható gépek a technika csúcsteljesítményét képviselik, a pici kubatúrába mindent belezsúfoltak. A folyamat hasonlóképpen zajlik, mint régen a kis tranzisztoros rádiók és a nagy csöves világvevő rádiók között történt: előbb-utóbb a „tranzisztoros rádiók" kerekednek az asztali PC-k fölé. De ahogy ez a folyamat a rádióknál is igénybe vett néhány évtizedet, úgy a számítógépeknél is hasonlóra számíthatunk. A hordozható gépek már 13 éve piacon vannak…

Visszatérve a Linuxhoz, azt gondolom, hogy meg kellene őrizni a Linux Közösségnek azt a családias, baráti hangulatát, amit a fejlesztők egymás között kialakítottak. Jó volna, ha ez a terület nem válna annyira a verseny területévé, mint a számítástechnika sok más része. A világnak egy olyan különleges közösségéről van szó – fejlesztőkről és használókról egyaránt –, akiknek nagy része önzetlenül, ingyen végzi a munkáját. Talán ennek az egész internetes világnak ez az egyik legfontosabb üzenete. Valami olyan újfajta emberi közösségszervezést tesz lehetővé, ami világméretekben hat, és amire korábban még nem volt példa. Ezt meglepetéssel fogadták más nagy programfejlesztők. Nem gondolták volna, hogy sikerülhet.

M. F.: *Mit jósol a Linux világának?*

Ha ez a hangulat megmarad, ha a kezdő, egyszerű felhasználói kérdésre továbbra sem gúnyos megjegyzések érkeznek vagy a hivatalos álláspont – hogy menjenek ehhez és ehhez a céghez, ahol ennyi és ennyi forintért meg fogják kapni a megfelelően vizsgázott szakmérnöktől a megfelelő választ, hanem megmarad egymás önzetlen segítése, akkor a Linuxot semmi sem tudja megállítani.

Mészáros Ferenc (francz@linuxvilag.hu) programozómatematikusként több cégnél gépközeli programozással foglalkozott, jelenleg a Linuxvilág munkatársaként dolgozik. Szereti a társaságot, a táncot; lányával szívesen vitorlázik, lovagol és kirándul.

III. GNU/Linux Szakmai Konferencia

A Linux-felhasználók Magyarországi Egyesülete 2001. szeptember 15-én (szombaton) tartja III. szakmai tanácskozását a Közép-Európai Egyetem (CEU) Konferencia Központjában, neves előadók részvételével. A rendezvény díszvendége *Richard M. Stallmann* (USA), a szabad programok guruja, a GNU atyja és a Free Software Foundation alapítója.

A tanácskozás területén kiállítás és vásár is helyet kap. A központi témák a szabad forráskódú programok, valamint a GNU/Linux lesz.

Az érdeklődők a következő témakörökben hallgathatnak előadásokat:
– GNU/Linux-rendszerfelügyelet,
– biztonság,
– programozás GNU/Linux-rendszereken,
– multimédia és grafika,
– magyarítás,
– GNU/Linux a cégeknél.

A tanácskozás napján a helyszínen sajtótájékoztatót is tartunk Richard M. Stallmann részvételével, ezért a szakújságírókat is szeretettel várjuk. Előzetes bejelentkezést a
➲ konf2001@lme.linux.hu címen fogadunk.
Minden érdeklődőt szeretettel várunk!

Nyomtatáspróba Brother HL-1670N nyomtatóval

A különféle eszközök Linux alatti kipróbálása, életre keltése mindig is kihívást jelentett a velük foglalkozó emberek számára. Mivel az alkatrészekhez, eszközökhöz általában nem mellékelnek linuxos meghajtóprogramot, életre lehelésük sokkal összetettebb feladat. Szerkesztőségünkben most egy Brother HL-1670N-es nyomtatóval próbálkozhattunk meg. Már kinézetre is nagyon komoly nyomtatónak látszik, kezelőfelülete egyszerű, emellett minden szolgáltatása gyorsan elérhető. Az LCD kijelző három színének köszönhetően könnyedén nyomon követhetjük a nyomtató állapotát is. A beépített biztonsági nyomtatásnak köszönhetően csak akkor nyomtatja ki az általunk titkosként megjelölt dokumentumokat, ha megadtuk a megfelelő jelszót. Ha a nyomtató hiba miatt leáll (papírelakadás, festékkifogyás), levélben értesíti a felhasználók megadott csoportját. Ez a nyomtató minden olyan tulajdonsággal rendelkezik, amire egy kis- és közepes méretű irodának szüksége lehet. A nyomtató beállításához a CUPS (Common UNIX Printing System ➲ http://www.cups.org) nyomtató meghajtóit használtuk. Ennek segítségével könnyedén szóra bírtuk, párhuzamos és USB kapun keresztül is, sajnos azonban a windowsos ügyfélgépeknek nem sikerült Sambán keresztül nyomtatniuk, valószínűleg a windowsos és a linuxos meghajtók összeférhetetlensége miatt. Viszont amikor a hálózati illesztőn keresztül kötöttük be, minden ügyfél tökéletesen tudott nyomtatni. Teljesítménye és adatai magukért beszélnek:

- 16 lap/perc nyomtatási sebesség,
- 1200 DPI felbontás (a szürkeárnyalatos képek is élvezhető minőségben jelentek meg),
- párhuzamos, USB-s és ethernetcsatlakozási lehetőség,
- beépített kétoldalas nyomtatási lehetőség,
- 16–144 MB-ig bővíthető memória,
- beépített webkiszolgálón keresztül minden operációs rendszerből kezelhetjük a nyomtatót.

Adatok
Brother Hungary
telefon: 06-1-382-74-53
e-mail: info@brother.hu
➲ www.brother.hu

Csontos Gyula

Szemet póbáló próba

A BaSys Magyarország Kft. jóvoltából az egyik hagyományos CRT monitor egy hét szabadságot kapott szerkesztőségünkben. Helyét a Neovo cég 17 hüvelykes TFT aktív mátrix monitora foglalta el. Szokatlanul kis helyigényének és „tűéles" képének köszönhetően azonnal belopta magát mindenki szívébe.
Az ajánlott felbontás az 1280x1024 képpont 60 Hz frissítéssel, ez azonban ne tévesszen meg senkit, mivel az LCD monitorok nem a szokásos módon frissítik a képet, ezért nem is villognak, ellentétben a hagyományos CRT monitorokkal (ezeknél minél gyorsabb a képfrissítés, annál kevésbé károsítják villogásukkal az emberi szemet).
A látható felület mérete megfelel egy 19 hüvelykes monitorénak. Az önműködő képernyőbeállítás minden VGA kártyával kiválóan működött, pár másodperces várakozás után a teljes hasznos felületet betöltő éles képet kaptunk. Áttekinthető és könnyen kezelhető menürendszerének köszönhetően a különböző egyéni beállítások – mint például a színhőmérséklet kiválasztása – sem jelentettek gondot. A család tagjai között találhatók 15, 17 és 18 hüvelykes képernyők, akár érintésérzékeny felülettel. A különleges képernyőbevonatnak köszönhetően tükröződésmentes felületen dolgozhatunk. A képernyő üvegfelülete pedig különösen ellenálló a karcolásokkal és egyéb sérülésekkel szemben. Ha valakinek a pénztárcája megengedi, mindenképpen érdemes megismerkednie ezzel a monitorcsaláddal! (Rendszergazdáknak ajánljuk az alábbi szöveget: „Főnök, értse meg, szükségünk van lapos monitorra, hiszen a gépteremben olyan kevés a hely, és nagy monitor kell, mivel olyan sok gépet kell egyszerre felügyelni…")

- Felbontás: 1280x1024,
- analóg, digitális és videobemenetek találhatók a készülék hátoldalán,
- 16,7 millió szín kezelésére képes,
- könnyen falra szerelhető,
- energiagazdálkodása megfelel az EPA, a NUTEK és a VESA DPMS előírásoknak,
- mérete 436x426x200 mm,
- tömege: 7,5 kg.

Adatok
BaSys Magyarország Kft.
telefon: 06-23-415-541
e-mail: basys@mail.basys.hu
➲ www.basys.hu

Csontos Gyula
(Csontos.Gyula@linuxvilag.hu),
a Linuxvilág hír- és CD-szerkesztője, valamint a www.linuxvilag.hu tartalomfelelőse.

3Com – a hálók tetején

Júniusban sajtótájékoztatót tartott a 3Com Hungary Kft. a dunai Fortuna állóhajón. Négy új termék kapcsán rendeztek bemutatót: három kapcsolóelosztó és egy hálózatfelügyelő programmal ismerkedhettünk meg. *Lehner Tamás* ügyvezető igazgató bevezetőjéből megtudtuk, hogy a 3Com cég a megelőző negyedévihez képest világviszonylatban 15 százalékos növekedést könyvelhetett el dollárban. Mindezt annak ellenére, hogy beszüntették a soros vonali útválasztók gyártását, így mintegy húsz százalékkal szűkítették termékkínálatukat, valamint az év folyamán csaknem harminc százalékos árcsökkenést vezettek be. Így is 110 százalékkal növelték az eladott kapuk számát! „A 3Com módosította partnerválasztását, jobban figyel a felhasználói igényekre, és ennek megfelelően alakítja át termékkínálatát, illetve változtatja a termékek árát is" – mondta az ügyvezető. „Ha egy termék jó, még nem jelenti azt, hogy csillagászati összegeket kell kifizetni érte. Mivel Magyarországon is befejeződtek azok a zöldmezős beruházások, amelyek támogatásával kialakították a hálózati háttérrendszert, most már mindenkinek van valamilyen létező hálózata, amire fejleszteni szeretne. Trade-up nevű programunk partnereink és ügyfeleink számára biztosítja annak kiszámolását, hogy a már meglévő hálózati eszközeik beszámításával az új eszközök árából hány százaléknyi kedvezményt adunk vásárlóinknak. Ez a kedvezmény nemcsak a 3Com termékekre, hanem bármely működő eszközre vonatkozik. Partnereinket ellátjuk mintaprogrammal, hogy ennek alapján e szolgáltatásunkat jobban megismerhessék. Emellett elindult a *3Com university* program is, amely partnereink és felhasználóink tudását és 3Com jártasságát hivatott növelni. Jelenleg hazánkban 55-en szerezték meg a Certifyed Solution Associated vizsgát (a 3Com kereskedelmi szintű vizsgája) a Weben keresztül. Ezenkívül tervezzük még egy módszertani vizsga bevezetését is. A teljesség igénye nélkül hadd említsek néhány sikeres hazai telepítést. Ilyen volt az Országos Nyugdíjfolyósító Intézetben kiépített hálózat, amelyet a Conet Kft. valósított meg 100-150 végpont kialakításával. A Synergon Rt. az Antenna Hungária Rt. új épületében sikeresen megoldotta a teljes hálózat telepítését, gigabites sebességű gerinccel, valamint 3300-as eszközökkel. A Data Contact Kft. a Magyar Könyvklub számára épített ki hálózatot. A Deffend Kft. pedig egy pályázaton nyert, ennek értelmében a nyáron 600 végponttal, hibatűrő hálózati szerkezettel, aktív eszközökkel valósítja meg a PSZÁF épületében a hálózatot. Tavaly volt egy bírósági pályázat, amit az AlbaComp nyert meg, így a megyei bíróságok hálózatba kötését is 3Com eszközökkel valósítják meg. A VPOP (Vám és Pénzügyőrség Országos Parancsnoksága) új határátkelőhelyeit is mi láttuk el

Lehrner Tamás a 3Com ügyvezető igazgatója

Készüléktípusok/ Tulajdonságok	***SuperStack 3 Switch 4900***	***SuperStack 3 Switch 4400***	***SuperStack 3 Switch 4300***	***SuperStack 3 Switch 4005 4005***
gazdag szolgáltatások	+	+	+	+
drótsebesség	+	+	+	+
toronyba építhető (darab)	-	8	-	-
felügyelt	+	+	+	+
10/100 Mb/mp sebességű kapu	-	24 fix	48 fix	max. 96
1000 Mb/mp kapu (darab)	12 fix* + 0-4	0-2	0-4	max. 12
rétegtámogatás	L3	L4**	L2	L2-4
3Com NBX támogatás	+	+	+	+
átviteli sebesség (Mp/mp)	47	6,6	13	18
sávszélesség (Gb/mp)	32***	16	34	48
modulok száma	12 / 4	2/16	2	12
TX10/100 kapuk száma max.	-	24 fix/192****	48 fix	96
FX kapuk száma max.	-	2	4	96
T kapuk száma max.	4	2	4	12
SX kapuk száma max.	4	2	4	12
LX kapuk száma max.	4	2	2	-
GBIC kapuk száma max.	4	-	-	12
listaár	4995 dollár	1750 dollár	2995 dollár	9995 dollár*****

* 100/1000 Mb/mp kapu, ** Toronyba építhető, *** 16 Gb/mp teljesen kétirányú,
**** Teljesen kétirányú, ***** 40 kapu TX alapkiépítésben

eszközökkel. Az Országos Állategészségügyi Információs Rendszer a keretén belül a Megyei Állategészségügyi Központokat kötjük össze a ComNetwork Rt. segítségével, webalapú telefonrendszerrel, ez országosan 25 telefonközpontot és 650 telefonállomást jelent. A telephelyek között ISDN- és analóg vonalak teremtik meg a kapcsolatot, míg a telefonálás VoIP-n (Voice over IP – beszédátvitel IP-protokollon) valósul meg, amire a 3Com NBX telefonközpontját választották ki."

Majd *Szabó Gábor* vette át a szót, aki idén június elseje óta a 3Com kelet-európai marketingigazgatója. Tőle tudtuk meg, hogy a már meglévő CommWorksön túl két új részleget állított csatasorba a 3Com: a BCC- és a BNC- részlegeket. A BCC a Business Connectivity Company rövidítése, ez a részleg készíti a hálózati csatlakozókat (Network Interface Connector – NIC), a PC-kártyákat és PCMCIA-kártyákat, valamint minden olyan eszközt, amelyek PC-oldalról biztosítják a hálózathoz való kapcsolódást. Míg a BNC-részleg (Business Networks Company) felel a hálózati eszközökért, a hálózati kapcsolókért, a belső hálózatok telefonos és vezetéknélküli eszközeiért. A CommWorks a szolgáltató piacot célozza meg, számukra biztosít hálózati megoldásokat.

„Cégünk nem adta el gyárait, a létszámleépítés háromezer embert érintett, így a cégen belül tudtuk megoldani a piac által kikényszerített változtatásokat. (A Cisco például két lépésben 35 százalékos munkaerő-leépítést hajtott végre, míg az IT-piaci visszaesés következtében mostanáig több mint 409 ezer az elbocsátottak száma.) A 3Com célul tűzte ki a helyi hálózatok piacán az elsőség megszerzését, efelé jó úton halad. Erre utalnak pénzügyi kimutatásaink is: a 3Com piacvezető a rézalapú gigabites kapcsolók tekintetében, 35,7 százalékos részesedést tud a magáénak az eladott kapuk számának tükrében. Piacvezető a hálózati telefonrendszerek területén 74 százalékkal, és szintén első a hálózati csatlakozók és a PC-kártyák piacán 48 százalékos részaránnyal. Itt jegyezném meg, hogy több mint kétezer könyvecskét küldtünk szét az országban cégvezetők, vállalatigazgatók számára a belső hálózatokban használható telefonrendszerek ismertetése céljából. Kedvező volt a válasz, hiszen érdeklődnek, beruházásokat terveznek, csöng a telefon, és tán a belső hálózatok telefonpiaca is megindulhat hazánkban."

Majd *Lehner Tamás* szakmai vezető vette vissza a szót, hogy bemutassa a *Piranha* termékcsaládot, mely négy új elemből áll. Közülük három hálózati kapcsoló, a negyedik pedig hálózatfelügyelő program, amelyet a kisebb cégek is szabadon használhatnak, mivel közzétették honlapjukon a ➲ http://www.3com.com/tns címen (lásd még keretes írásunkat!). E program érdekessége, hogy a felhasználók igényeire való tekintettel nem csak a 3Com termékek felügyelete kapott helyet a programban. Ilyenek például a Cisco-útválasztók, ezek igen elterjedtek és a 3Com új Network Supervisor Version 3 programja alatt is felügyelhetők.

A 3300-as sorozat – mely toronyba építhető (stackable) 10/100-as eszközcsalád volt – kiváltására jelent meg a SuperStack 3 Switch 4400-as család, amely szintén toronyba építhető. Két fő különbség van közöttük: az

3Com Network Supervisor Version 3

A 3Com Network Supervisor Version 3 nagy hatékonyságú és könnyen használható alkalmazás: lehetővé teszi a hálózat felfedezését, valamint a topológia és a terhelés figyelését. Továbbá megkönnyíti a hálózat rendszerbe állítását és csúcsteljesítményen való üzemeltetését is.

Egyszerű használat és felügyelet
A program önműködően felfedezi a hálózati kapcsolatokat és az IP-eszközöket, azaz a hálózati eszközöket, a munkaállomásokat és a kiszolgálókat. Felismeri a hibásan beállított eszközöket és optimalizálási lehetőségeket kínál fel. A hálózat felfedezése után önműködően feltérképezi annak szerkezetét, és grafikusan ábrázolja az eszközöket és kapcsolatokat. A térkép segítségével a hálózatfelügyelő könnyen figyelheti a hálózaton lévő terhelést; küszöbértékeket és riasztásokat állíthat be, láthatja a hálózat eseményeit, saját maga tervezte formátumban kimutatásokat készíthet, és készülékkonfiguráló eszközöket indíthat el.

Könnyebben telepíthető az intelligens hálózat
A 3Com Network Supervisor segítségével egyszerűbb beállítani a SuperStack 3 Switch 4400 egész hálózatra érvényes forgalom-rangsorolását. A Felhasználó Varázsló segítségével azonosíthatja a kulcsfontosságú hálózati alkalmazásokat és kiszolgálókat. A 3Com NBX telefóniarendszerek beszédforgalma önműködően elsőbbséget kap.

Intelligens eseménykezelés
A 3Com Network Supervisor figyeli a hálózaton fellépő szokatlan eseményeket és naplózza a hálózat fontos tevékenységeit. Az események szűrhetők, annotálhatók, csoportosíthatók és rendezhetők. Az intelligens eseményfigyelő motor az ismétlődő eseményeket bezárható csomópontokba gyűjti, összeveti az összefüggő eseményeket és önműködően kiszűri a kevésbé fontosakat. Bármely hálózati eseményhez rendelhető riasztás is, így a felhasználó előre figyelmeztethető a hálózat lehetséges hibáira. A riasztások eljuttathatók elektronikus üzenetben, érkezhetnek hangjelzésként vagy beugró üzenetablakkal, vagy pedig elindíthatnak egy külső alkalmazást, illetve parancsfájlt (script). Az SMS-üzenetet és a személyhívó használatát is támogatja.

Frissítés egyetlen kattintással
A 3Com Network Supervisor egyetlen kattintással csatlakoztatható a 3Com webhelyére, ahonnan letöltheti a legújabb változatokat és az új berendezéseket támogató programokat. Így a 3Com Network Supervisor folyamatosan frissíthető. A javítócsomagok és más frissítések letöltését egyszerű kezelőfelület segíti.

A 3Com Network Supervisor (korábbi nevén Transcend Network Supervisor) elérhető a ➲ http://www.3com.com/tns címen, és a minden 3Com termékkel együtt szállított CD-k is tartalmazzák majd.

egyik az, hogy nagyobb lett a gyűjtősín (back-plane) teherbírása; az előbbi 3,3 millió csomag/másodpercről 6,6 millióra változott, így teljes mértékben drótsebességű (wirespeed), gátolatlan (non-blocking), késleltetés nélküli csomagtovábbításra alkalmas eszköz. Továbbá nőtt a tornyozhatósága is, négy helyett nyolc egységet lehet összekapcsolni, amivel az egy toronyba helyezhető kapuszám százról kétszázra növekedett. Sokat beszélünk mostanában a QoS-ről (Quality of Service – Biztosított minőségű szolgáltatás), a 4400-as család négyes rétegű (4layer), amely alkalmazásszinten már a csomagok szűrésére, rangsorolására és forgalomirányítására is képes. Például meghatározható, hogy torlódás esetén a beszédalapú VoIP-csomagok előnyt élvezzenek a nem ilyen elsőbbségű csomagokkal szemben, vagy egy pénzügyi központban az SAP-forgalom előnyben részesíthető a HTTP-forgalommal szemben. Ebből következik, hogy fontossági sorrend alakítható ki a csomagok között. Sőt, csomagok háttérbe is szoríthatók, mindössze azt kell engedélyezni, amit fontosnak tart a felhasználó, amit viszont nem, azt ki is tilthatja a hálózatából, ugyanis a hálózati kapcsolókban szűrhető a forgalom. Így könnyen megoldható, hogy NetBIOS üzenetek ne rohangáljanak – csak a TCP/IP-alapú hálózatban.

A hálózatfelügyelet kifogástalan megvalósítása az, amikor bármilyen felületről vezérelhetők az eszközök. Lehetőség nyílik webes felületről, parancssoros felületről és a 3Com Network Supervisor felügyeleti programból – ennek sajnos nincs még linuxos változata – beállítani és felügyelni eszközeinket. Külcsínre ugyanolyan a 4400-as, mint a 3300-as: elöl 24, 10/100-as kapu van, hátul viszont nem egy, hanem két bővítőhely található. Ide lehet csatlakoztatni a gigabit-, illetve a tornyozó (stacking) modulokat.

Egy másik újdonság a SuperStack 3 Switch 4300-as, amit a felhasználók nem kimondottan alkalmazás-irányultságú csoportjának szántunk. Jelenleg hálózatépítés során a leggyakrabban kapcsolókat, valamint 10/100-as kapukat használnak, itt a jókora teljesítmény és a nagy kapusűrűség mindenképpen érdekes lehet. Ezért a 3Com új 4300-asa egy 48 kapus, 10/100-as kapcsoló, hátul két bővítő modullal, azaz legfeljebb 4 gigabit uplink (külvilág felé irányuló) teljesítménnyel. Gátolatlan, késleltetés nélküli, drótsebességű, 13 millió csomag/másodperc átvitelű eszköz. Egy 100-as kapu forgalmának átviteléhez 144 ezer csomag/másodperc sebesség szükséges, így a 48 kapcsolathoz 6,9 millió, a másik fennmaradó 6 millió csomag/másodperc pedig a négy gigabites kaput tudja ellátni. Tehát teljesítményben ténylegesen nyújtani tudja az azonnaliságot, a késleltetés-mentességet, és kiszolgálja a megnövekedett hálózati igényt.

A SuperStack 3 Switch 4005-ös egy chassis-alapú hálózati kapcsoló, amelybe gigabit ethernet, 10/100-as kapuk, illetve százas optikai kapu csatlakoztatható két tápegységgel. Hibatűrő ellátást tesz lehetővé a nagy üzembiztonság elérésére, amihez használat közben cserélhető (hot swappable) I/O modulok és tápegységek tartoznak – egy esetleges meghibásodást követő minél kisebb üzemidőkiesés eléréséhez. Ez hármas réteg képességű, tehát a hálózatok központjában alkalmazható eszköz. Teljes kiépítésében 12 gigabites kaput, vagy 80 (96) 10/100-as, illetve százas ethernetkaput tud a hálózatban kezelni. Mindezt fele annyiért (tízezer dollár alatt) kínálja a 3Com, mint a vetélytársai. Teljesítményben 18 millió csomag/másodperc átviteli sebességet tud – mindezt hármas réteg képessége mellett, drótsebességen. A 3Com élettartam-jótállást vállal mindhárom hálózati kapcsolóra, ami annyit jelent, hogy a készüléktípus a forgalomból való kikerülését követő öt évig teljes körű szavatosságot vállal, és azt követően még szállít alkatrészt a készülékhez. A piaci folyamatok alapján úgy gondoljuk a 3Comnál, hogy érdemes alacsony áron adni az eszközöket, hiszen két éve még a 3300-as került háromezer dollárba, ma 1700 dollár a listaára, ami alig valamivel több, mint a fele az akkorinak. A 3Com a baseline-osztályt szorítja valamivel lejjebb az által, hogy a 3300-as lesz a legkisebb hálózati kapcsoló a palettán, míg a 4400-as a négyes réteg kezelhetőséggel, illetve a 4300-as a kettes réteg kezelhetőséggel, de nagyobb kapusűrűséggel fog a helyére lépni. A 4005-ös a maga hármasréteg-képességével már a nagyobb központok kapcsolója lesz, nem beszélve arról, hogy több mint 80 optikai fastethernetes vonalat tud egyetlen dobozban összefogni, ami világszerte a kormányzati körök igénye.

Újságírói kérdésre válaszolva *Lehner Tamás* elmondta még, hogy a 3Com nem pusztán eszközöket, hanem megoldásokat kíván forgalmazni, ezért tette szabaddá 3Com Network Supervisor Version 3-as programját. Az alapkiépítés annak idején legfeljebb ötszáz végpont vezérlésére alkalmas program volt, kimondottan kis- és közepes méretű vállalkozások számára, ami aztán nőtte ki magát egy 1000-1500 végpont kezelésére képes programmá. Ha az ügyfél vásárolt egy vagy két 3300-ast, akkor erre a 24–48 kapura nem fog még kétezer vagy tízezer dollárt kiadni, mivel a kapcsolója már így is kétezer dollárba került. Viszont igénye lenne arra, hogy az eszközeit megfelelően tudja kezelni. És ha ezt az igényt a 3Com ki tudja elégíteni azzal, hogy szabaddá teszi a felügyeleti programját, miért ne tenné, ha pénzügyileg is megéri, és a szolgáltatás bővítésével értékesebb lesz az, amit nyújt. A program Windows 9x-re, vagy 2000-re telepíthető rendszer, unixos változata egyelőre nincs.

A témával kapcsolatos további adatok a ➲ http://www.3com.hu címen olvashatók.

Mészáros Ferenc (francz@linuxvilag.hu) programozómatematikusként több cégnél gépközeli programozással foglalkozott, jelenleg a Linuxvilág munkatársaként dolgozik. Szereti a társaságot, a táncot; lányával szívesen vitorlázik, lovagol és kirándul.

Postás a gépben: a SuSE Email Server II

A SuSE Email Server II mindent tud, amit egy jó postás: külső és belső levelezésre is kiválóan alkalmas, ráadásul nem ismer határokat.

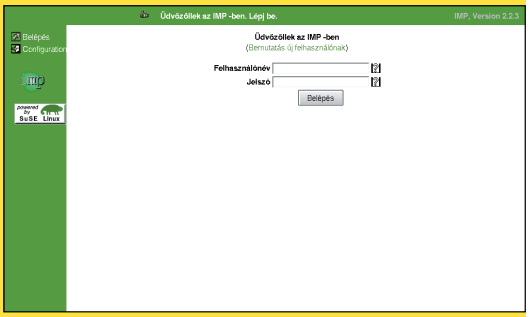
Gombamód szaporodnak az országban a 20-25 fő körüli „legénységgel” dolgozó vállalkozások. Ezek a cégek azt szeretnék, ha meglévő vagy leendő állandó internetkapcsolatukat nemcsak netezésre, esetleg azon lévő webkiszolgálójuk elérhetőségére tudnák használni, hanem levelezésre is. Nézzük, milyen lehetőségeik voltak eddig:

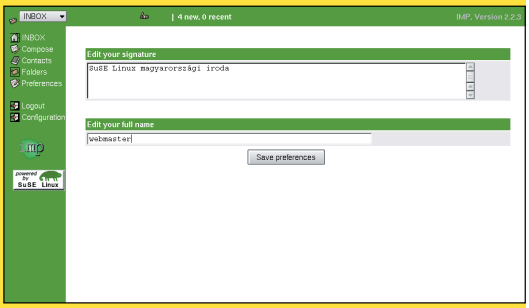
- használhatták a szolgáltató által nyújtott cegnev@szolgaltato.hu postafiókot, esetleg kaphattak még 1-2 kacifántos címet;
- pár szolgáltató – borsos összegért – vállalta, hogy a cegnev.hu tartománybejegyzéshez tartozó levelezőszolgáltatást is ellátja. Ez a megoldás viszont egy új felhasználó hozzáadása vagy egy elfelejtett jelszó lecserélése esetében igen körülményessé válna;
- ha jó rendszergazdájuk volt, megszerezte az ehhez szükséges programokat, illetve a szükséges tudást.

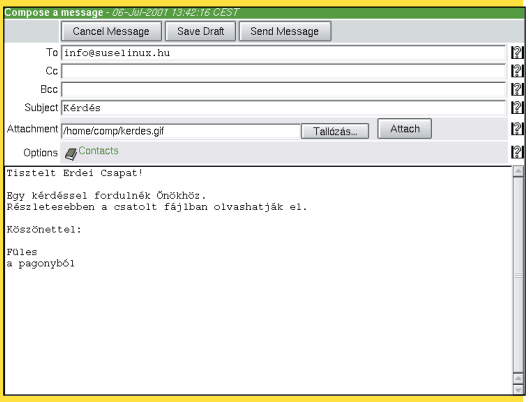
Írásunk a Linuxszal most ismerkedő rendszergazdáknak mutat be egy újabb választási lehetőséget. Szerkesztőségünkbe két doboz érkezett: egy nagyobb, amely az IBM Magyarország Kft. (➲ http://www.ibm.hu) által a rendelkezésünkre bocsátott szép fekete tornyot (a kiszolgálót), a billentyűzetet és az ugyanolyan színű egeret rejtette. A kisebbik doboz a SuSE Magyarországi Irodájától (➲ http://www.suselinux.hu) érkezett zöld „SuSE eMail Server II” felirattal. A tartalma a következő volt:

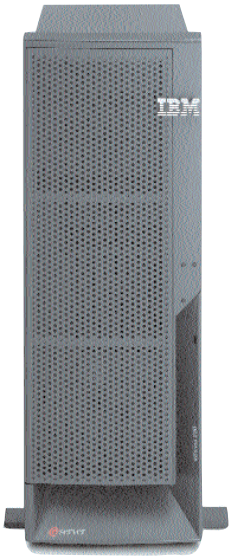
- SuSE Linux vállalati csomag (angol nyelvű alapbeállítással),
- eMail II kiszolgálóprogram (szintén angol nyelvű alapbeállítással),
- az összes program forráskódja,
- rendszerindító lemez,
- 60 napos ingyenes telepítési segítség,
- kétéves operációs rendszerváltozat-követés,
- kézikönyv a vállalati programcsomaghoz,
- kézikönyv a levelező kiszolgálóhoz.

Az első kézikönyvből linuxos tudásunkat felfrissítve a SuSE-változattal ismerkedhetünk meg; a másodikból a levelezőkiszolgáló telepítését, beállítását és használatát, valamint az Arkeia Backup alkalmazását sajátíthatjuk el. A Fekete Toronyon azaz az IBM xSeries 220-on múlott, hogy milyen hévvel közelíthetek Geekóhoz. Az erőt sejtető torony azonban első ránézésre nem sokat árult el magáról: egy CD-ROM és hat menetközben cserélhető lemezkeret virított a nyitható ajtó mögött, ebből három volt „élesítve” egy-egy 9 GB-os Quantum SCSI merevlemezzel. Az oldallap eltávolítása után egy Intel Pentium III 933 MHz processzorral találtam szemben magam – ennél a darabnál sajnos csak az egyik foglalatot használták, a négy memóriafoglalat is csak részben volt foglalt egy 128 MB-os SDRAM ECC RDIMM modullal, pedig legnagyobb befogadó képessége 4 GB lenne. Az alaplap Intel lapkái között még az integrált S3 Savage4 áramkörei lapultak meg. A merevlemezek kezeléséről egy Ultra160 SCSI (64-bit integrated single channel) vezérlő gondoskodik. Lássuk, mit tud a SuSE Linux üzleti e-mail megoldása! A SuSE eMail Server II a cég jól bevált IMAP kiszolgálójának legújabb változata, mellyel lehetőség nyílik az osztott postafiókok használatára (ez csoportmunkáknál, illetve projektek megvalósításakor lehet a segítségünkre). A SuSE eMail kiszolgáló nyílt forráskódú megoldás, mely



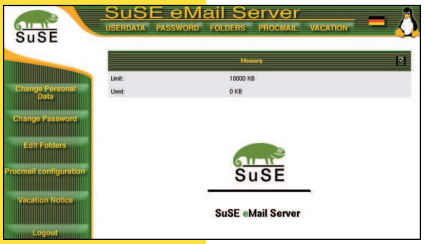


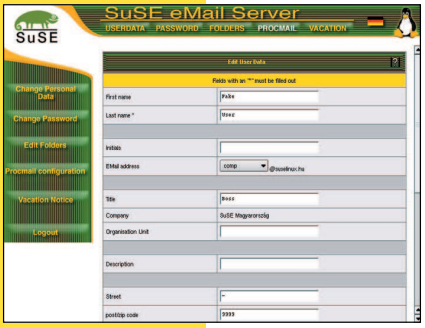


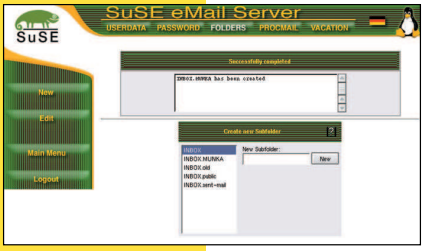


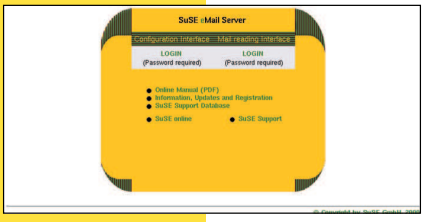
bevált és megbízható elemeken alapul. Ilyenek például: SMTP, IMAP4, POP3, LDAP, TLS, SASL, SMTP-AUTH, hogy csak néhányat emeljünk ki. Ezeket a SuSE Linux Enterprise vállalati csomag alapjaira építették és kifeje-

zetten a levélkezelésre hegyeztek ki. Mind a helyi hálózaton, mind az Interneten keresztül támogatja a levélküldést és -fogadást, így kínálva megoldást a vállalat teljes külső és belső levelezőrendszerére. A Linux nagy teljesítőképességének és hatékony erőforrás-kihasználásának köszönhetően a SuSE Linux eMail Server kiválóan alkalmazható a most kipróbált vasnál erősebb környezetben is.
A termék nagy előnye, hogy a hasonló rendszerekkel ellentétben nem korlátozza a levélfelhasználók számát felhasználási szerződéssel, azaz teljesítőképességének határai csak az alá pakolt gép kiépítésétől függnek. Nincs ügyfélmegkötés sem: minden ismert ügyfélprogrammal és szabványprotokollal (POP3, IMAP) együttműködik. Másik kedvező szolgáltatása a webmail (tetszőleges internetböngészővel felhasználóink kényelmesen kezelhetik leveleiket), így akkor is levelezhetünk, ha levelezőprogramunk nincs kéznél, vagy más gépét kényszerülünk használni.
Leendő levelezőkiszolgálónk gyorsan és könnyen telepíthető; ezt a YaST2, a SuSE alapértelmezett grafikus felügyeleti felülete biztosítja. Ha még nem telepítettünk SuSE Linuxot, észre sem vehető, mikor fejeződik be az operációs rendszer telepítése, és mikor kezdődik a levelező beállítása. A biztonságot a kódolt csatornák használata (SSL) és a tanúsítványalapú (CA) levéltovábbítás jelenti. A meglévő rendszerekkel való együttműködés az elfogadott Unix fetchmail, vacation, illetve procmail szabvány átjárhatóságán keresztül valósul meg. A webes felületen keresztül a cég szakemberei nemcsak leveleink kezelését oldották meg, hanem a felügyeletet is. A Horde könnyű és kényelmes kezelhetőséget biztosít.
Felhasználóként LDAP kiszolgáló(k)hoz csatlakozhatunk és lekérhető(k) a külső postafiók(ok); leveleinket könnyen megírhatjuk, olvashatjuk, módosíthatjuk személyes beállításainkat. A felügyelő könnyedén tudja kezelni a felhasználókat, a démonokat (FetchMail, Postfix, Apache) és a levélsorokat (mail queue).
Ha már átrágtuk magunkat a kézikönyveken, kezdjük el a telepítést! Az első, rendszerindításra is képes CD a legtöbb esetben jól használható, ha valami oknál fogva ez nem működne, használjuk a mellékelt hajlékonylemezt. Következő lépésként a Yast2-vel találjuk szemben magunkat, amely szép sorban kikérdez bennünket a rendszer beállításairól. A billentyűzet kiosztásával és a telepítés nyelvével kezdi, utóbbit használja utána a rendszer nyelveként is. A magyar nyelv beállítása sajnálatos módon nem tartott sokáig; telepítés után a webmail bejelentkező oldalán még találni magyar szavakat, de rövidesen visszavált angolra és a nyelvet a kijelentkezés után is megtartja. Értesüléseim szerint a SuSE Magyarországi Iroda tervezi a Horde IMP felületének magyarítását, remélhetőleg ez a következő kiadásba már belekerül.
A nehezebb kérdés annak eldöntése, hogy milyen lemezszerkezetet használjunk. Könnyelmű voltam és a SCSI RAID adta lehetőségeket nem használtam ki – ezt egy éles rendszeren saját adataink védelme érdekében azért érdemes használni! Az egyik merevlemezt a `/` könyvtár számára jelöltem ki, a másik kettőn a `/home` és a `/var` könyvtár osztozott. A következő kérdések a rendszerindításra, a rendszergazdai felhasználó jelszavára és a hálózati beállításokra vonatkoznak. A tartománybeállítások után a tulajdonképpeni eMail Server beállítása következik. Nagyon gyors lesz: két képernyő közül az elsőn az LDAP-beállításokat, a következőn pedig cégünk nevét és nemzeti hovatartozásunkat kérdezi meg.
A rendszer feltelepüléséhez viszont mintegy 30-40 percnyi idő szükséges, amit vasunk „keménysége” azonban nagymértékben befolyásolhat!
A megfelelő programok kiválasztása utánra maradt a legnehezebb feladatunk: az adott körülményekhez el kell végezni a finomhangolást. A Horde nyújtotta felület nagyon jól átlátható menürendszert ad, melyet már ismereteinkhez mérten is tudunk használni, például ha jártasak vagyunk a Postfixben (*lásd még 66. oldalon*), bátran igénybe vehetjük a *Postfix (Expert)* menüpontot. Ettől a lépéstől kezdve a gépet átadhatjuk a rendszerfelügyelőnek, hogy munkatársaink között kiossza az emailcímeket.
A webmail rész is jól vizsgázott, a próbafelhasználók ugyanolyan könnyedséggel levelezhettek, mint bármely ingyenes szolgáltató webfelületén – csak a magyar szavak hiányozhattak, de az a pár szükséges angol kifejezés gyorsan megtanulható. A levelezőügyfelet használók pedig csak azt veszik észre, hogy leveleikhez gyorsabban jutnak hozzá.
Ha felhasználóink eddig még nem leveleztek, ránk vár egy kis feladat: be kell őket vezetni az elektronikus levelezés rejtelmeibe.





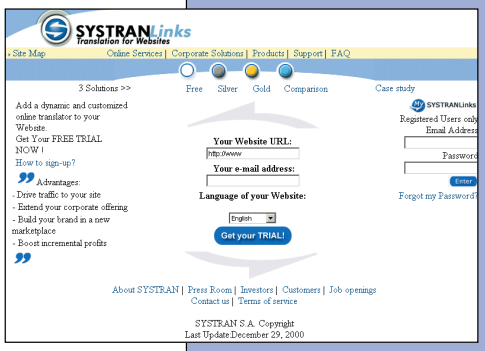




Fricska Sándor (fricska.sandor@linuxvilag.hu) rendszergazdaként dolgozik. Szabadidejének egy részét a természetben tölti (pingvinekkel és hüllőkkel), fennmaradó idejében pedig az LME aktív tagja.

Linuxos feljlesztések a számítógépes fordításhoz

A Systran Internet Translation Technologies a hidegháború alatt született, amikor az Egyesült Államok kormánya nagy mennyiségű orosz szöveget akart gyorsan lefordíttatni. A hatvanas évek végén magánkézbe került a cég és Systranra módosították a nevét, székhelyéül pedig a kaliforniai La Jollát választották.



A kilencvenes években a Systrannál úgy döntöttek, hogy kidobják az MVS-t futtató OS/390-et és az egész rendszert átültetik Unixra. Ekkorra a PC-k már elegendő teljesítménnyel rendelkeztek ahhoz, hogy futtassák a fordítómotorokat. Az Assembly-kód nagy részét önműködő program segítségével fordítottuk le C-re. Elsőként Solarisra ültettük át, de hamarosan olcsóbb gépekre váltottunk, így jött a PC és a Slackware kombinációja (azóta már RedHatet használunk). A Linux választásakor a következő szempontokat vettük figyelembe: többféleféle géptípuson is futtatható legyen; minden programot elérhessenek a fejlesztők, amire csak szükségük lehet; a természetes nyelvek feldolgozása nagyméretű szöveges állományok kezelésével jár, ehhez pedig hatékony programok szükségesek; a fordítómotor összetett szabályrendszert használ, ezért a kód átültetése nagy C, illetve C++ programokat eredményez, ehhez szükség van az olyan hatékony eszközökre, mint a gcc, illetve a g++, valamint a GNU make; a széles közönséget ellátó, például az AltaVistához hasonló méretű ügyfelek számára a legfontosabb az alkalmazás biztonsága, illetve a rendszer megbízhatósága; és ne feledkezzünk el arról sem, hogy minden felhasználó költségkímélő megoldást kíván.

Ezekhez még az is hozzáadódik, hogy az új alkatrészekhez a meghajtóprogramok leggyorsabban Linuxra jelennek meg, továbbá más rendszerekhez képest sokkal kisebb az erőforrásigénye. A Linux beállításai egységesek és könnyen másolhatók további gépekre, valamint a rendszer igen jól méretezhető. A Linuxhoz tűzfal, sendmail, Apache, mod_perl és PostgreSQL is tartozik, ezeket mind használják a cég webes szolgáltatásaihoz (➲ http://www.systranlinks.com/, ➲ http://www.systranet.com/). Emellett a KDE és a GNOME lehetővé teszi, hogy a cég nem programozó alkalmazottai szintén használhassák a Linuxot. Ez azért olyan fontos, mert a Systrannál sokan inkább nyelvészek, mint programozók. Végül a POSIX-szabványhoz való igazodás lehetővé teszi, hogy igény esetén könnyen átültethessük a rendszert más Unixokra is.

A Systran programjai találhatók meg világszerte a legtöbb önműködő fordítórendszer mögött. Ügyfeleink közé nem csak az Egyesült Államok kormányügynökségei sorolhatók, de olyan cégek is, mint az AltaVista, a Microsoft, az Apple, a Lycos vagy az AOL.

A számítógépes fordítás a nyelvészet és a számítástechnika határterülete. A termék fejlesztése valójában az emberi nyelv szabályainak gépi nyelvre (kódra) történő átültetését jelenti. A nehézségek nagy része nyelvészeti, mivel először pontosan le kell írni a kérdéses nyelveket. A folyamat a következőképpen zajlik: elemezni kell a kiindulási nyelvet, majd leírni, azután létre kell hozni a célnyelvet. A kódkészítés négy részre bontható: 1. a kiindulási nyelv elemzése; 2. a célnyelven történő összeállítás; 3. az átviteli szabályok; és 4. a fordítómotorok által közösen használt eljárások (például tárkezelés, parancssor-értelmezés, szótárazó eljárások, szűrők, elő- és utófeldolgozók stb.).

A cégnél különleges szótárakat alkalmazunk. Ezekben nemcsak az adott kifejezés fordítása található meg (pl.: manger = to eat), hanem a kapcsolódó mondattani és lexikai adatok is (például: „ez az ige tárgyas, ebben az adott szövegkörnyezetben ezt és ezt jelenti"). Három különböző szótárat használunk. Az első kettő úgynevezett belső, az egyik az egyszerű szótöveket tartalmazza, a másik az összetett szavakat és kifejezéseket. A harmadik a „külső" szótár. Ez utóbbit az adott ügyfél igényeinek megfelelően állítjuk össze, meghatározott témakörökben.

A Systran olyan segédfájlokkal is rendelkezik, amelyek az igék, a főnevek és a melléknevek ragozásával kapcsolatos szabályokat írják le (ezek természetesen nyelvfüggők), továbbá meghatározzák az elsőbbségi szabályokat az ügyfél szótára számára, valamint megadják szövegsablonokat. Ezt mind C-ben valósítottuk meg, de az újabb programegységeket általában már C++-ban készítjük.

A szabályok előállításához a nyelvészek GTK-alapú grafikus programot használnak. Az adatokat ASCII fájlokban tárolják, ezek egy Perl programon folynak át, így készülnek el az adatokból a kód számára értelmezhető makróutasítások. A szótárak félig önműködő módon épülnek fel, azon szabályok felhasználásával, amelyeket a nyelv elemzése közben állítottunk fel. Először minden nyelvhez egynyelvű mesterszótár készül. A nyelvi rész feltöltése után a Systran-rendszer táblázatok alapján önműködően lényeges nyelvi adatokat ad a bejegyzésekhez. Például az „automatically" angol szót a program határozószónak ismeri fel, mert -ally végződésű. Ezután hozzák létre a kétnyelvű szótárakat, egyszerű kétbejegyzéses listaként, amely a lekérdezésnek megfelelő szintaktikai adatokat az egynyelvű mesterszótárból kéri le.

A szótárakat csak a legvégső lépésnél fordítjuk le bináris alakra, ezzel gyorsabbá tehető a szótárt használó program. Amikor az AltaVista weblapján valaki rákattint a *Translate* gombra, nem is gondol bele, milyen összetett folyamat áll a fordítás mögött.

A Systran tervei között szerepel egy szabad linuxos változat kiadása, amely tudását tekintve megegyezik a Systran Personal windowsos kiadásával.

Thunus F.
A Systran Luxembourg igazgatója

Csúcsfejek a tökfejek ellen

Ez egy jó év a vállalkozók számára. A kutya ott van elásva, hogy a befektetők nem a vállalkozásokat támogatják, hanem a valódi szakmai tudás nélküli bábokat.

Dave Winer

Linux-index 2001. június–július

1. Azon levelezőprogramok száma – a Microsoft Outlook Express kivételével –, amelyekre veszélyes volt az Anna Kournikova vírus: 0
2. Azon – Anna Kournikova vírust tartalmazó – levelek száma, amelyeket a Linux Journal egyik szerkesztője kapott 2001. február 11-én: 15
3. Az Eazel Inc. (linuxos grafikus felületet készítő cég) által 2001 márciusában elbocsátott munkatársak száma (becsült érték): 40 fő
4. SuSE-alkalmazottak száma (becsült érték): 600 fő
5. A 2001 májusáig megjelent Linux-változatok száma: 188
5. Beágyazott Linux operációs rendszerek száma tavaly augusztusban: 17
7. Beágyazott Linux operációs rendszerek száma idén: 25
8. A vékony ügyfelek eladásának növekedése 1999 folyamán világszerte: 90 százalék
9. A Fortune 100 cégek közül a vékony ügyfeleket alkalmazó cégek százaléka: 80
10. Bejelentett elbocsátások száma az Amazon.com-nál 2001. januárban: 1300 fő
11. Az Amazon elbocsátott alkalmazottai számára létrehozott bizalmi letétként kezelt pénzalap mennyisége tavaly januárban: 2,5 millió dollár
12. Az internetfelhasználók aránya 2001. februárban az USA-ban: 60 százalék
13. Az Internetre bejelentkezők száma 2001. februárjában az Egyesült Államokban: 169 millió
14. Az internetfelhasználók táborából a hálózatot munkára használók 2001 februárjában az Egyesült Államokban: 14 százalék
15. Vezeték nélkül továbbított üzenetek száma 2004 végére: havonta 244 milliárd
16. Átlagos felhasználó gépén található MP3-ak száma az Egyesült Államokban: 500
17. Az Iridium nevű műholdalapú telefonrendszer fejlesztési költsége milliárdokban: 5
18. Az Iridium ennyi millióért kelt el azután, hogy csődbe ment: 25
19. Azon Iridum-műholdak száma, amelyeket el kellett volna égetni a Föld légkörében, ha nem találnak rájuk vevőt: 60
20. A Napster ennyi milliárd dollárt ajánlott – visszautasították – azért, hogy szerzői jogokkal védett anyagokat is terjeszthessenek: 1
21. Az Egyesült Államokban 90 perc zene egy felhasználóhoz való sugárzásának becsült költsége: 81 dollár naponta
22. Ugyanennek a költsége feliratkozott felhasználótól felhasználóig történő terjesztés esetén: 15 dollár naponta
23. Az ezer legnépszerűbb webhely látogatása a weblapletöltések ennyi százalékát tette ki 2000 júniusában: 53
24. Az ezer legnépszerűbb webhely látogatása a weblapletöltések ennyi százalékát tette ki 2001 januárjában: 48
25. PDA-eladások 2000-ben (millió dollár): 9,39
26. Tervezett PDA-eladások 2004-ben (millió dollár): 33,7
27. A Sharp ekkora piaci részesedés elérésére számít új Linux-alapú PDA-i révén: 50 százalék
28. A Sharp terveiben szereplő értékesítés végösszege 2002 végére (millió dollár): 1
29. A Sharp szerint ennyi Java-alapú alkalmazás létezik majd Linux-alapú PDA-jukra 2002 októberében: 10 000
30. A Sharp ennyi főre becsüli a linuxos PDA-t tevékenyen programozók számát: 100 000
31. A Sharp ennyi főre becsüli a Microsoft PDA-t tevékenyen programozók számát: 50 000
32. Azon francia iskolák száma, melyek részt vesznek a nemzeti Linux-programban: 350

Forrás:
1., 13–16.: ➲ News.com
2.: *Doc Searls*
3–4.: *Michael Hasenstein*
5–7.: echWeb
8.: International Data Corp.
9.: Giga Information Group
10–11.: Moneybox
16.: ➲ Eric Garland of BigChampagne.com, a The Standardból
17–19: Hoovers
20–22: ZDNet
23–24: Industry Standard from Alexa Internet, 2001. március
25–26: Gartner Group
27–31: CNET
32.: LinuxWorld

Tuti Tipp – Böngéssz gyorsabban!

Ha takarékoskodni szeretnél a sávszélességgel és ezáltal növelni szeretnéd a böngészés sebességét, telepíts a gépedre olyan névkiszolgálót, amely csak gyorstáraz. Nyugi, nem kell BIND-ot telepítened. Töltsd le a pdnsd-t a következő webhelyről: ➲ http://home.t-online.de/home/Moestl/.
A pdnsd megjegyzi a névfeloldás eredményeit, tárolja azokat a merevlemezen, így a gyakran látogatott webhelyek címét nem kell újra és újra lekérdezni. A programot kifejezetten telefonon keresztül csatlakozó felhasználók számára írták. Beállítása pofonegyszerű. A pdnsd felhasználói szerződése a GPL alá esik. A RedHat- a SuSE- és a Debian-rendszerekre egyaránt léteznek hozzá csomagok.

Választás a vállalati informatikában: a Linux

Különlegesebb területeken is alkalmaznak Linuxot: beléptetőrendszerük olyan egy vagy több linuxos gépből áll, amelyek akár több ezer kártyás, PIN-kódos, ujjlenyomat-olvasós helyet képesek vezérelni.

Érdekes megfigyelni, milyen sok feladatra alkalmazható a Linux. Akár minden nap találkozhatunk vele anélkül, hogy tudnánk, mit is használunk. Olyan alkalmazási területeket igyekszünk most bemutatni, amelyek ugyan ismertek a nyilvánosság előtt, eddig azonban sehol sem hallhattunk arról, hogy ezek linuxos megoldások.

Integrity Kft.

A 2000. évi középiskolai felvételi rendszer internetes adatgyűjtő és feldolgozó kiszolgálóoldali moduljait SuSE Linux kiszolgálókon futtatta az Integrity Kft. Három független internetkapcsolattal rendelkező kiszolgáló fogadta a több mint ezer iskolából érkező adatokat, valamint egy Java-alapú adatbázis-alkalmazást futtató kiszolgáló dolgozta fel a 400 ezer rekordnyi adatot. Mindegyiken Linux futott, a gépek IBM Netfinity 3000-es és 3500-as kiszolgálók voltak. A rendszer hibátlanul teljesített, ezért – a tervek szerint – a jövőben is Linuxon fog futni.

Másik feladatuk az INI aldomén-átirányítási és regisztrációs kiszolgáló kezelése. A legnépszerűbb ilyen hazai alkalmazás fő kiszolgálója egy IBM Netfinity 5500-as gép 512 MB RAM-mal. Ezen a gépen – Linux alatt – Java-alkalmazások és SQL adatbázis-kezelő fut. A Netfinity 5500-as magas rendelkezésre állást biztosító megoldásait a Linux a menet közben cserélhető (hotswap) PCI síneket leszámítva támogatja, idén azonban várhatóan ez a támogatás is megjelenik.

Egy mostanában indult projekt keretében közérdekű – elsősorban államigazgatási vonatkozású – adatokat tartalmazó honlapok és adatbázis-alkalmazások indulnak linuxos kiszolgálókon. A sorozat első tagja a ➲ www.kozjegyzo.hu, további tagjai a közeljövőben jelennek majd meg. A terv az, hogy az Interneten át el lehessen érni a közjegyzők, a hivatalok, a bíróságok, valamint az ügyvédek adatait.

Általában nem túl erős gépeket alkalmaznak, főként IBM Netfinity 3000-es, 3500-as darabokat, a legerősebb pedig a fent említett Netfinity 5500-as. A feladatokat inkább több kisebb gépre igyekeznek szétosztani, ezzel is növelve az üzembiztosságot. Adatbázis- és alkalmazáskiszolgálási célokra szinte kizárólag Linuxot használnak.

Az Integrity több mint tíz Linux-alapú gépe mint webes alkalmazáskiszolgáló, adatbázis-kezelő, valamint mint internetkiszolgáló működik. Tűzfal, proxykiszolgáló, fájlkiszolgáló és munkaállomás céljára is alkalmaznak Linuxot a következő programokkal: adatbázis-kezelés – PostgreSQL, Interbase, MySQL; fájlkiszolgáló – SaMBa, NFS; proxykiszolgáló – Squid; tűzfal – TIS; levelezési lista – mailman; webkiszolgáló – Apache (Roxen Challenger csak kipróbálásra), Deneb (saját fejlesztésű, Javában készült). Vannak saját fejlesztésű alkalmazásaik is, melyeket Javában, Perlben, PHP-ben és C-ben írtak.

Különlegesebb területeken is alkalmaznak Linuxot: beléptetőrendszerük olyan egy vagy több linuxos gépből áll, amelyek akár több ezer kártyás, PIN-kódos, ujjlenyomat-olvasós helyet képesek vezérelni. A Linuxot megbízhatósága és kis gépigénye következtében választották. A programot C nyelven (vezérlés) és PHP-ban (webfelület) írták. Folyamatban vannak különböző mérő és kapcsoló fejlesztések, melyeket szintén Linux-rendszerre alapoztak.

Prím Communications & Média Rt.

Öt kiszolgálót üzemeltetnek, melyek közül a legkisebb egy Pentium II 300 MHz-es processzorral, 256 MB RAM-mal, SCSI merevlemezzel; a legnagyobb pedig két Pentium III 800 MHz-es processzort, 512 MB RAM-ot és 70 GB SCSI merevlemezt tartalmaz. Ez utóbbiak Intel-alapú vezérlő segítségével gépszinten RAID5-be vannak kötve.

Minden kiszolgálón Debian GNU/Linux fut és a következőket nyújtja:

- *Internetes szolgáltatások*
  - dinamikus, adatbázis-alapú hírrendszer cikkadatbázissal és kapcsolódásokkal,
  - webmail típusú levelezőrendszer (PrímPosta),
  - weboldal-készítő (weboldal.com),
  - fórum,
  - „internetes” mappa képek és más fájlok tárolására a kiszolgálójukon,
  - egyéb adatbázis-alapú szolgáltatások,
  - Adserver (hirdetések kiszolgálását végző program),
  - pop3 és levelezőkiszolgáló,
  - DNS-kiszolgáló,
  - rendszerfelügyelet.
- *A cégirodában*
  - IP-álcázás (masquerading),
  - proxykiszolgáló,
  - levelezőkiszolgáló.

A fentiek megoldására a következő főbb programokat alkalmazzák: exim, BIND, Apache, PostgreSQL és PHP4. A biztonsági mentéseket az Amanda nevű programmal végzik.

A felhasználók számát a webes szolgáltatás miatt nehéz pontosan megmondani, de a Prim-rendszerben feliratkozott felhasználók száma meghaladja a nyolcvanezret, a PrimPostát pedig több mint tízezer ember használja rendszeresen.

Az összes PHP parancsfájl saját fejlesztésű, amelyekhez a Linux kiváló fejlesztőkörnyezetet biztosított. Mivel internetes kiszolgálókról van szó, már a munka legelején világossá vált, hogy a kívánt célok eléréséhez a Linux megfelelő eszközöket nyújt. Az elsődleges cél nem szabad, hanem használható rendszer fejlesztése volt.

Média 6 Rádió Szeged

Két telephelyet (Szeged és Szentes) köt össze két SuSE Linux, melyek 166 MHz-es Pentium processzorral, 32 MB RAM-mal és 2 GB-os merevlemezzel felszerelt számítógépeken futnak. Mindkét gépen található webkiszolgáló – Apache, levelezőkiszolgáló – qmail, FTP-kiszolgáló – proftpd, és SSH a távoli eléréshez. A két telephely között a biztonságos adatforgalom megteremtéséhez VPN-t alakítottak ki. Szegeden a linuxos kiszolgálónak körülbelül negyven felhasználója van, Szentesen pedig tíz. A gépek másfél éve üzemelnek, ez idő alatt legfel-

jebb tízórányi kiesés volt tapasztalható, ám ennyi is csak a rendszermag-frissítések, illetve a kapcsolati hibák miatt keletkezett.
Az üzenetrögzítő feladatát szintén SuSE Linuxra osztották. Ebben a gépben egy 133 MHz-es Pentium processzor ketyeg, 32 MB RAM-mal és 2 GB merevlemezzel megtámogatva. Erre a feladatra csak programból megvalósítható megoldást nem találtak, ezért a következőt választották: a vgetty call programjához készítettek egy parancsfájlt, ami a hangkártyán (GUS MAX-on) lejátssza az üdvözlő üzenetet, majd felvételre kapcsol és harminc másodpercig rögzít. A modem kimenete a hangkártya bemenetére van rákötve, a hangkártya kimenete pedig egy olyan áramkörhöz csatlakozik, amelynek ha a bemenetén jel található, akkor egy relé segítségével párhuzamosan trafón keresztül kapcsolódik a telefonvonalra. Ezáltal sugárzási minőségű üzenetrögzítőt sikerült készíteniük. A rögzített üzenetek a SaMBa fájlkiszolgálón keresztül érhetők el a kívánságműsor szerkesztője számára. Ezekről az üzenetekről biztonsági mentés nem készül, miután adásba mentek, törlik azokat.

Pécsi Tudományegyetem, Állam- és Jogtudományi Kar

A karon öt linuxos kiszolgálót üzemeltetnek.

1. Nyilvános webkiszolgáló:
– Celeron 366 MHz, 256 MB RAM, 2x15 GB merevlemez, 3C905B hálózati kártya,
– Renegát I (RedHat 6.1-alapú Linux),
– Apache + stunnel,
– wu-ftpd.
2. Álcázó és útválasztó-kiszolgáló:
– Pentium 75 MHz, 32 MB RAM, 2x210 MB SCSI merevlemez, EEPRO-100, 3c905B és SMC-EZ hálózati kártyák, szekrény-kivitel (3U magas 19"),
– RedHat 6.2-alapú Linux,
– ipchains + ipportfwd + FWTK csomag,
– BIND (DNS-kiszolgáló).
3. CD-torony és nyomtató-kiszolgáló:
– Celeron 366 MHz, 128 MB RAM, 2x8 GB merevlemez, 3C905B hálózati kártya, öt nyolcszoros Sony SCSI CD-ROM,
– Renegát II (RedHat 6.1-alapú Linux),
– SaMBa kiszolgáló,
– NFS, YP-ügyfél.
4. CD-torony, HTTP proxykiszolgáló és X-alkalmazáskiszolgáló:
– Celeron 366 MHz, 128 MB RAM, 2x15 GB IDE merevlemez, 3C905B hálózati kártya, négy nyolcszoros Sony SCSI CD-ROM, két negyvenszeres Plextor SCSI CD-ROM, valamint Sun DAT,
– Renegát II (RedHat 6.1 alapú Linux),
– xdm X-kiszolgáló,
– SaMBa kiszolgáló,
– Squid proxykiszolgáló,
– NFS, YP-ügyfél.

Grafikus alkalmazások:
– qvwm, fvwm és kde ablakkezelők,
– Star Office 5.1,
– Netscape,
– IRC-ügyfelek,
– teljes jogtár (Kerszöv fejlesztésű).
5. Levelezőkiszolgáló és fájlkiszolgáló:
– Pentium II 300 MHz, 256 MB RAM, 2x15 GB IDE merevlemez, 3C905B és SMC-EZ hálózati kártya,
– Renegát II (RedHat 6.1-alapú Linux),
– sendmail,
– Apache + stunnel (külső elérésre),
– SSH,
– NFS, YP-kiszolgáló;
– SaMBa kiszolgáló,
– APC UPS kezelés (APC hálózati kiszolgáló).

➲ www.jpte.hu

A hálózat kiépítése a következő: a kiszolgálók között 100 Mbit/mp UTP-hálózat található (3Com Superstack jelelosztón keresztül). Az ügyfelek felé többszörös csillagpont elrendezésű hálózat vezet 100 Mbit/mp gerinc és 10 Mbit/mp végponti sebességgel. Az internetelérés üvegszálon keresztül történik, 10 Mbit/mp sebességgel. Az Internet továbbszolgáltatása egy mikrohullámú hídon keresztül 2 Mbit/mp sebességgel, valamint négy 56 Kbit/mp sebességű analóg modemen keresztül valósul meg.
Az ügyfelek:
– nyolcvan Windows 95 és 98, Windows 3.11 az irodákban,
– harminc Windows 98 és Linux-ügyfél a gépteremben,
– 12 X-terminál (NCD, Sun, DEC, vt2000 gyártmányok vegyesen) külön gépteremben,
– tíz vt320 soros terminál,
– valamint i386 terminálkiszolgáló – ez a folyosói levelezőrendszer.

A fentieket kiszolgáló személyzetként három fő üzemelteti:

egy rendszergazda és két operátor. A felhasználók száma 1470, ebből körülbelül kétszázan vannak tevékenyen Linuxszal dolgozók, a többi SaMBa-, illetve levelező felhasználó. Rövidtávú fejlesztési terveik az alábbiak:

- *Gépek*
  - A belső kiszolgálók és a tűzfal cseréje szekrénykivitelű hibatűrő gépre,
  - a modemes behívókiszolgáló bővítése 12 darab 56 Kbit/mp sebességű modemre.
- *Programok*
  - Kettős kapu és protokoll tűzfalrendszer,
  - SQL-alapú naplózási rendszer PostgreSQL és syslog segítségével.

Index.hu Informatikai Rt.

A ➲http://index.hu/ címen található webkiszolgáló tulajdonképpen négy gépből áll. Négy Compaq dl360-asból alkottak telepet, ezek mindegyike a következő alkatrészekből épül fel: két Pentium III-as, 800 MHz-es processzor, 512 MB RAM és gigabites hálózati kártya. Merevlemez egyik gépben sincs! Operációs rendszerüket és a szükséges programokat a fájlkiszolgáló megosztott meghajtóiról töltik be.

➲ http://www.osb.hu/biblio

A fájlkiszolgáló Compaq ml530-as gép, Pentium III-as 933 MHz-es Xeon processzorral, 1 giga RAM-mal, nyolc 18 GB-os merevlemezzel és gigabites hálózati kártyával. A merevlemezek megoszlása a következő: 2x18 GB tükrözve az operációs rendszernek és 6x18 GB RAID5-be kötve az adatoknak. A gép NFS-en keresztül szolgálja ki a webes felületet biztosító többi egységet. Egy adatbáziskezelő-kiszolgáló is szerepel a „csapatban": egy Intel sitka – két 550 MHz-es Xeon processzorral, 1 GB RAM-mal, valamint öt 9 gigás merevlemezzel. A használt adatbázisok mérete is tekintélyes: körülbelül 4 giga egy-egy tábla mérete, a rekordszám pedig hárommillió körül van. Külön érdekesség, hogy ez rácáfol arra a hiedelemre, miszerint Intel-felületen nem lehet 2 gigabájtnál nagyobb fájlokat létrehozni. Létezik megoldás erre a feladatra is!
A telephez visszatérve: tudomásuk szerint Magyarországon az egyetlen fellelhető Cisco localdirectort ők használják. Ez terheléselosztást végez a telep négy gépe között. A beérkező kéréseket attól függően osztja ki, hogy megy-e az adott webkiszolgáló, és mekkora a pillanatnyi terhelése, illetve a válaszideje. Így a gépenkénti terhelés elvileg a legmegfelelőbb, és egy gép kiesése nem befolyásolja számottevően a rendszert működését. Amennyiben egy gép valamilyen okból kifolyólag kiesne a telepből, pillanatok alatt kicserélhető, és egy hajlékonylemez segítségével már el is indulhat a fájlkiszolgálóról.
A webes felület előállítására és kiszolgálására Apache-ot használnak PHP, Perl, MySQL és Oracle-kiegészítésekkel. A felhasználók száma körülbelül napi félmillió, ami nagyjából 0,8-1 GB adatmennyiségnek és hárommillió találatnak (hit) felel meg.
A merevlemezek RAID-be kötéséhez Mylex RAID vezérlőkártyákat használnak, továbbá minden gépen Debiant alkalmaznak.
Figyelemreméltó, hogy a régi rendszer bizonyos részei is Linuxokon futottak, így a Squid proxyk és az adatbázis-kezelők is, valamint az egyik webkiszolgáló is.

Pannonhalmi Főapátság

Sok különböző PC-t használnak a 486-ostól kezdve a Pentium III-ig, RedHat és Debian Linux alatt. A hálózat körülbelül száz gépből áll, ezeken Novell NetWare, Linux, DOS, Windows, Macintosh, valamint SCO Unix is megtalálható. A Főapátsági Könyvtár eddigi DOS-os adatbázisának átültetése folyik PostgreSQL alapokra, így az Interneten keresztül is kereshető lesz. A próbaváltozat elérhető a ➲ http://www.osb.hu/biblio címen.
A megoldandó feladatok: levelezőkiszolgáló (sendmail), webkiszolgáló (Apache), hírező-kiszolgáló (dnews), levelezési listák (mailman), fájlkiszolgáló (SaMBa), ftp-kiszolgáló (wu-ftpd), proxykiszolgáló (Squid), valamint PPP kialakítása. Közvetve vagy közvetlenül körülbelül négyszáz felhasználó alkalmaz Linuxot. A legkisebb – linuxozásra használt – gép 486 DX2 66 MHz-es processzorral, 16 MB RAM-mal van felszerelve, ezt X terminálként használják SCO Unix alatt futó adatbázis-kezelőhöz való kapcsolódásra.
A legnagyobb – Linux alatt futó – gép kétprocesszoros Pentium III-as, 256 MB RAM-mal és 50 GB háttértárral, programból megvalósított RAID-del. Ez a web-, FTP- és listakiszolgáló; továbbá fut rajta egy PostgreSQL adatbázis-kezelő is a vebkiszolgálóhoz.
Géphiba ritkán szokott előfordulni, leggyakrabban a merevlemezek „adják meg magukat"– ezért választották a RAID különböző formáit. Alkalmaznak gépi és programból megvalósított RAID-megoldásokat is. A biztonsági mentések szalagra történnek. A windowsos ügyfeleket megtámadó vírusok veszélyére is igyekeztek megoldást találni, ehhez a Sophos cég vírusölő programját szerezték be. Elég a SaMBa kiszolgálóra feltenni az új változatot, és az összes windowsos munkaállomás önműködően frissül. A linuxos kiszolgálón fut a Sophos intercheck démon, ez tartja a kapcsolatot az ügyfelekkel és vírus esetén jelez. Tény, hogy amióta ezt a vírusirtó megoldást megvásárolták és beüzemelték, azóta a vírusgondok az eddigiek töredékére zsugorodtak.
Néhányan irodai munkát is végeznek Linuxon, ehhez StarOffice-t, TEX-et és Lyxet használnak. A pannonhalmi hatosztályos gimnáziumban a diákok iskolai szakkörben ismerkedhetnek a Linuxszal és a Perl programozással.

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

Új termékek

Kapital a KDE-hez

A Kompany.com Kapital nevű terméke amerikai személyi pénzügyi csomag KDE-hez. Segítségével

nyomon követhetők a különböző banki műveletek, valamint beállítható „számlakövető" figyelmeztetés is a befizetések időzítéséhez. Szolgáltatásai közé tartozik még a csekk- és jelentésnyomtatás, keresés, illetve jelentéskészítés ábrákkal és grafikonokkal. Új számlaszámvarázslót kapott és import/export lehetőséget a Quickenhez.

Adatok: theKompany.com, Inc., PO Box 80265, Rancho Margarita, California 92688
telefon: 949-713-3276
e-mail: info@thekompany.com
➲ http://www.thekompany.com/

Smoothwall v. 0.9.8

A Smoothwall biztonsági program legújabb változata szabadon letölthető a GNU GPL szabályai szerint. A Smoothwall segítségével az otthoni felhasználók, a kisebb irodák és a vállalatok PC-i Weben keresztül kezelhetők, biztonságos internetes útválasztót építhetnek, amely védett internetkapcsolatot tesz lehetővé. A mostani változat újdonságai között szerepel az ISDN-eszközök önműködő megkeresésének és beállításának szolgáltatása, több internetes eszköz támogatása (így az útválasztó webkiszolgálásra is alkalmas), IPSEC VPN képesség, valamint az ADSL és a kábelmodemet használók támogatása. A program a ➲http://www.smoothwall.org/dyn/get/download/html/ címről tölthető le.

➲ Thames Court, Richfield Avenue, Reading, RG1 8EQ, United Kingdom
telefon: +44-0-118-956-6116
➲ http://www.smoothwall.org/

SuperScaler alkalmazáskiszolgálók

A nagysebességű alkalmazáskiszolgálók gyártója, az e-Appliance bemutatta új termékét, a SuperScaler alkalmazáskiszolgáló-családot. A SuperScaler 600 19 hüvelykes 1U szekrény, amelybe négy független, menetközben cserélhető kiszolgálót építettek be. Minden kiszolgálónak saját processzora, memóriája, lemezegysége, tápegysége és ethernetkapcsolata van. A SuperScaler 1000 két független kétprocesszoros rendszerből áll, amelyeket száloptikával kapcsoltak össze. Ezt a gépet nagy sebességet igénylő számítási feladatok – például médiafolyamok feldolgozásának – elvégzésére tervezték. Az SuperScaler 500 egy kétprocesszoros kiszolgálót tartalmaz, melyhez optikai kábel is csatlakoztatható. Ezt a gépet olyan alkalmazások futtatására élezték ki, amelyek nagy számítási és hálózati teljesítményt igényelnek.

Adatok: e-Appliance Corporation, 3052 Bunker Hill Lane, Suite 101, Santa Clara, California 95054
telefon: 408-980-1990
e-mail: operators@eappliancecorp.com
➲ http://www.eappliancecorp.com/

StoreSense a VA Linuxhoz

Az e-kereskedelem zászlóshajója, a Kurant Corporation cég StoreSense nevű terméke most már a VA Linux

KURANT™

1U és 2U kiszolgálóira is telepíthető. A StoreSense számos e-kereskedelmi eszközt és szolgáltatást foglal magába, ezek segítségével a vállalkozások internetes áruházat építhetnek, kezelhetnek és tarthatnak fenn. A StoreSense használatával a cégek testre szabott szolgáltatásokat nyújthatnak ügyfeleiknek az egyszerű katalógus-áruháztól kezdve a beszállítási lánc kezelésén át a vezeték nélküli vásárlás lehetőségéig. A StoreSense előfizetéses rendszerű és az alábbi felületeken fut: Linux, Windows NT, Windows 2000, Solaris és a Cobalt RaQ-ja.

Adatok: Kurant Corporation, 32 Cleveland Street, San Francisco, California 94103-4014
telefon: 916-984-5400
e-mail: info@kurant.com
➲ http://www.kurant.com/

Agenda VR3 PDA

Az Agenda Computing bemutatta az Agenda VR3 PDA-t a tavaszi Comdexen. A VR3 az Agenda Linux nevű beágyazott operációs rendszert futtatja, amelyet kézi és hordozható eszközökhöz fejlesztettek ki. A VR3 motorja a NEC VR4181 típusjelű, 66 MHz-en működő 32 bites processzor, emellé 8 MB memóriát és 16 MB flash-memóriát építettek be. A CIR áramkör és program szabványos, ez lehetővé teszi, hogy az eszközt háztartási berendezések távirányítására is használjuk. A VR3 ezenkívül vezeték nélküli infravörös átvitellel feljegyzéseket és üzeneteket tud küldeni a nyomtatónak vagy más eszköznek.

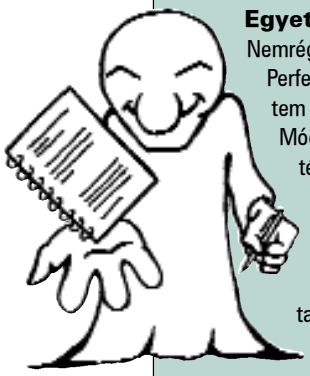
Adatok: Agenda Computing, #368, 4521 Campus Drive, Irvine, California 92612
telefon: 888-741-8181
e-mail: sales@agendacomputing.com
➲http://www.agendacomputing.com/

Game Channel Solution

A Not a Number cég, a Blender nevű háromdimenziós program készítője bejelentette Game Channel Solution (GCS) nevű elosztott, többszemélyes hálózati játékszolgáltatását. A GCS leendő felhasználói a játékosok társadalmának tagjai: a fejlesztők, az internetszolgáltatók, a viszonteladók és a fogyasztók egyaránt. A szolgáltatást mindenki az igényeinek megfelelő tulajdonságokkal választhatja, ugyanis a GCS előfizetési díj ellenében rendelkezésre bocsátja a számítógépet és a programot, a hálózati kapcsolatot és a sávszélességet. Naplózza a használatot, majd számláz, tartalomkezelést nyújt és vállalja a terjesztést, továbbá lehetőséget ad többszemélyes valós idejű játékokra. A GCS széles játékválasztékot kínál a hagyományos PC-fejlesztésektől kezdve az egyedi Blender termékekig, melyek számos támogatott felületen elérhetők.

Adatok: Not a Number, van Eeghenstraat 84, 1071 GK Amsterdam, The Netherlands
telefon: +31-(0)20-3058250
➲ http://www.blender.nl/

A hónap szakmai tanácsai



Egyetlen modul lefordítása?

Nemrég vettem egy USB lapolvasót (Epson Perfection 1640SU), és sikeresen beüzemeltem Linux alatt (2.2.16-22 rendszermag). Módosítanom kell viszont az egyik időértéket a rendszermag forrásában lévő scanner.h-ban. Miként fordíthatom újra a scanner.o modult anélkül, hogy az egész rendszermagot újra kellene fordítanom? Mentsem el a /lib/modules/2.2.16 tartalmát a fordítás előtt? Visszaállítható ez biztonságosan hiba esetén?
Jin, j.r.ong@ieee.org

A fájl módosítása után add ki a `make (cél)` parancsot a /usr/src/linux vagy hasonló nevű könyvtáradból: például `make bzImage`. Ez csak a szükséges fájlokat fordítja újra. A Linuxhoz tartozó Makefile-készlet úgy van beállítva, hogy bizonyos fájlokat mindenképpen lefordítson, ilyen például a main.c (ebbe a csoportba azonban csak néhány fájl tartozik). A módosítottak kivételével azonban átugorja az összes meghajtóprogramot.
Ha csak a .h fájlt módosítottad, a make program esetleg nem fordítja le a meghajtóprogramot. Ekkor használhatod a `touch` parancsot a scanner.c-re, hogy a rendszer úgy érzékelje, a scanner.c is megváltozott. Ezenkívül megkönnyítheted az életedet, ha csak a modult használod, ebben az esetben nem kell a rendszermagot újrafordítani, csak a `make modules` parancsot kell újra kiadni.
Chad Robinson, crobinson@rfgonline.com

Make zdisk SuSE-n Box zdead

A `make zdisk` segítségével 2.2.16-os rendszermagot fordítottam SuSE alatt. Amikor elindítom a rendszert, a betöltés előrehaladtát jelző pontsorozat kifutott a képernyő széléig, és megjelent az `out of memory` hibaüzenet.
Eskinder Mesfin, amesfin@uhc.com

A fordítást `make bzdisk` segítségével kell elvégezned, ez megoldja a gondodat. A `make bzdisk` másféleképpen rendezi a memóriát, így nagyobb rendszermagok működését is lehetővé teszi.
Marc Merlin, marc_bts@valinux.com

Sokféle segédanyag található a Linux Journal webhelyén. A Sunsite-tükrözés, a GYK-k és a HOGYAN-ok mind megtalálhatók a ➲ http://www.linuxjournal.com/ címen.

Nincs névkiszolgálóm, de használnom kell az nslookupot

Linuxot szeretnék telepíteni egy olyan gépre, amely elrejti a munkaállomásaimat, valamint kaputovábbítást biztosít a web- és levélkiszolgálóimhoz. Kell-e belső DNS kiszolgálót telepítenem a linuxos gépre, hogy a belső munkaállomásokról érkező DNS kérésekre – amelyek webböngészés közben keletkeznek – válaszoljon? Szívesebben használnám a hosts fájlt a név-IP-cím fordításra a belső hálózatomon.
Brandon Zumwalt, bzumwalt@seventhview.com

Nem kell napokat töltsened DNS kiszolgáló telepítésével, de a hosts fájl sem használható névfeloldásra. A hosts fájlt csak a linuxos gépen futó szolgáltatások használhatják. Az egyik lehetőség, hogy belső rendszeredet az internetszolgáltatód által nyújtott DNS kiszolgálók használatára állítod be. Ha az elrejtés helyesen van beállítva, munkaállomásaid el tudják érni és minden további nélkül fel tudják oldani a webcímeket. A másik lehetőség, hogy telepítesz egy DNS kiszolgálót a linuxos gépre, ez gyorstárazza a munkaállomások kéréseit. Ha egyszerűsíteni szeretnéd ezt a beállítást, használd a BIND „csak továbbítás" (forward-only) mintabeállítását. Ennek hatására a kiszolgáló csak gyorstáraz, nem épít fel saját helyi táblázatot.
Chad Robinson, crobinson@rfgonline.com

Nézz körül Linux-változatod honlapján, tartalmaz-e BIND biztonsági frissítést. Ugyanis a BIND régi változatai – például amit a változatod CD-je tartalmaz – gépesített támadások célpontjává válhatnak. Másik megoldásként futtathatod a pdnsd-t.
Don Marti, dmarti@linuxjournal.com

fsck újraindítás nélkül

Kiszolgálókat építek, amelyeknek folyamatosan kell üzemelniük egy évben 52 héten keresztül (természetesen, amíg szét nem égnek).
Lehetséges-e önműködően időről-időre ellenőrizni a felcsatolt fájlrendszereket?
Franco Favento dei Favento da Trieste, f.favento@ieee.org

Térj át a ReiserFsre (ez jegyzőkönyvező fájlrendszer Linuxra)! Enélkül is voltak ugyan olyan linuxos kiszolgálóim, amelyek éveket át futottak fájlrendszerhiba nélkül, de a ReiserFs a legmegbízhatóbb. Nagyon kisméretű főfájlrendszert indíts (vagy még jobb, ha memórialemezt használsz), és csak azokat a végrehajtható fájlokat tedd rá, amelyekre feltétlenül szükséged van az alaprendszer futtatásához. Így lecsatolhatod azokat a fájlrendszereket, amelyeket ténylegesen ellenőrizned kell.
Chad Robinson, crobinson@rfgonline.com

Egy felcsatolt fájlrendszer kizárólag csak olvasható módban ellenőrizhető. Nem lehet a hibákat javítani.
`fsck.ext2 -fn` *`eszköz.neve`*
Ez felfedezi a hibákat, és ezután megtervezheted a leállást a javításhoz.
Keith Trollope, keith@wishing-well.demon.co.uk

X és hang hálózaton keresztül

Megjelenhet-e egy X ablak egynél több X megjelenítőn? Megvalósítható-e, hogy DVD-kimenetet jelenítsek meg lakásszerte (csak olvasható módban)?
Amikor távoli gépen X-alkalmazást futtatok, hogyan varázsolhatom rá a hangot a helyi gép hangszórójára is? Eddig csak azt sikerült elérnem, hogy az alkalmazást futtató gépen szólalt meg a hang. A fentihez hasonlóan átirányítható-e a hang egynél több címre? (Még mindig a „DVD-lejátszás lakásszerte" az álmom.)
Matthew Holmy, mholmy@yahoo.com

Az xmx lehetővé teszi a szokásos X-alkalmazások megjelenítését több X képernyőn ➲ http://www.cs.brown.edu/software/xmx/, viszont a megjelenítési sebességre érzékeny alkalmazások (például a DVD-lejátszók) megkerülik az X-kiszolgálót, amikor a videokártyára írnak. Hacsak nincs egy lassú, hagyományos megjelenítési módjuk, ezeket nem lehet távoli képernyőre átvinni. Nem tudsz teljes képsebességű tömörítetlen videoanyagot ethernethálózaton átküldeni, mert ehhez túl sok az adat. Azt viszont megteheted, hogy a DVD-t az egyik gépen beolvasod, átküldöd a hálózaton, és a másik géppel lejátszod.
Hálózati hangrendszer segítségével megoldható a hanglejátszás a hálózaton keresztül is:
➲ http://radscan.com/nas.html.
Egy másik kipróbálásra érdemes lehetőség a ➲ http://rplay.doit.org/ címen található program.
Marc Merlin, marc_bts@valinux.com

Memóriazabálók felkutatása

Nemrég telepítettem Linuxot 256 MB memóriával rendelkező számítógépre, melyet kísérleti kiszolgálóként szerettem volna alkalmazni Oracle kiszolgálóval, továbbá Java-fejlesztésre is használnám.
Meglepetésemre a rendszerben memóriahiány lépett fel. Újraindítottam X nélkül, és megvizsgáltam a memóriahasználatot a `free` segítségével. Kiderült, hogy körülbelül 110 MB foglalt. Hogyan állapíthatom meg, hogy melyik folyamat mennyi memóriát használ?
Dhimant Patel, a24z57k@runbot.com

Ne feledkezz meg a `-/+ buffers` sorról, amikor a `free` parancsot futtatod. A Linux önműködően felhasználja a szabad memóriát az I/O kérések időszakos tárolására, és ezt a memóriát szükség szerint felszabadítja a programok számára. Leginkább a cseretár `used` állapotjelzője miatt kell aggódnod. Ennek kicsinek kell lennie – néhány megabájtnál kevesebbnek.
Használhatod a `ps aux | less` parancsot, ez megadja minden futó folyamat memóriafelhasználását. Csak a tárban maradó méret (RSS) értéke fontos, de ne feledd, hogy a memóriafelhasználás jelzett mennyisége nem feltétlen tükrözi a valóságot. A különbség abból adódik, hogy a ps az összes memóriát mutatja, a megosztott könyvtárak által elfoglaltat is, azok viszont csak egyszer töltődnek be, és minden folyamat használhatja őket, amelyiknek szüksége van rá.
Chad Robinson, crobinson@rfgonline.com

Két gigabájtnál nagyobb fájlok

Örömmel jelentem, a munkaadóm tett egy nagy lépést a Linux használata felé, és vett nekem egy többprocesszoros „pingvint” földrengésadatok feldolgozásához. Viszont én gyakran dolgozom 2 GB-nál nagyobb fájlokkal. RedHat 7.0-t használok 2.4.2 rendszermaggal és ReiserFssel, de továbbra sem tudok 2^{32} bájtnál nagyobb fájlokat kezelni.
Adam Cherrett, adam.cherrett@elfgrc.co.uk

El kellene olvasnod a ➲ http://www.suse.de/~aj/linux_lfs.html weboldalt. Nem elég, ha a rendszermag és a glibc támogatja a nagy fájlokat, neked is meg kell tenned a következőket a programjaidban:
1. Fordítsd a programjaidat a gcc `-D_FILE_OFFSET_BITS=64 -D_LARGEFILE_SOURCE` paranccsal. Ezzel minden fájlelérési hívást arra kényszerítesz, hogy a 64 bites változatot használják. Megváltozik számos típus is, például az off_t off64_t lesz. Ezért nagyon fontos, hogy mindig a megfelelő típust használd, és ne írj például `int`-et az off_t helyett.
2. Add meg a
`_LARGEFILE_SOURCE`
`_LARGEFILE64_SOURCE`
beállításokat. Ezekkel közvetlenül használhatod az LFS függvényeket, például az open64-et.
3. Használd az O_LARGEFILE jelzőt, amikor nagy fájlokat nyitsz meg.
Marc Merlin, marc_bts@valinux.com

PPP PCMCIA modemen

Dell gépemet gyárilag telepített RedHat 6.2-vel, 2.2.16-6.1.1 változatszámú rendszermaggal és gyárilag telepített PCMCIA modemmel vásároltam. Egy napon, a számítógép bekapcsolása után a rendszerbetöltési folyamat során az alábbiakat írta ki:

```
Bringing up interface ppp0 FAILED
```

A rendszerindítás későbbi szakaszában ezt az üzenetet kaptam:

```
Couldn´t configure serial #1 (port=760,
irq=3):
  device already open
serial_cs: register_serial() at 0x2f8
IRQ 3 failed
```

A modemkártyát kipróbáltam Windowst futtató gépemben és ott működik.
Steve Lohr, steve@azstarnet.com

Lehetséges, hogy egy olyan program fut, amely leköti a soros kapudat. A két leggyakoribb „tettes”: a GPM (szöveges módú egérvezérlő démon) és a sokfajta UPS-felügyelő démon egyike. A tettesek felkutatásához add ki a `ps ax | less` parancsot.
Ha még ezután is gondjaid vannak az azonosítással, indítsd el a rendszert egyfelhasználós üzemmódban, és egyenként zárd be a felesleges programokat. Ezután használd a minicom nevű programot, amellyel közvetlenül elérheted a soros kaput, és ellenőrizd, hogy látod-e a modemet.
Chad Robinson, crobinson@rfgonline.com

Lehet, hogy egy rendellenesen véget ért kapcsolat zárolófájlt hagyott a rendszerben. Keress régi zárolófájlokat a /var/lock könyvtárban! Ilyenek például a LCK..modem vagy LCK..ttyS1. Ha van ilyen, töröld le, és indítsd újra a gépet.
Keith Trollope, keith@wishing-well.demon.co.uk

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

Kinek a keze van a zsebünkben?

Néha a túl sok sem elég.
Dean Landsman

A .NET ámításként, a HailStorm ezzel szemben már csatába hívó szóként hatott a szakmai közvélemény soraiban.

Abban az időben, amikor *Steve Jobs* még a NeXT-nél dolgozott, *Robert X. Cringely* beszélgetett vele egy PBS-különkiadásban, amit azután a *Számítógépzsenik győzelme* címmel adtak le. A műsor Cringely könyvének (*Hogyan szereznek milliókat a Szilikon-völgyben dolgozó srácok, és miért nem tudnak csajozni?*) televíziós változata lett. Az adás legjobb pillanata az volt, amikor Cringely a Microsoftról kérdezte Jobsot.

Jobs hátradőlt, arcára öltötte leggúnyosabb mosolyát, és azt mondta: „Nincs ízlésük”. Ennyi: ezzel a két szóval Jobs nemcsak a Microsoftról mondott lesújtó ítéletet, hanem némi öndicséretig is elmerészkedett. Valóban, a Microsoftnak nincsen ízlése, Jobsnak viszont csak az van.

Az ízléstelenség „bűne” ugyanúgy terheli a Microsoftot, mint ahogyan például a McDonald’sot – ez egyértelmű. Az már viszont kevésbé az, hogy milyen termékeket rejtenek el ízléstelen nevek és az „új lehetőségeket feltáró szolgáltatások” semmitmondó leírásai mögé. Vegyük például a .NET-et. Egy évvel ezelőtt jelentették be, és még ma sem tudja senki biztosan, hogy mit takar. Megjelenésekor *Joel Spolsky* és a hozzá hasonló nagyágyúk átverésnek minősítették: „Olvassuk csak el a szakmai leírásokat és máris láthatjuk, hogy az egész egy nagy humbug; a .NET valójában egy vékonyka ködréteg a FUD felszínén. Az égvilágon semmi újat nem hoz. Minél jobban próbálnánk kihámozni a leírásokból a lényeget, annál jobban kicsúszik a kezünkből az egész.”

A HailStormmal más volt a helyzet. A .NET ámításként, a HailStorm ezzel szemben már csatába hívó szóként hatott a szakmai közvélemény soraiban. Amitől igazán felbolydult a méhkas, az a Passport (útlevél) nevű, ártatlannak tűnő szolgáltatás volt. Hallgassuk ismét Joelt:

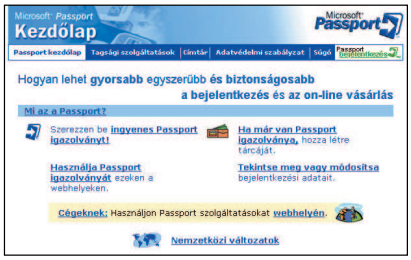
„Csak én lennék az egyetlen, akit elborzaszt a Microsoft Passport ötlete? Az egész úgy tűnik számomra, mintha a Microsoft a világ legnagyobb, legsokoldalúbb vásárló-adatbázisát akarná felépíteni, majd hatalmas bevételeket szerezni annak kibányászásával. Óriási veszélyt jelent a magánjellegű internethasználatra. A Passport fényében a mai sütik (cookies) az üzleti szellem kimondottan szelíd megnyilvánulásának tűnnek. A legrémisztőbb az, hogy a Microsoft az egészet úgy hirdeti, mintha ezzel a vásárlók járnának jól, és az emberek beveszik ezt!”

Álljunk most meg egy pillanatra és vizsgáljunk meg két jól álcázott tényt, melyek a Microsoftot segítették abban, hogy termékeiket a világ minden pontján megtalálhassuk. Először is nézzük meg, honnan indult az egész. A Microsoft a kezdetektől (1975-től) fogva a személyi számítógépek piacán terjeszkedik. A felhasználó az első, minden más csak ezután következhet. A nevükben lévő „Micro” szó sem véletlenül van ott. Másodsorban nagyon figyelnek arra, amit a felhasználók akarnak. Fogadni mernék, hogy amit egyetlen felhasználó javasol a terméktámogatáson ücsörgő ügyfélszolgálatosoknak, annak a Microsoft esetében jóval nagyobb esélye van a megvalósulásra, mint bármely más programfejlesztő cégnél. Egyszer a Microsoftnál dolgozó „fejesek” egyike elárulta nekem, hogy programjaik telis-tele vannak olyan szolgáltatásokkal, melyeket „pontosan egy, azaz egyetlen vásárló javasolt”. A legtöbb olyan céghez hasonlóan, amely milliók számára fejleszt programokat, a Microsoft is komoly bajban találhatja magát, ha vásárlóit egyszer csak nem ügyfélként kezdi kezeli, hanem valami másként. Nézzük csak meg a Passport címét ➲ http://www.passport.com/Consumer/default.asp. „Consumer”, azaz fogyasztó. Igen, fogyasztó vagy. Sőt, *Jerry Michalski* szavával élve, egy „garat”, melynek az a feladata, hogy a futószalag végén álljon, a Nagy Gyárból érkező termékeket gondolkodás nélkül benyelje és pénzt ürítsen. Szánalmas lények ezek a fogyasztók. Nézzük csak a HailStorm ismertetőjének „A felhasználó-központú élmény születése” című részét, amely arról a szörnyűségről szól, amit a Palm-OS-sel dolgozó szegény kicsi fogyasztóknak kell átélniük:

„Ha barátaink új telefonszámát szeretnénk bevinni otthoni számítógépünkbe, akkor csak a billentyűzetre és egy programra, például a Microsoft Outlookra van szükségünk, és néhány jól ismert, megszokott kézmozdulattal elvégezhetjük a feladatot. De ha ugyanezen adatokat egy Palm Pilotra szeretnénk bevinni, akkor teljesen új kezelőfelületet kell megismernünk – olyan ez, mintha újra kellene tanulnunk a kézírást! Egy ilyen környezet, melyben a felhasználóknak kell megszokniuk a megoldásokat, és nem a megoldás alkalmazkodik hozzájuk, jelentős mértékben rontja az alkalmazások, honlapok hatékonyságát…”

Na, vajon mi ez a szörnyűséges környezet? Úgy hívják: piac. Aggódó, szerető édes-

anyánk, a Microsoft viszont megkérdi tőlünk: de hát ki akar valódi piaci körülmények között élni? Hiszen az olyan zavaros. Olyan zajos. Túl sok benne a kereskedő, túl sok benne az okoskodó vásárló. Túlságosan valóságszagú. A fogyasztóknak rovarbábhoz hasonló körülményekre van szükségük, olyanokra, mint amilyeneket például a tévé is biztosít a számukra: egyirányú adatközlésre szolgáló csatornát, mint amilyenek a Mátrix hatalmas elemfeltöltő hálózatai is.
A Mátrix oldaláról nézve a HailStorm jelenti a „szolgáltatásokat”, az elem oldaláról pedig a Passport:
„A HailStorm használatára képes eszközök és alkalmazások önműködően a megfelelő HailStorm-szolgáltatásokhoz csatlakoznak. Mivel az életünk részét képező rengeteg alkalmazás és eszköz egy általunk irányított, közös adattárral kapcsolódik majd össze, ezért a különböző szakmai megoldások, emberek, szolgáltatások közötti adatmegosztás biztonságossá és egyszerűvé válik (…) A HailStorm úgy épül fel, hogy a szolgáltatások között következetes átjárhatóságot és bővíthetőséget alakítson ki. Lehetővé teszi az általános azonosítást, üzenetküldést, elnevezéseket, közlekedést, biztonságot, szereptérképezést, adatmodellezést, méréseket és a hibakezelést valamennyi HailStorm-szolgáltatás között.
A HailStorm dinamikus, megosztott, egyszerűsített XML tárolóhoz hasonlít. XML üzenőfelületeken (XMI-ken) keresztül érhetjük el, ahol a szolgáltatásfelületek hagyományos SOAP üzenetekként valósulnak meg, melyek bemenő és visszatérési értékei XML formátumúak, és minden szolgáltatás a HTTP Postot támogatja üzenetátviteli protokollként.”
Természetesen ennek ára is van:
„A Microsoft a HailStorm-szolgáltatásokat üzleti alapon fogja működtetni. A HailStorm-szolgáltatásoknál valós működési költségek jelentkeznek majd. A felhasználóközpontú modellt veszélyeztetné, ha e kiadásokat például hirdetők bevonásával fedeznénk, így a szolgáltatásokat igénybe vevők – a végfelhasználók – válnak a Microsoft első számú bevételi forrásává. A HailStorm segít majd abban, hogy az Internet végfelhasználói előfizetéses rendszerré alakuljon át, ahol a felhasználók fizetnek a megkapott tartalomért.”
Más szavakkal: a Microsoft az Internet üzleti oldalát egyetlen nagy online-szolgáltatássá szándékozik egybeolvasztani.
De nem a kábeltelevíziós rendszer mintájára, ahol a cső végén található dugóért fizetünk. Ó nem, hiszen te *a la carte* modellt szeretnél, ahol Te, az Elem fizetsz mindenért. Ez ugyebár jóval becsületesebb és hatékonyabb, mint a jelenlegi hitelkártyás rendszer, amely a végső tranzakció eladási oldaláról szép nagy szeletet vág le.
Emlékszünk még arra, amikor a Microsoft fel akarta vásárolni az Intuitot? A banki és hitelkártyás ügyletekkel foglalkozó vállalatok azonnal fellázadtak és eredményesen lobbiztak a szerződés megkötése ellen. A HailStorm éppen őket iktatja ki a játékból, szinte mind egy szálig, de gyanítom, hogy elképzelésük sincsen arról, mi történik a hátuk mögött.
Az átverés működőképesnek látszik.
A fejlesztőket azonban nem lehet átverni.
A Microsoft erre nem képes egyedül. Nézzük meg, hogy milyen csalit vetnek oda a kis halaknak:



➲ www.passport.com/Consumer/default.asp

„A Microsoft a fejlesztőktől is beszed némi pénzt, amely az általuk igényelt szolgáltatások és termékek költségeit fedezi. (...)
A szolgáltatások működtetői szerződéses viszonyban fognak állni a Microsofttal (...). Ez a szerződés lehetővé teszi, hogy a szolgáltatást ártalmas célokra használókat kiiktassuk a rendszerből. A szerződés megkötéséhez és ezzel egyidejűleg a HailStorm szolgáltatások használati jogához szintén költségek kapcsolódnak.”
Na, szépen vagyunk: körvonalazódni kezd a Microsoftnak behódoló programipar által felépített internetalapú kereskedelem, amely a Microsoft és a vele szövetséges fejlesztők által épített háttérrendszert fogja használni.
Álljunk meg egy percre! Szeretném megjegyezni, hogy én itt és most nem egyszerűen a Microsoft ellen buzdítok. A mindenki által „fogyasztói szemlélet” névvel illetett elvet támadom, bár ha ezt pontosan akarnánk megfogalmazni, akkor inkább „termelői szemléletnek” kellene neveznünk. Ez nem más, mint az a hit, hogy a Termelő megmondhatja a Fogyasztónak, hogy az mit akar.
Néhány hónappal korábban ezeken az oldalakon a Dellt és a Gatewayt szapultam azért, mert honlapjaik látogatóira a választás lehetősége nélkül erőltetnek rá bizonyos termékeket. A HailStorm segítségével a Microsoft ezt a rendszert olyanná bővíti ki, ahol minden „garat” odaállhat bármelyik futószalag végére. A Windows rendkívül széles körű elterjedtségét figyelembe véve elmondhatjuk, hogy valószínűleg ez fog történni. Vagy legalábbis ezt szeretnék.
Nem azzal van a gond, hogy a HailStorm elvágja a piactól a versenytársakat, a közvetítő feleket vagy bárki mást. Magát a piacot vágja el a működését biztosító forrásoktól. Megkerüli a bazárt. Azt dicsőíti és viszi tovább, amit *John Wanamaker* által az 1800-as évek végén feltalált árcédula óta a termelői szemlélet elért.
Nekünk, a többieknek egyetlen nagy kihívásunk maradt: ugyanazt kell tennünk a piaccal, mint amit a Nettel tettünk. Olyan, mindenki számára elérhető feltételeket kell teremtenünk, melyek a Mátrixok építgetését lehetetlenné teszik. Ehhez természetesen szükségessé válik egy dolog, amihez eddig nem mindenkinek volt alkalma hozzászokni: az, hogy a kereskedelemről gondolkodjunk. Pontosabban: a piacokra találkozóhelyekként kell tekintenünk, nem pedig egyirányú, a termékek „benyeletésére” szolgáló terjesztőhálózatként.
Ez már csak a kulturális okok miatt sem lesz könnyű. *Eric S. Raymond* Homesteading the Noosphere című könyvében írja: „a betyáreszmények különböző változatainak” egyik fontos közös vonása „a kereskedelmi programokkal és az ezeken nyerészkedő vállalatokkal szembeni ellenállás”. De a betyárok a Net építésében is tevékenyen részt vettek és vesznek, és az Internet olyan ígéretes terepet biztosított a vállalkozások számára, hogy milliószor millió dollárokat fektettek bele, és ezzel az üzletágat olyan „kokainkúrának” tették ki, amelyből az csak hosszú évek alatt lesz képes felépülni.
Nyilvánvaló, hogy a legtöbb befektetőnek elképzelése sem volt arról, hogy a betyárok valójában mire készültek. Eric S. Raymond 1999 elején, jóval a nagy összeomlás előtt a következőket mondta: „Mi, betyárok azt a célt tűztük ki magunk elé, hogy a hagyományos kereteken kívül működő új csatornákat hozzunk létre. Ezt nem a számítógépguruk tevékenységének melléktermékeként kell elképzelni, ez az 1970-es évek óta célunk volt. Ez tudatos forradalom.”
A Linuxszal a fejlesztők számára építettünk bazárt. Most itt az idő, hogy mindenki számára építsünk egyet. Hozzunk létre a kereskedelem számára olyan rendszert, amelyet, akárcsak az Internetet, senki sem birtokolhat, viszont mindenki használhat és bárki fejlesztheti. Az igazi piacok nyilvános helyek. Nem kaparinthatjuk meg azt, ami működik, csak azért, mert mindenki számára elérhető. De ha nem lépünk színre egy olyan rendszerrel, amely mindenkinél működik, akkor valakik hamarosan építeni fognak maguknak egyet. És abban nem lesz köszönet.

Doc Searls
(doc@ssc.com)
a Linux Journal szerkesztője és a Cluetrain Manifesto társszerzője.

Bukás lett a Linux?

Félre a szalagcímmel: természetesen nem.
Miután ezt tisztáztuk, lássuk miből élünk. A közelmúltban láttuk, ahogyan a linuxos cégek részvényei soha nem látott mélységekbe zuhantak. Linuxos újságok szűntek meg. Linuxos cégek húzták le a rolót vagy kényszerültek megválni alkalmazottjaik nagy részétől. A több lábon álló cégek kivonultak a linuxos piacról.
Mit jelent mindez?
A kristálygömböm helyett a nyolc évre visszatekintő linuxos tapasztalataimhoz folyamodom, és ezt egészítem ki csöppnyi logikával. Remélem, az eredmények nem maradnak el a régi, jól bevált kristálygömbhöz képest.
Amikor *Linus Torvalds* a Linux mellett döntött, elhatározása azon alapult, hogy hite szerint valami olyat képes alkotni, amelyre a világnak igazán szüksége van, ami valami újat nyújt az emberiségnek.
Az idő bebizonyította, hogy igaza volt. 1993-ban, amikor úgy döntöttem, hogy a világnak szüksége van a Linux Journalra, azért gondoltam így, mert hittem abban, amit Linus elkezdett, és úgy éreztem, igény van egy magas színvonalú szaklapra, amely segíti megismertetni ezt a rendszert a világgal. Az idő és az olvasók visszajelzései alapján elmondhatjuk, hogy igazam lett.
Most mi is a helyzet? Vonatkoztassunk el egy kicsit a linuxos közösségtől. A Sun Microsystems részvényeivel tavaly 120 dolláros áron kereskedtek, ma ezek a papírok 40 dollárt érnek (ami valójában húsz dollár alatti árat jelent, mert a papírokat időközben felezték). Az Apple részvényei egy nap alatt 50 dollárról 25-re estek vissza, és most, hat hónappal később is csupán húsz dollár körül mozognak. A Cisco rövidebb mint két hónapos mélyrepüléssel, több mint negyven dolláros vesztés után kevesebb mint húsznál tart. Folytathatnám…
Igaz, hogy az utóbbi időkben nem találkozhattunk a tőzsdén kiemelkedően nyereséges cégekkel, mégsem zuhant minden részvény több mint ötven százalékot. Mindössze annyi történt, hogy a befektetők megijedtek, kivették a pénzüket a spekulációs befektetésekből és biztonságosabbakat kerestek. Minden, aminek köze lehetett a dot-com cégekhez, bizalmatlanságot keltett, ezért ezek a cégek jártak a legrosszabbul.
Nem hiszel nekem? Vess egy pillantást az IBM-re. Amikor minden informatikával foglalkozó cég részvénye mélyrepülésbe kezdett, az IBM tartotta magát. Hogy miért? Mert a Nagy Kék biztos alapokon álló cég.
Nézzük most a másik végletet, a Linuxot, az új jövevényt. A Microsoft azt mondja, hogy a Linux rossz ómen. Azok a spekulánsok, akik nemrég még versengtek a Linuxért, most visszavonulót fújtak. Azok a cégek, amelyek a Linux irányába kezdtek terjeszkedni (legyen szó programfejlesztőkről vagy gépgyártókról) ráeszméltek, hogy ingoványos talajon járnak és visszaléptek. A Linux áldozatul esett a rövidtávon nagy haszonra számítóknak.

Ez sajnos azt jelenti, hogy néhány jó ember, aki hitt a Linuxban és spekulációba kezdett, rosszul járt. Ez azonban semmiképp sem csökkenti a Linux értékét.

Eszembe jut egy telefonbeszélgetés, amit az 1980-as évek közepén folytattam. Az SSC tartott néhány Unixszal kapcsolatos tanfolyamot: volt gyorstalpalónk vezetők számára, volt képzésünk nem műszaki beállítottságú emberek számára, természetesen programozókat is okítottunk, továbbá C-t is tanítottunk. A C-tanfolyamról volt szó. A beszélgetőpartnerem úgy gondolta, hogy sokat foglakozunk a unixos C-fordítóval, a unixos programkönyvtárakkal, a `make` paranccsal és sok más Unixra jellemző programmal. Azt állította, hogy tanfolyamunk sokkal több tanuló érdeklődését keltené fel, ha elhagynánk ezeket a részeket. Például számíthatnánk az MS-DOS-felhasználókra is.
Elmagyaráztam neki, hogy az SSC munkájának középpontjában a Unixszal kapcsolatos oktatás áll, és nem az általános számítógépes ismereteké. Azt gondolta, hogy megőrültem. Hát, lehet hogy őrült voltam. Lehet, hogy az voltam akkor is, amikor úgy döntöttem, hogy szükség van a Linux Journalra, de legalább jól érzem magam tébolyomban. Megéltük, hogy akik a haszon reményében szálltak be a Linuxba, most visszaléptek. Ez sajnos azt jelenti, hogy néhány jó ember, aki hitt a Linuxban és spekulációba kezdett, rosszul járt. Ez azonban semmiképp sem csökkenti a Linux értékét.
Ha már itt tartunk, beszéljünk a Linux igazi értékéről. Ha igazi gazdasági visszaesés tanúi vagyunk és nem csak a pénzek mozognak egyik területről a másikra, a Linux értéke növekszik. Például az Embedded Linux Journal testvérlapunk május-júniusi számában olvashattunk az amerikai postáról, ahol 7200 Linux-alapú számítógépet használnak – az irányítószámok beolvasására és vonalkód-nyomtatásra. Ez 7200 olyan számítógép, amelyen szabadon felhasználható, ingyenes program fut.
Ez csak egy példa a sok közül. Nézzük meg, hány webkiszolgáló fut a világon. Minden Linuxot futtató webkiszolgáló egy szerzői joggal levédett kereskedelmi programmal kevesebbnek enged teret. A Linux használatával felszabaduló pénzt át lehet áramoltatni más területekre. Lehet belőle további gépeket venni, ki lehet fizetni az alkalmazottaknak, vagy egyszerűen félre lehet tenni. Egyre több helyen használnak Linuxot.
Ha nem hiszed el, kezdd el számolni a webkiszolgálókat és a beágyazott rendszereket. Igaz, az asztali gépeken csak ritkán látni Linuxot, de a hatás ezen a területen érezhető a legkevésbé. A Linux köszöni, jól van. Tovább fog növekedni mind a felhasználói számát, mind az értékét tekintve. Nagyon sajnáljuk, hogy vannak olyan Linux-hívők, akik úgy gondolták, hogy a végcél a pénz, és nem a világuralom. Ez a kis megrázkódtatás remélhetőleg segít, hogy visszatérjünk a helyes útra.

Phil Hughes (phil@ssc.com) a Linux Journal kiadója, elkötelezett Linux-őrült. 1993 óta rengeteg embert nyert meg a Linux számára.

A hanyatlás napos oldala

A zuhanó alkatrészárak és a csökkenő kereslet előnyös a vásárlók számára.

A gazdaságban érzékelhető visszaesés megtépázta a számítógépek és más csúcseszközök iránti keresletet. Ez a megtört ütemű fejlődés súlyosan érintette a legtöbb félvezetőgyártó cég árbevételét és nyereségét. A cégek veszteségeiből azonban végül is a vásárlók húznak hasznot. A hanyatló alkatrészárak felszítják a keresletet a webpadok, a merevlemez-alapú képmagnók, a digitális filmfelvevők és az MP3-lejátszók iránt, amelyek közül számos Linux-rendszerrel működik.
A számítógépes lapkagyártás már számos fellendülési és hanyatlási időszakot megélt, és jelenlegi állapotát tekintve éppen leszálló ciklusban van. A baj az, hogy egy új lapkagyártó üzem vagy gyár építése egymilliárd dollárnál is több befektetést igényel, és a helykiválasztástól a tömegtermelés beindításáig mintegy két év telik el.
A kereslet fellendülésének szakaszaiban – ilyen volt a tavalyi év is –, a félvezetőgyártók képtelenek megfelelő ütemben növelni a termelést, ez viszont hiányt és magas alkatrészárakat okoz. Jelenleg azonban épp azt tapasztaljuk, hogy a növekvő kapacitás csappanó kereslettel találkozik.
A közgazdaságtani alapismeretek szerint az árak zuhanni fognak. Valóban: az elmúlt félév során a memóriamodulok ára több mint ötven százalékkal zuhant, akárcsak a lapos képernyők (LCD-kijelzők) árai.
A flashmemóriák és egyéb alkatrészek értéke szintén lefelé ereszkedik. Ezek az alacsony árak sok felfutó terméknek hajtanak hasznot. Például itt vannak a webpadok: ezeket a készülékeket még csak néhány cég árulja, általában ezer dollárnál drágábban. A magas árért a lapos képernyő ára a felelős, amely több száz dollárra rúgott.
A képernyőárak zuhanásával a kereskedők végre jóval ezer dollár alatt is összeállíthatnak webpadokat, sőt akár ötszáz dollár körülire is mérséklődhet az áruk. Nincs kétségem afelől, hogy a gyorsan alkalmazkodó piac értékelni fogja a kedvező árakat, és csökkenésével párhuzamosan folyamatosan bővülni fog. A mai napig bejelentett webpadok a Transmeta cég Crusoe lapkáját Linux-rendszerrel működtetik, ezzel teremtik meg a megfelelőséget a Világháló felé.
A TiVo cég csúcsképmagnója – amely a televíziós programokat videoszalag helyett merevlemezen tárolja – megkapta ugyan a megfelelő elismerést, mégis csupán körülbelül kétszázezret adtak el belőle. Az alacsony értékesítési arány részben a készülékek magas árával magyarázható: ötszáz dollárt kérnek érte. A TiVo tavalyi indulásakor a 30 GB-os merevlemez ára

A TiVo honlapja
➲http://www.tivo.com

háromszáz dollár körül mozgott. Mostanra ugyanez a merevlemez már száznál is kevesebbe kerül. Alacsonyabb árak mellett több ember fogja megvásárolni a TiVo rendszerét, hogy ezen keresztül élvezze kedvenc televíziós műsorát.
A TiVo-rendszerhez hasonlóan az MP3 – CD-lemeztár is olyan egyszerű számítógép, amelyet nagy tárhelyű merevlemez köré építettek. Ezek a lemeztárak közvetlenül kapcsolódnak a házi sztereorendszerekhez. Éppen mostanában kerülnek a boltokba, csakhogy az olyan árak, mint az AudioReQuestért kért 799 dollár, kizárják a nagy kereslet lehetőségét. Ezeket a rendszereket szintén a megszelídülő alkatrészárak tehetnék népszerűbbé. A SonicBlue – amely az MP3-as rendszerek Rio névvel fémjelzett termékét állítja elő –, a Kerbango és más gyártók egyöntetűen Linuxot használnak MP3-as lemeztáraik működtetésére.
Az MP3-lejátszók költségének legnagyobb részét a flashmemória teszi ki. A 250 dollár árú lejátszóban található 64 MB flashmemória tavalyi áron 120 dollárba kerül. Még ez a memóriaméret is csupán kb. néhány órányi zene tárolására képes. Alacsonyabb flashmemória-árakkal a szerényebb minőségű lejátszók ára 100 dollár körül fog mozogni, míg a 128 MB vagy nagyobb memóriával felszerelt készülékek ára 300 dollárnál kevesebb lesz.
A Napster átengedése ellenére e rendszerek népszerűsége tovább fog növekedni, mert lehetővé teszik, hogy a számítógép-használók zenei gyűjteményüket digitalizálják és mindenhová magukkal vigyék. A Linux kulcsszerepet játszhat a géphasználók MP3-rendszereinek hálózattá alakításában, így lehetőség nyílik a zene zavartalan hordozhatóságára.
Sajnálatos módon a lapkák mérséklődő árai nem mozdítják elő a számítógépek eladását. 399 dolláros árszintnél a gépek már olyan olcsók, amennyire csak lehetnek, ráadásul a kulcsfontosságú alkatrészek eltávolítása nélkül. Emiatt a számítógépgyártók a kisebb lapkaköltségeket használják fel arra, hogy gyorsabb processzort, nagyobb memóriaméretet és merevlemeztárhelyet nyújtsanak ugyanazért az árért. Néhányan talán elcsábíthatók azzal, hogy ugyanazért az összegért jobb gépet kapnak, ámde a nehezebb feladat mégis-csak a felhasználók többségének meggyőzése lesz, hogy márpedig nekik nagyobb teljesítményű számítógépre van szükségük. A mostani gépek többsége kitűnően megfelel levelezésre, szövegszerkesztésre vagy éppen a Világhálón való böngészésre. Mindemellett a digitális képrögzítés, fényképezés és zenefeldolgozás növekvő népszerűsége siettetni fogja a következő gépfejlesztési időszakot.
Az iparban tapasztalható borúlátás ellenére a módszer még mindig nem vált rossz ötletté. Az Internet továbbra is alapjaiban változtat meg mindent, amivel dolgozunk vagy amit előállítunk. Sőt, a digitális adathordozókhoz való viszony még a személyi szórakoztatást is forradalmasítja. Az 50–75 százalékkal mérsékelt lapkaárak következtében ezen irányzatok fejlődése folyamatosan gyorsulni fog a jövőben is.

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.

Nemzetköziség és a fejlődő piacok

Amikor visszatértem az idei hannoveri CeBIT kiállításról – ahol Jon „maddog” Hall (Linux International) kihasználta a kiváló alkalmat arra, hogy a németországi kormány számára a Linux által nyújtott előnyöket elemezze – azt hiszem, rájöttem, miért is lelkesedik ennyire Nyugat-Európa a Linuxért. A Linux Journal standjánál megálló lelkes érdeklődők között egyaránt találtam holland, angol, francia, spanyol, portugál és belga szakembereket. Ez természetesen várható is volt. Melyik Linux-rajongó ne ismerné a SuSE vagy a Mandrake nevet? A Linux – tulajdonképpen – az európai szellem szülötte. Az a hatás azonban meglepett, amelyet a Linux gyakorol a világra, s erre azóta kezdtem felfigyelni, amióta elkezdtem a Linux Journalnál dolgozni. Például az egyik legérdekesebb Linux-alapú termék, melyet a CeBIT-en láttam, nem egy európai, nem is egy amerikai cég terméke volt, hanem a Galleo izraeli vállalat Mobile Multimedia Communicator nevű készüléke. Nem lep meg, hogy a Linux nemzetközi hódító körútra indult, hiszen az Internet gyermeke nincs tekintettel a földrajzi határokra, és terjedése elé a magas költségek sem gördítenek akadályt – hacsak nem az internet-elérés létesítésével kapcsolatosak. Gondolom, olvasóink már észrevették, hogy a cikkek szerzői a világ különböző pontjain élnek, és különféle számítástechnikai kihívásokkal szembesülnek. Megoldásaik azonban mind a Linux által nyújtott méretezhetőséget és rugalmasságot igazolják.

John Biggs nemrég két évet töltött Lengyelországban; a 38. oldalon található a Lengyel Linux című cikkében arról számol be, hogy a lengyelek milyen területeken karolták fel a Linuxot, és milyen akadályok tornyosulnak még előttük.

Jorge Eduardo Nieto Lema emlékeztet bennünket arra, hogy a nyílt forrású programok alkalmazásának lehetséges előnyei a nemzetek számára immár a programok jogdíjának megfizetésénél sokkal fontosabb kérdések megoldására teremtenek lehetőséget. A 35. oldalon ismerteti velünk a terminálkiszolgáló-építés módszereit, melynek segítségével lemeznélküli munkaállomásokat telepíthetünk, amelyek egy hálózati linuxos kiszolgálóról indítják az operációs rendszert; így olyan környezetet alkotnak, mely főképp a szűkös költségvetésű cégek számára lehet előnyös. Jorge elmagyarázza a telepítés folyamatát, majd beszámol tapasztalatairól, melyeket az LTSP üzembe helyezésekor szerzett egy kolumbiai vállalatnál.

Richard Vernon

A nemzetközi helyzet nálunk is érezteti a hatását. De mint minden országban, hazánkban is egyedi kép alakult ki. *Szaló István* arra vállalkozott, hogy egy elmélkedő cikksorozat keretében áttekintse, milyen a Linux és a nyílt forrás helyzete ma Magyarországon a legfontosabb területeken: az oktatás berkein belül.

A szerző tanárként saját bőrén érzi a változásokat, és mint minden jó pedagógus, hosszú távra tervez. Milyen oktatási környezetben nőnek fel gyermekeink? Melyek a választási lehetőségek és vajon a legjobb lehetőség mellett döntünk-e?

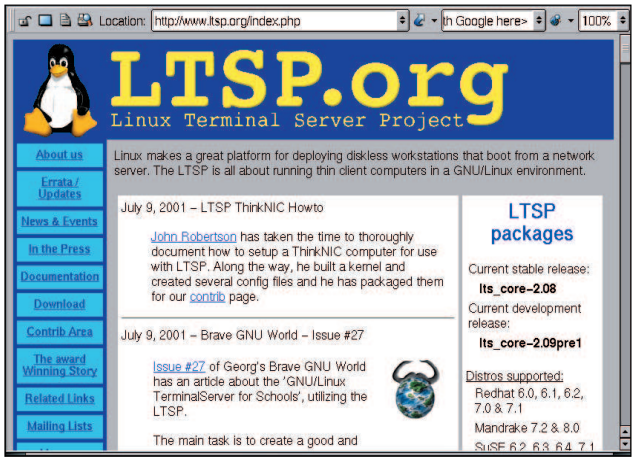
Nagyon fontosnak tartom, hogy ezen a területen is értsék és megértsék a Linux, a nyílt forrás és a szabad programok előnyeit, erősségeit. Világszerte villámgyorsan befogadták ezeket az újdonságokat, nemigen akad már olyan fejlett ország, ahol a főiskolákon, egyetemeken ne oktatnának Linuxot. Vajon lesz-e kis hazánkban oktató, tanár, aki hajlandó az új rendszerrel foglalkozni? Vajon lesz-e vállalatvezető, beszerző, ügyintéző, aki átlát az erőszakos reklámhadjáratot folytató nyereségközpontú cégek plakátjain, saját jelenén és nem fél az újba vágni? Lesz? Szerencsére van! Még messze nem elegendő számú ember ismeri ezt az új világot, de már most többen vagyunk, mint azt az ellentábor kívánná.

Szy György

Építsünk linuxos terminálkiszolgálót

Csökkentsük a költségeket és növeljük a termelékenységet könnyen telepíthető lemeznélküli munkaállomásokkal.

A nyílt forráskód elterjedésével sok latin-amerikai ember és intézmény juthat olyan módszerek és tudás birtokába, amely egyébként csak a fejlettebb országokra jellemző. A megfelelő háttérrendszer, a gazdasági források hiányosságai és a lakosság túlnyomó többségének hiányos műszaki képzettsége az oka annak, hogy Latin-Amerika a számítástechnika területén messze a világelsők mögött kullog, és az Internet által létrehozott információs forradalomból nem képes hasznot húzni. Például Kolumbia népességének (megközelítőleg 40 millió fő) kevesebb mint tíz százaléka jelentkezett be valaha is az Internetre. Ez azonban nem tekinthető kivételnek, hiszen az egész térségben hasonló gondokkal küszködnek. A költségtakarékos, hatékony és könnyen kivitelezhető nyílt forráskód szabad felhasználhatósága és ingyenessége azonban segíthet abban, hogy ezek az országok beérjék az iparosodott világot.



➲ http://www.ltsp.org/index.php

Létezik egy nyílt forráskódú fejlesztés, amely nagy változásokat idézhet elő Latin-Amerikában: a Linux Terminal Server Project vagy LTSP ➲ http://www.ltsp.org/. Ennek a fejlesztésnek az igazi vonzereje az, hogy egyszerű módszert kínál igen kedvező árú, lemeznélküli munkaállomások felállítására, melyek egy Linuxot futtató hálózati kiszolgálóról indíthatók el. Jelenleg kevés latin-amerikai intézmény engedheti meg magának, hogy új számítógépeket vásároljon vagy drága szerzői jogdíjakat fizessen ki a programokért.
A Linux és a lemeznélküli munkaállomások segítségével viszont jókora összegeket takaríthatnak meg, valamint termelékenyebbé válhatnak. A módszer felhasználható arra is, hogy gyermekeket és munkásokat tanítsanak meg arra a műszaki tudásra, amelyre a jövőben mindenképpen szükségük lesz.
Ebben az írásban az LTSP-ről fogok beszámolni, azokat a tapasztalataimat összegzem, melyeket egy kis kolumbiai intézetben szereztem, és bemutatom, mire is volt szükségem a telepítéskor és a beállításkor. Ezek a sorok azonban pusztán figyelemfelkeltésül íródtak, nem egy LTSP HOGYAN-nak szántam őket. A pontos telepítési és beállítási utasítások azonban a projekt honlapján megtalálhatók:
➲ http://www.ltsp.org/documentation/.

Mi is az a Linux Terminal Server Project (LTSP)?

Az LSTP célja egy olyan egyszerű módszer kidolgozása, amellyel lemeznélküli munkaállomásokat lehet felállítani Linux alatt. A meghatározás szerint a lemeznélküli munkaállomás olyan számítógép, amely azután indul el, hogy a helyi hálózaton található kiszolgálóról letöltötte az operációs rendszert. Az LTSP-eszközök ezt a feladatot úgy oldják meg, hogy a kiszolgáló tartalmaz egy kis rendszermaglenyomatot, XFree86 kiszolgálót, és néhány más hálózati eszközt, amelyeket kérelem esetén a munkaállomásoknak továbbít. A lemeznélküli munkaállomásnak mindössze egy indító ROM-ra van szüksége a megfelelő hálózati kártyához, hogy a megfelelő programokat a kiszolgálóról letölthesse. Jó hasznát véve az Etherboot project nyílt forráskódú programjának ➲ http://etherboot.sourceforge.net/, az LTSP elkészítette és programjával feltöltötte saját indító-ROM-ját. Látván, hogy egy hálózati kártyához való indító-ROM készítése (égetése) elég bonyolult feladat, egy, a fejlesztéshez közel álló cég, a Diskless Workstations ➲ http://www.DisklessWorkstations.com/ honlapján alacsony vételárral kínálta saját, előre beállított változatát. Érdemes megemlíteni, hogy az LTSP indító-ROM-okat hajlékonylemezre is ki lehet írni, amelyről aztán a lemeznélküli munkaállomás elindulhat.

Tapasztalataink az LTSP-vel

Az elmúlt évben kezdtem LTSP-n alapuló lemeznélküli munkaállomásokkal foglalkozni, amikor a cukornád-feldolgozóipari munkások egyik kis szervezetének olcsó, könnyen használható számítógépes és adatkezelő rendszerre volt szüksége. A cég, a Productivos Ltda. csak igen szűkös anyagi keretekkel gazdálkodhatott, és munkásai sem voltak különösebben jártasak a számítástechnika és a programok területén. Mindössze egyetlen gépük volt, melyen Windows 95 és egy kalóz MS Office futott – ezzel kezelték a levelezést, a könyvelést és a bérlistákat. Az iroda egyéb teendőit továbbra is kézzel végezték. A szervezet új vezetője öt új gépet kívánt vásárolni, hogy emberei a munkával könnyebben megbirkózhassanak. Szeretett volna kialakítani egy helyi hálózatot is, hogy az alkalmazottak megoszthassák az állományokat, a nyomtatókat és elérhessék az Internetet.
Miközben alacsony költségű megoldás után kutattam a Hálózaton, beleütköztem az LTSP-be. Kéznél volt a megoldás. A Productivos Ltda.-nak nem kellett költséges, mindentudó számítógépeket vásárolnia és drága jogdíjakat fizetnie. Mindössze egy LTSP-hálózatot volt szükséges beszereznüük és termelőeszközül a StarOffice-t használniuk.
A tulajdonképpeni telepítést próba előzte meg, amelynek során egy öreg lemeznélküli Pentium-gépet indítottunk hajlékonylemezről. Néhány kisebb fennakadás után, amit az X alatti nemzetközi billentyűzetkiosztás helytelen beállítása okozott, úgy találtam, hogy ez a rendszer a legmegfelelőbb a Productivos Ltda. számára.
A szervezet vásárolt négy új AMD K6-osztályú lemeznélküli gépet 32 MB memóriával (ez volt az elérhető legkisebb kiépítése), hogy lemeznélküli munkaállomásokként működjenek, és egy Pentium II 350 MHz 128 MB memóriájú gépet kiszolgálónak, illetve munkaállomásnak. Bár ez a kialakítás a legtöbb kisméretű irodába megfelelő megoldásnak tűnhet, a Productivos költségvetése más körülmények között (pl.: windowos hálózatban) csak kevesebb gépet tett volna

➲ http://www.redhat.com

➲ http://www.icewm.org/index.php?page=shots

lehetővé. A Productivos Ltda. csak a programok szerzői jogdíján körülbelül 3000 dollárt takarított meg. A munkaállomásokba rendszerindító hálókártyákra volt szükség, hogy ne hajlékonylemezről kelljen indítani a rendszert. Négy Linksys 10/100 Mb típusú rendszerindító hálózati kártyát vásároltunk közvetlenül a DisklessWorkstations cégtől, egyenként 34 dollárért. A hálózathoz használt 8-kapus Linksys 10/100 Mb elosztót az Outpost.com-tól szereztük be ➲ http://outpost.com/. A szükséges UTP Level 5 kábelezés egy héten belül elkészült. Az elosztó és a teljes kábelezés ára megközelítőleg 350 dollárnyi összeget tett ki.

A kiszolgálóhoz RedHat 6.2 ➲ http://www.redhat.com/ terjesztést választottunk, ugyanis ez egy teljes körűen kipróbált, könnyen beállítható és karbantartható Linux-változat. Mivel az X és a StarOffice 5.2 ➲ http://www.sun.com/staroffice/ az összes lemeznélküli munkaállomáson távolról futott, úgy döntöttünk, nem használunk olyan memóriafogyasztó grafikus felületeket, mint a Gnome vagy a KDE, hiszen ez csökkentette volna a rendszer teljesítményét. Olyan ablakkezelőt kerestünk, amely csak az alapszolgáltatásokat tartalmazza (indítóasztal és menük), ezért választottuk a legegyszerűbb fellelhető kezelőt, az IceWM-et ➲http://sourceforge.net/projects/icewm/. Mindössze néhány bejegyzést kellett megváltoztatnunk az IceWM fő beállításfájljaiban, hogy azt összeállításunkhoz igazíthassuk. Tíz különféle felhasználóazonosítót készítettünk a kiszolgálón, mindegyiket saját StarOffice-munkaállomás telepítéssel. Nem sokat számított, hogy a StarOffice meglehetősen nagy memóriafogyasztó alkalmazás, hiszen ez volt a másik ok, ami miatt a Gnome vagy a KDE helyett egy kis ablakkezelőt választottunk. A nyomtató telepítése egyszerű volt, a RedHat nyomtatóeszköz kiváltására szánt LTSP-nyomtató része a projekt programjainak. Az új LTSP-hálózat, a teljes telepítés és beállítás egyetlen nap alatt elkészült.

Az éles kipróbálás ideje akkor jött el, amikor a felhasználók először jelentkeztek be a hálózatba. Teljesen lenyűgözte őket az a tény, hogy munkájukban immár számítógépek segítik őket. Természetesen némi előképzésre is szükség volt ahhoz, hogy valóban használni tudják a gépeket. Minden felhasználó kéthetes, házon belüli oktatáson vett részt, hogy megismerkedjen az új eszközökkel. A képzésben a számítástechnikai alapok és a StarOffice használatának elsajátítása szerepelt. A képzés végére mindenki gond nélkül használta a lemeznélküli munkaállomásokat a korábban kézzel végzett feladataihoz. A szervezet LTSP-hálózata immár több mint egy éve kitűnően működik, olyannyira, hogy újabb lemeznélküli munkaállomások beszerzését tervezik. A Productivos Ltda. ezáltal sokkal hatékonyabbá vált, és emellett megadta munkásainak a lehetőséget, hogy megismerhessék és megtanulhassák az új módszert. Mindez meglehetősen nehéz lett volna az LTSP és a nyílt forráskódú fejlesztések segítsége nélkül.

Mire van szükség az összeállításhoz?

A LTSP-hálózat rendszerkövetelményei attól függően változhatnak, hogy hány lemez nélküli munkaállomás éri el egyidejűleg a kiszolgálót, illetve milyen alkalmazások futnak majd rajtuk. A két legfontosabb dolog, amit figyelembe kell venni: a kiszolgáló memóriája és a hálózat sebessége. Bár egy 10 Mb-es hálózat is számos lemeznélküli munkaállomást képes ellátni konzolalapú alkalmazások futtatásakor, ha a munkaállomásokon távoli X-folyamatok és a StarOffice-hoz hasonló programok fognak futni, gyorsabb (100 Mb) hálózat szükséges. Az LTSP-hálózaton a legkisebb kiszolgáló kiépítése egy Pentium-osztályú gép legalább 64 MB memóriával és 2 GB merevlemezzel. Munkaállomásként pedig egy 486 vagy K5 processzorral felszerelt számítógép 16 MB memóriával és 1 megás videokártyával (az X-hez) már képes ellátni a feladatot.

A teljes LTSP-hálózat hatékony és gördülékeny működése a hálózati kártyák, elosztók és a vezetékek jó állapotától függ. Mindenképpen győződjünk meg erről a telepítés előtt.

A program

Az LTSP-eszközkészlet jelenleg a következő Linux-változatokon működik:

- RedHat 6.0, 6.1, 6.2 és 7.0,
- Mandrake 7.2,
- SuSE 6.2, 6.3, 6.4, 7.1 és 7.2-beta,
- Debian 2.0, 2.1 és 2.2,
- Caldera eDesktop 2.3, 2.4 és eServer 2.3,

Az LTSP-hálózat futtatásához szükséges összes program és leírás honlapjuk letölthető részéről ➲ http://www.ltsp.org/ beszerezhető. A szükséges előre lefordított csomagok RPM és TGZ formátumban is megtalálhatók. Ne feledjük elolvasni a leírást, mielőtt a program telepítéséhez vagy a parancsfájlok futtatásához kezdenénk!

Gyors telepítési útmutató (kivonat az LTSP telepítési útmutatójából)

A gyors telepítéshez a következő feltételeknek kell megvalósulniuk:

- RedHat 6.0, 6.1, 6.2 vagy 7.0-alapú rendszeren vagy Mandrake 7.2-alapú rendszeren készül.
- A kiszolgáló IP-címe: 192.168.0.254.
- A munkaállomások címtartománya 192.168.0.1-től 192.168.0.253-ig terjedhet.

➲ http://www.riverdale.k12.or.us.linux/k12ltsp.html

- Az alkalmazások a kiszolgálón fognak futni, a munkaállomásokon csak a megjelenítés történik.
- DHCP-t használunk.

Töltsük le az LTSP RPM csomagokat az LTSP letöltési oldaláról ➲ http://www.ltsp.org/.

- lts_core-2.xx-xx.i386.rpm-the alap LTSP-csomag, amely a főfájlrendszert, a beállítási eszközöket és a munkaállomásokhoz tartozó leírást tartalmazza.
- lts_kernel_xxxx-2.xx-xx.i386.rpm-precompiled rendszermagok a lemeznélküli rendszerindításhoz. A lemeznélküli munkaállomás hálózati kártyájához megfelelő rendszermagot kell választani.
- lts_xxxx-2.xx-xx.i386.rpm-precompiled X-kiszolgálók. A lemeznélküli munkaállomás hálózati videokártyájához megfelelő X-kiszolgálót kell választani.

Telepítsük az rpm -i lts_core-2.xx-xx.i386.rpm, rpm -i lts_kernel_xxxx-2.xx-xx.i386.rpm és az rpm -i lts_xxxx-2.xx-xx.i386.rpm csomagokat.
Ellenőrizzük, hogy lett-e dhcpd telepítve a kiszolgálóra! Futtassuk a következő parancsot:

```
rpm -qa | grep dhcp
```

Ennek ilyesféle sort kell visszaadnia:

```
dhcp-2.0-5
```

Ha nem így történik, le kell töltenünk a DHCP RPM-et a RedHat telepítőlemezről.
Ha a fenti csomagok telepítése elkészült, be kell lépnünk a /tftpboot/lts/templates könyvtárba. Néhány itt található fájl beállítja a rendszerfájlokat a kiszolgálónkon – minden fájl egy rendszerfájlért felelős. Mindenképpen nézzünk bele ezekbe a fájlokba és bizonyosodjunk meg arról, hogy mindennel egyetértünk-e, hiszen egy ilyen parancsfájl kiválóan alkalmas lehet arra, hogy rendszerünket külső behatolással szemben sebezhetővé tegye. Elképzelhető az is, hogy a rendszerfájlokon való változtatásokat kézzel szeretnénk elvégezni. Ha az önműködő megoldást választjuk, futtassuk le az `ltsp_initialize` parancsot:

```
$ cd /tftpboot/lts/templates
$ ./ltsp_initialize
```

Másoljuk a /etc/dhcpd.conf.example állományt a /etc/dhcpd.conf-ba. Majd módosítsuk a dhcpd.conf fájlt úgy, hogy a munkaállomásban található hálózati kártya MAC-címét is tartalmazza. Végül adjuk a következő sort az /etc/hosts fájlhoz:

```
192.168.0.1     ws001
```

Következő lépésként átszerkeszthetjük a /tftpboot/lts/ltsroot/etc/lts.conf fájlt, hogy az ott található beállítások megfeleljenek a munkaállomásokhoz. Ezután indítsuk újra a kiszolgálót és kapcsoljuk be a munkaállomásokat. A munkaállomásokon most egy grafikus bejelentkező felületet (login prompt) kell látnunk. A kiszolgáló bármely felhasználó-azonosítójával bejelentkezhetünk. Látható, hogy az LTSP-hálózat telepítése elég egyszerű, akár öt perc alatt is futásképes állapotba hozható. Az utasításokat mindig pontosan kövessük, és ha valamilyen gondba ütközünk, nézzük át a leírás hibamegoldó (troubleshooting) fejezetét. A fejlesztés levelezőlistája szintén jó lehetőséget kínál, ha valamilyen kérdésre keresünk megoldást, netán új fejlesztéseket akarunk kigondolni vagy egyszerűen mások tapasztalataiból szeretnénk okulni.

Összefoglalás

Érdemes ennek a nyílt forráskódú fejlesztésnek néhány főbb ismérvét kiemelni: a leírás mindig naprakész és a telepítés összes lépésén végigvezeti a felhasználót; kitűnő beállítás- és hibakereső rendszere van; a fejlesztés hozzájárulásgyűjtő oldalakkal (contributions area) rendelkezik, ahol az emberek saját fejlesztéseiket tehetik közzé, ideértve az olyan rendszerek támogatását is, mint az LDAP és a dinamikus DNS. Végül, de nem utolsó sorban igen aktív levelezőlistája van, ahol a legtöbb kérdésre maguk a fejlesztők válaszolnak – igen gyorsan.
Az LTSP immár a SourceForge része, ez jó hír a felhasználók és a fejlesztők számára egyaránt. Számos új fejlesztés várható a közeljövőben, így az LTSP lemeznélküli munkaállomáshoz az egyik legjobb választás lehet mind Latin-Amerikában, mind a világ többi részén. A nemrégiben megjelent RedHat-alapú K12 terjesztés ➲ http://www.riverdale.k12.or.us/linux/k12ltsp.html már tartalmaz programokat az LTSP-től. Az, hogy a K12 fejlesztést főként iskolákba és gyerekeknek szánták, talán a legjobb bizonyíték az LTSP jelentőségére, és arra, hogy jelentős fejlődés előtt áll.
A projekt leírását spanyol nyelven kívül más nyelvre még nem fordították le. Reméljük, hogy az LTSP-nek a SourceForge-ba történő felvétele rengeteg tettre kész fordítót vonz majd a világ minden országából, hogy hamarosan elkezdjék munkájukat.
Az LTSP-vel való első találkozásom óta több lemeznélküli munkaállomáson alapuló rendszer kialakításában is volt szerencsém részt venni általános iskolákban, Internet-kávézókban és kisvállalatoknál. Legnagyobb megelégedésünkre mindezen hálózatok mind a mai napig kitűnően működnek és a legtöbb felhasználónak javára vált az új módszer. Talán az egész Nyílt Forráskód-elképzelés is pontosan erről szól: segítsünk egymáson.
Szeretném megköszönni *Jim McQuillan*-nek (jam@McQuil.com) és barátainak a Linux Terminal Server Project fejlesztésével és karbantartásával kapcsolatos kiemelkedő munkájukat, amivel hozzájárultak a Nyílt Forráskód fejlődéséhez.

Jorge Eduardo Nieto Lema (jnieto@yupimail.com) Kolumbiában él és független Linux-szaktanácsadóként dolgozik. Jelenleg azon fáradozik, hogy az LTSP-programot kedvenc Linux-terjesztése alá, a Slackware-re vigye át.

Lengyel Linux

A Linux kedvező fogadtatásra és értelmes fejlesztői közönségre talált Lengyelországban.

Varsó központjától öt percre már előtűnik a város. A szovjet időkben épült Kultúra Palotájában a programok csaknem ingyenesek. Akárcsak a többi közép- és kelet-európai országban, itt is féktelen programkalózkodás zajlik: a Stadionnál mindössze körülbelül öt dollárba kerül a Microsoft Windows Millennium Edition vagy egy *Ricky Martin* CD.

A Stadion mostanában már omladozik és azoknak a hetivásároknak ad otthont, melyeken jellemzően kalóz popzenét és játékokat, olcsó ruhákat, orosz emléktárgyakat, rozsdás bajonettet, repülős szemüveget lehet vásárolni. A lengyel országgyűlés által elfogadott törvények az elmúlt három évben csökkentették ugyan a kalózkodás mértékét, az International Intellectual Property Alliance 2001-es becslése szerint a szóra-koztató programok 85 százaléka, valamint a Windows-termékek 55 százaléka mégis kalózmásolat Lengyelországban.

Az egyetlen piaci terület, amivel a kalózok nem képesek megbirkózni: a Linux. Varsó jelentős szakmai folyóirata, a Linux Plus teljes RedHat-változatokat ad ki körülbelül négy dollárért, és a Lengyel Linux Terjesztés (Polish Linux Distribution, azaz PLD ➲ http://www.pld.org.pl/) gőzerővel halad, mivel diákok és szakemberek sokasága választja a szabad rendszert. Bár a Stadionnál időről időre feltűnik a RedHat is, rövid időn belül eltűnik, ugyanis egyszerűen nem kel el; így aztán a kalózok CD-iket inkább a drágább programok másolására használják fel.

„Sok fiatal ébred rá arra, hogy nem csak Windows létezik a világon” – véli *Tomasz Kloczko*, a PLD szóvivője és a Gdanski Műszaki Egyetem programozója. „Újabb és újabb felhasználók lépnek ki az egyetemek és a középiskolák kapuin, és az utóbbi pár évben jól látható folyamat, hogy a Linuxot már az üzleti életben is elfogadják.”

Varsó városközpontjában, a III. Jan Sobieski Hotelben egy másik programforradalom zajlik. A szálloda teljes gépparkja, valamint a fizetős televíziórendszer – néhány jól ismert grafikus program kivételével – SuSE és Mandrake Linux-változatok alatt működik. A szobazáraktól kezdve egészen az egészségügyi rendszerig minden nyílt forráskódú programon fut. A Pro-Test cég, amely a programot telepítette, a varsói székhelyű Softomat cég tanácsadóinak segítségével

➲ www.linuxclub.pl

a Sun Microsystems StarOffice programját lefordította lengyelre. A fent említett projekt megvalósításával nem csak dollárezreket takarítottak meg a szálloda üzemeltetőjének, de közreműködésükkel egy új, helyi StarOffice-kiadást is létrehoztak a Nyílt Forrás Közössége számára.

„Van egy szabály, amelyet immár elég sok ember megértett: ha megoldást keresel valamire, az már valószínűleg elkészült Linux alá. Ez hihetetlen dolog ezen a felületen” – mondta *Jaroslaw Szumski*, a Linux Plus szerkesztője. „Az üzleti világ is a Linux felé mozdul, mivel az emberek nagy feladatokhoz célszerű megoldásokat keresnek.”

Szumski elmondta, a kalózkodás elleni lépések következményeként az emberek más alapprogramokat keresnek az irodáikba. Az üzleti világban általában ez a Windows NT-t és az Office 98-at jelenti, a dolgok azonban egyértelműen afelé haladnak, hogy egyre több ember ismerkedik a StarOffice-szal vagy más irodai csomaggal.

Lengyelországban a Linux igazi bástyái azonban az egyetemek és az internetszolgáltatók. A PLD – ez a RedHat Linuxon alapuló, Kelet-Európára kihegyezett és az adott

➲ www.linux.pl

nyelvre lefordított terjesztése – megszületése szükségszerű következmény volt. Az olyan, műszaki iskoláikról híres városok, mint Varsó, Wroclaw és Krakkó diákjai és szakemberei úgy gondolták, hogy a legtöbb terjesztésből hiányzik az egyszerű lengyel kezelőfelület. „Egy lengyel nyelvű telepítőprogrammal

Mégsem adóztatják meg a Linuxot

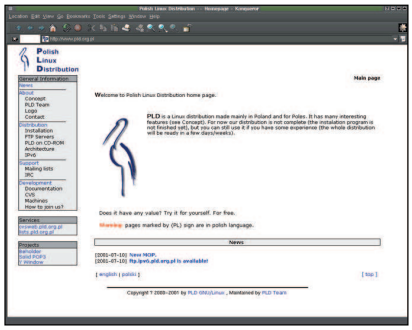
Lengyelországban adót vetettek ki a Linuxra
Egy cégre, mely linuxos kiszolgálókat telepített, új adótételt róttak ki, mert a GNU-programokat adománynak minősítették. Az összeg meghatározásakor a Nagy Cég programjait vették alapul. (2000. 11. 19.)

Linux-adó!
Lengyelországban adót szabtak ki egy vállalatra, amely Linux-kiszolgálókat üzemeltet, és StartOffice programokat használ. A lengyel adóhatóság azért szab ki adót a GNU-programokra, mert azok adománynak minősülnek, és így a felhasználók az azonos csoportba eső Microsoft-termékek árának harminc százalékát kötelesek az államkasszába befizetni. (2000. 11. 24.)

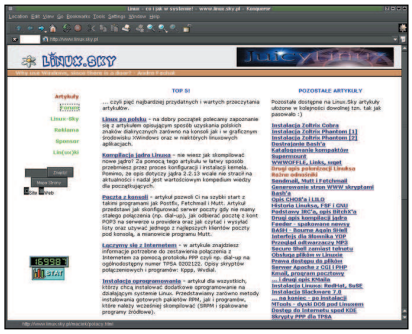
A Lengyel Országgyűlésben száz Linux X terminált állítottak munkába. (2000. 11. 30.)

Immár adómentesek a szabad programok Lengyelországban is
Lengyelországban nagy felháborodást váltott ki a hír, miszerint az egyik helyi adózási iroda adót akart kivetni a szabad programokat üzleti célra felhasználó vállalatokra. A lengyel pénzügyminiszter azonban a következőképpen válaszolt: „A GNU GPL felhasználási szerződés keretében terjesztett programok… a jövedelemadózásról szóló törvények értelmében nem eredményeznek megadóztatható bevételt az ilyen jellegű programok használói számára. Ennek fő oka az, hogy a mindenki számára ellenszolgáltatás nélkül nyújtott szolgáltatások anyagi értékének megbecslése teljességgel lehetetlen.” (2001. 05. 09.)

Az összeállítás a ➲ www.linux.hu hírei alapján történt.



➲ www.pld.org.pl



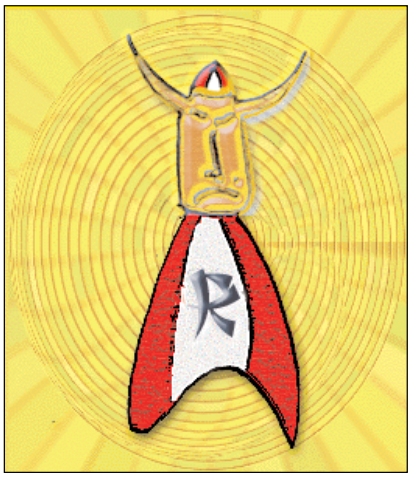
➲ www.linux.sky.pl/

kezdtük. A találmányok szülőanyja a hiány: ha a terjesztésben szükségünk volt valamire, beépítettük.”– mondta Kloczko.
A PLD-csoport munkája eljutott a Linux Documentation Projecthez is. Jelenleg a műszaki leírást fordítják lengyelre és a HOGYAN-okat frissítik.
A Lengyel Oktatási Minisztérium *Internetet az iskolákba* programjának bemutatkozásával a PLD beindult. Egyre-másra bukkannak fel országszerte az alkalmazások lengyel változatai. „A PLD serkenti a számítástechnikai változásokat. Nyilvánvaló, hogy rengeteg ember dolgozik rajta, így önfenntartó projektté válik az idők folyamán. Minél többet teszel meg, annál több teendőd marad” – vélte Kloczko.
A Linux önmagában is belép a piacra.
Az egyik olyan cég, amely nagy hullámokat kavart a Nyílt Forráskód világában, a Krakkó királyi városában székelő ABA volt (➲ http://www.aba.krakow.pl/). Vezető fejlesztésük, az ABA-X általános célú, hálózati vékony ügyfélprogram, tulajdonképpen egy beágyazott RedHat-alapú Linux-rendszer, amely a könnyű fejlesztés céljából Flash-ROM rendszeren fut.
Az ABA tulajdonosa, *Tomasz Barbaszewski* büszke a Lengyel Országgyűlésben nemrégiben elkészített munkájukra: csapata húsz munkaállomást telepített, és szerződésük van a Polish Customs céggel további kétszáz, jelszóolvasóval kiegészített ABA-X munkaállomás rendszerbe állítására.
„A Nyílt Forráskód az egyetlen útja annak, hogy ilyen rövid idő alatt ehhez hasonló dolgot alkossunk. Egy ilyen munkaállomáshoz szükséges kód megalkotása éveket vett volna igénybe” – mondta. „Operációs rendszernek nem a Windows CE-t választottuk, mivel nem lehet tudni, mi megy végbe az operációs rendszer belsejében, ráadásul még csak ki sem lehet deríteni. Inkább létrehozom a saját rendszeremet, mintsem egy olyan tömeggyártott telepítéstől kelljen függnöm, ahol a kinézet a legfontosabb, és nem az üzembiztonság.”
Barbaszewski azonban úgy gondolja, űr található a Nyílt Forráskódú Közösségben, ha a jó minőségű végfelhasználói alkalmazások kerülnek szóba.
„A Linux a hátsó ajtón, a kiszolgálók világán keresztül lépett be a számítástechnika színpadára, azonban még mindig nincsenek igazán jó alkalmazásaik. Láthatunk ugyan egy vagy két megoldást bizonyos ügyfelekre vonatkozóan, de ezek zárt fejlesztések, így kódjuk elérhetetlen” – vélte. „Tegyük fel, hogy üzlettel foglalkozol és Linuxot szeretnél használni. Van Linuxod, StarOffice-od és sok más egyebed. Szükséged van azonban számlázóprogramra, pénzkezelőre – valódi alkalmazásokra.”
A teljes Lengyel Nyílt Forráskód Barbaszewski panaszaitól visszhangzott. „Minden készen áll – jelentette ki Kloczko –, de senki sem szeretne olyan unalmas dolgokkal foglalkozni, ami a Linuxot egyszerűbbé tenné a végfelhasználók számára.”
Évekkel ezelőtt, miközben Lengyelország éppen az évekig tartó szovjet uralom alól lépdelt kifelé, a számítógépek – például az Atari 800XL és a TRS-80 – annyiba kerültek, mint egy kisebb autó, mainframe azonosítóra pedig csak az egyetemi professzorok vágyakozhattak. Most az ország négy nagy (Linuxot futtató) mobiltelefon-szolgáltatóval, számtalan (Linuxot futtató) ISP-vel és két nagyobb, Gdanskban és Czestochowában székelő (és természetesen Linuxot futtató) párhuzamos feldolgozást végző gépteleppel (parallel computing test bed) dicsekedhet. Ez az egyik leggyorsabban növekvő gazdaság a térségben és virágzik az üzlet a linuxos tanácsadói piacon is.
A középkorban a „sasfészek” kastélyok megépítésével Lengyelország első nemzeti hálózatát alkotta meg északkeleti irányban a célból, hogy az országot megóvja a betörésektől. Az egyes kastélyokban állomásozó katonák füstjelekkel tarthatták a kapcsolatot egymással. Ez a védelmi vonal évekig távol tartotta innen az ellenséget. Jelenleg a világ sokkal kisebbé vált, Lengyelország pedig gyorsan gyarapodik, így nagyobb szüksége van a hálózatfejlesztésre, mint valaha. Ahogy a Linux lassan gyökeret ver, a cél már nem az lesz, hogy az országot elvágják a külvilágtól és visszaverjék a támadókat, hanem éppenséggel az, hogy ezt a közösséget befogadják, a növekedést ösztönözzék, és életben tartsák a Nyílt Forráskódot ezen a hihetetlen vidéken.

John Biggs
író és tanácsadó Brooklynban, New York városában.

Linux a közoktatásban

A választható Linux



Sokak számára a Linux választási lehetőségként jelent meg a PC-ken jelenleg egyeduralkodónak számító Windows operációs rendszer mellett. Még sokan emlékezhetnek arra, hogy néhány évvel ezelőtt a Sulinet-gépeket kiszállító nagy cégek képviselői látványos grafikonokkal sulykolták a tájékozatlan tanáremberek fejébe, hogy a Windows NT kiszolgálóé a jövő, s az előadóteremben mindenki jól láthatta, milyen meredeken emelkedik az égbe a Windows NT kiszolgálók terjedését jelképező vonal. Heves igyekezetük ellenére a Windows NT-láz mára csillapodott, a grafikon vége lekonyult, és manapság már visszavonulóban van a linuxos kiszolgálókkal szemben. Egyes becslések szerint tavaly több kiszolgálón futott Linux, mint Windows NT.

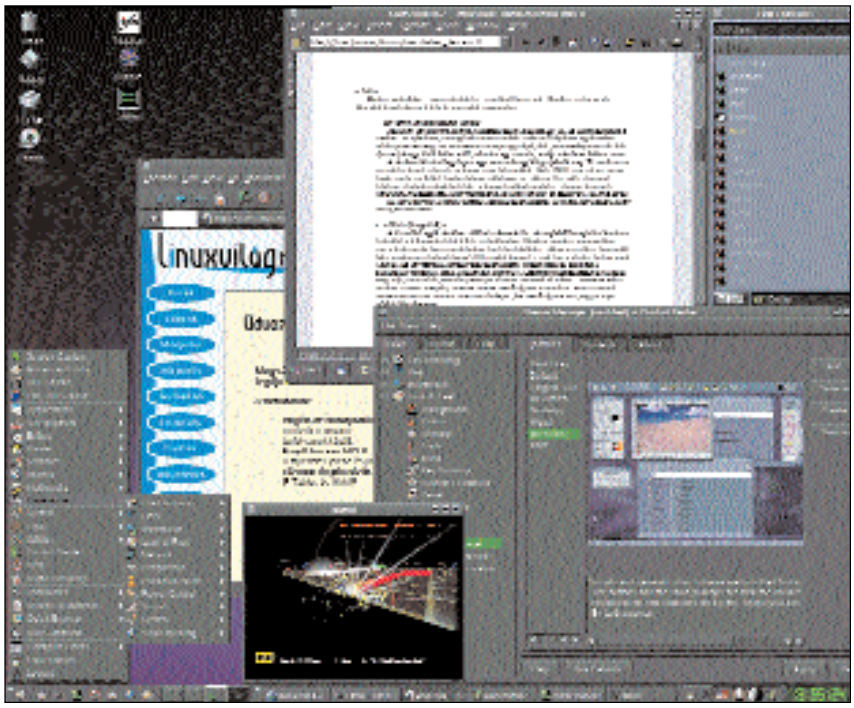
A személyi számítógépeken és a munkaállomásokon azonban más a helyzet. A linuxos fejlesztések máig sem érték be teljesen a windowsos munkaasztalt. A különböző Windows-változatokra írt programok ma még barátságosabbak, ritkán még üzembiztosabbak is, mint linuxos párjaik. Bár többfajta irodai alkalmazás közül választhatunk Linuxon is, azok egyike sem annyira kiérlelt, mint a Microsoft Office. Kis nemzet ide vagy oda, feszültséget okoz számunkra, ha elektronikus levelünk írásakor képtelenek vagyunk a Netscape levelező programjában a hosszú *ő* és *ű* begépelésére, és csalódást jelenthet az is, ha a program nem magyar nyelven beszél velünk, hanem angolul vagy németül, esetenként akár csehül vagy szlovákul, mert történetesen még senki sem fordította le magyarra. Jelenleg nincs hatékony magyar helyesírás-ellenőrző egyik linuxos irodai csomagban sem, bár a napokban megjelent SuSE Linux 7.2-ben forgalmazóik szerint a StarOffice 5.2 már tartalmaz magyar helyesírás-ellenőrző modult. Még nem volt alkalmam kipróbálni, hogy ez mennyire jó. Kevés az ingyenesen használható, elfogadható minőségű magyar betűkészlet a Linuxhoz, ami pedig elérhető, annak kinézete hagy némi kívánnivalót maga után. A linuxos munkaasztal azonban gyorsan fejlődik, és a saját fejlesztések mellett egyre több windowsos program jelenik meg Linuxon is. Pillanatnyilag úgy tűnik, hogy a nagy világcégek közül egyedül a Microsoft nem fejleszt Linux alá. Néhány szakértő úgy véli, hogy a Microsoft lassítani akarja a Linux terjedését, ezért nem írja át a Microsoft Office-t vagy akár az Internet Explorert Linuxra; mások szerint viszont nem tekinti elég nagy piacnak, hiszen úgy becsülik, hogy tavaly világszerte az otthoni számítógépeknek csak 4 százalékán futott Linux. Ma még nehéz megjósolni, hogy

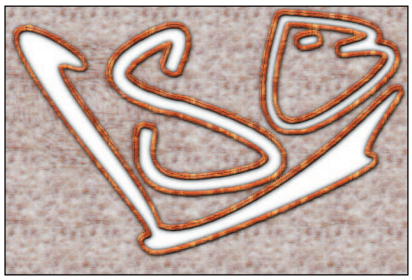
a jövőben milyen szerepet fog játszani a Linux a munkaasztalokon, azt azonban már ma is kijelenthetjük, hogy a Macintosh mellett a Linux vált a másik választási lehetőséggé.

A Linux nem egyetlen cég terméke, mégis rendkívül üzembiztos és jól használható operációs rendszer. Ennek egyik legfontosabb oka a nyílt forráskód, hiszen az esetleges programozási hibákat a Linux-közösség az Internet segítségével azonnal vagy rövid idő alatt felderíti és kijavítja. A kiérleltség másik oka a unixos örökség. A Unix az elmúlt évtizedekben a csúcsgépek méregdrága operációs rendszere volt, amit nagy teljesítményű szuperszámítógépekre és kiszolgálókra telepítettek. Ezek a drága gépek gyakran légkondicionált helyiségekben, az épület többi részétől elszigetelten működtek, kezelőik különleges munkaruhákat hordtak, és a hívatlan látogatók számára a számítástechnika felkent papjainak tűnhettek, akiknek a puszta megpillantása is borzongást keltett a mindennapi halandókban. Manapság azért hallunk egyre többet a Unix térhódításáról, mert a személyi számítógép az utóbbi néhány évben képessé vált arra, hogy ilyen üzembiztos rendszereket is futtathassunk rajta. Így válhatott a Unix és a Linux komoly választási lehetőséggé, legalábbis a kiszolgálókat illetően.

Drága az ingyenes Linux

Szabad választáson alapul a Linux terjesztési modellje is, hiszen bárki szabadon és ingyenesen letöltheti a programcsomagokat a számtalan kiszolgálóról – bár egy átlagos számítógép-felhasználó gyakorta képtelen telepíteni azokat, és végül elveszik a kiírt hibaüzenetek tengerében. Ezért kezdetben jobb valamilyen Linux-változatot megvásárolni, hogy az ember hamar rájöjjön:

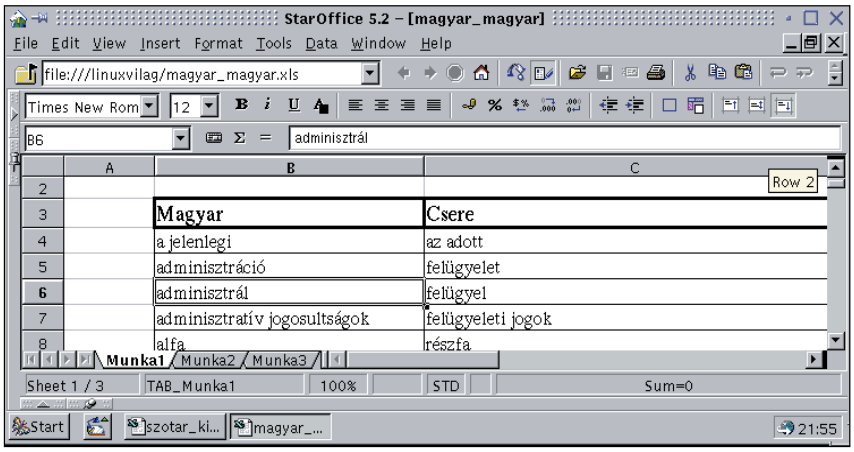




ez sem jelent tökéletes megoldást, mivel néhány hónap elteltével ismét meg kell tapasztalnia, hogy képtelen frissíteni programjait. Azt pedig mégsem várhatja el tőlünk senki, hogy négyhavonta húszezer forintot adjunk ki az amúgy felhasználóbarát és könnyen kezelhető SuSE-változatért. Érthetetlen vagy talán nagyon is érthető, hogy miért nincs egy- vagy kétlemezes SuSE-frissítés, hogy miért várják el tőlünk ugyanannak a négy kézikönyvnek az évente háromszori megvásárlását, hogy miért sóznak a nyakunkba számunkra teljesen használhatatlan DVD-t, amikor csak CD-olvasónk van… A RedHat esetén ugyanez a helyzet, csak négyszer ennyiért. A RedHat 7.0-s változat kétségtelenül tiszteletet parancsolóbb volt, mint a SuSE 7.0, hiszen 15 lemezen adta szinte teljesen ugyanazt, mint versenytársa. A növekedett terjedelem annak köszönhető, hogy a biztonság kedvéért mindent kétszer másoltak fel a CD-kre, és tizenötödikként kaptam egy picurka CD-csodát, amivel máig nem tudok semmit sem kezdeni, hiszen Magyarországon még nem láttam olyan gépet, amivel használni lehetne. A Corel Linuxból egy maréknyi pingvin maradt az asztalomon emlékeztetőül; a linuxos Corel Office pedig még azt sem teszi lehetővé a számomra, hogy ékezetes magyar betűket gépeljek a WordPerfecttel. Az Applixware Office 5.0 kissé engedékenyebb, de még mindig nem tökéletes. Valami van, de még nem az igazi. S hogy mindez majdnem annyiba kerüljön, mint a Microsoft Office!

Ez a világ már nem az a világ

A Sulinet-program kezdetére a PC hazánkban egyeduralkodóvá vált. Szinte hihetetlen, hogy milyen lendülettel szorította ki az irodákból és az iskolákból az addig szintén vezető szerepet játszó Commodore 64-eseket és 128-asokat. A világ fejlettebb részein már akkor sem volt ennyire egyértelmű a helyzet, az IBM PC-k mellett többek közt ott voltak (és vannak) az Amigák és az Apple-gépek is. Aki grafikus programokat akart írni, Amigát vett, az Apple-gépek és az Adobe-programok pedig uralták a szerkesztőségeket és a nyomdákat. A Windows olcsó tömegcikk volt, ami elárasztotta a piacot, és a használhatóság érzését keltette a vásárlókban. A kis vásárlóerejű magyar polgár számára az ár-teljesítmény mutatóiban kedvező PC jelentette akkor és talán még ma is az egyetlen választási lehetőséget. Az a kellemetlen tapasztalat, hogy a Windows van hogy óránként lefagy, keveset nyomott a latban, hiszen minden számítógépházon ott volt a Reset feliratú gomb, amit bármikor meg lehetett nyomni.
A Windows újraindítása rövid csevegésre és kávészünetre adott alkalmat, a hazai ügyfél pedig hosszasan várt. Senkinek sem okozott gondot, ha hosszú sorok alakultak ki a bankokban vagy a hivatalokban. A magyar türelemre szoktatott nép.
A PC és a Windows hazai elterjedésének és egyeduralkodóvá válásának legfőbb okát én a magyar programvásárlási vagy inkább -szerzési szokásokban látom. Magyar ember nem vásárol(t) programot, hiszen az elmúlt több mint egy évtizedben hozzászokott ahhoz, hogy a programért nem kell fizetni, azt szabadon lehet másolni és bárki bárhová ingyen telepítheti. A felhasználási szerződéseket nem olvassuk el, hiszen azok veretes angol nyelven íródnak, és mint tudjuk, idegennyelv-ismeretünkkel sincs minden rendben. Aki nem akart, nem akar a boltba menni a programokért, annak PC-t kellett és kell vásárolnia „szerzett” Windows-rendszerrel, hiszen csak ehhez tudott viszonylag könnyen programokat felhajtani. Levonhatjuk tehát a következtetést, hogy sok magyar felhasználó immár régóta a „szabad” programok híve, hiszen számára az mindig is ingyenes termék volt, és az is marad.
Manapság azonban a törvénytelen programhasználatot üldöző Business Software Alliance (BSA), a szoftverrendőrség és az útszéli óriásplakátokra felragasztott bilincsek keltenek rossz érzéseket a többnyire becsületesnek született, feszengő felhasználókban, akik haladó világunkban egyre több helyen juthatnak hozzá ingyenesen a kereskedelmi programokhoz. Az Internetről kalózmásolatok mindig letölthetők, és hirdetésekben felkínált törvénytelen másolatokat rendelhetünk meg áron alul, postai kézbesítéssel. Egy tizenöt éves diákom is rendelt ilyen tiltott gyümölcsöt, és hamar fennakadt a szoftverrendőrség hálóján. Házkutatást tartottak nála és nyilvántartásba vették. A fiú érettebbé vált, és most már óvatosabban „szerzi be” az ismert világcég programjait. Változnak az idők és változunk benne mi is, de ez a fiú – aki gyakran látja magát *Bill Gates* lehetséges utódaként – máig azt vallja, hogy inkább a börtönt választja, mint a GNU hatálya alá tartozó szabad programokat. Egy másik tanítványom ismerőse két hónappal a Windows 98 piacra dobása előtt már feltelepíthette a hivatalosan még nem is létező legújabb változatot a gépére, és erre még büszke is volt. Rangot és szakértelmet kölcsönzött neki ez a rendkívüli tett. S valóban, ki ne érezné sértőnek, ha az egyszerű és pillanatok alatt lezajló programmásolást lopásnak minősíteném? Ezért inkább ezentúl törvénytelen programhasználatnak fogom nevezni – sokkal kíméletesebb megfogalmazás.
Aztán itt vannak a kapuk előtt a (fel)jelentő programok, amelyek időről időre jelentkeznek alkotóiknál, és különböző adatokat



szolgáltatnak. A korszerű programok telepítésük során bevallottan megkeresik a már telepített alkotóelemeket, s vajon mi akadályozná meg őket abban, hogy a merevlemez áttekintése után ne továbbítsák az összegyűjtött adatokat a központban lévő adatbázishoz – a sorozatszámokkal és az adott gép címével együtt? A Windowst használók gyakorta tapasztalhatják, hogy az ismert világcégek termékei időnként megkérdezés nélkül telepítgetni kezdenek gépeinkre a távolban lévő ismeretlen kiszolgálókról. Ilyet amúgy csak a vírusok tesznek. Mivel ezeknek a programoknak a forráskódja ismeretlen, bármi megtörténhet! A frissítő (s talán egyúttal jelentgető) programok ellen csak úgy lehet védekezni, hogy nem kapcsoljuk rá gépeinket a Világhálóra, de ez a számítástechnika fejlődésének tükrében egyre inkább képtelen és kivitelezhetetlen ötletnek tűnik. Miután egy nagy cég felépítette a törvénytelen példányok adatbázisát, csak át kell adnia a listát a kiszemelt ország hatóságainak, akiknek hivatalból el kell járniuk. Ki fog ilyenkor fizetni az iskolákban és a hivatalokban lévő kalózpéldányokért, már ha vannak ilyenek? A miniszter? A minisztérium? A fenntartó önkormányzat vezetője? Az iskola igazgatója, tanára vagy éppenséggel a diák? És ki fog ülni? Ezeket a kérdéseket legtöbbször mégsem a rendőrség teszi fel az intézmények vezetőinek, hanem a windowsos programokat értékesítő ügynökök, akik sejtelmes mosollyal kínálják ügyfeleiknek a portékájukat, és nyíltan vagy burkoltan börtönnel fenyegetik őket, hogy eladási mutatóik jobbak legyenek. Ezt a fenyegetésre alapozott üzletmenetet testesíti meg a Business Software Alliance, amely a törvénytelen programhasználatot hivatott felderíteni, illetve jelenteni a bűnüldöző szervek felé. A feltárt bűnt azonban nem feltétlenül követi bűnhődés, hiszen utólagos elismertetéssel meg lehet bocsátani azoknak, akik hajlandók visszamenőlegesen is fizetni. A világot azonban nemcsak a BSA fenyegető jelenléte változtatta meg, hanem az Internet is. Jó egy évtizeddel ezelőtt a gépek elsöprő hányada nem kiszolgáló vagy munkaállomás volt, hanem önálló munkaasztal, és csak néhányuk volt hálózatba kötve. Ma viszont egyre ritkább az olyan gép, amelyik legalább az Internetre ne lenne rákötve.

Ez új helyzetet teremtett, hiszen a klasszikus Windows teljességgel védtelenül hagyja a számítógépeket. A vírusok szabadon pusztíthatnak a Microsoft-logikával felépített programokban, és nagyméretű helyi hálózatokat tehetnek tönkre néhány óra alatt.

A Unix-típusú rendszerek a kezdet kezdetétől fogva hálózati környezetben fejlődtek, így teljesen alkalmazkodtak a hálózatok által támasztott követelményekhez. Senki sem tagadja, hogy az átlagfelhasználó számára bajos jelszavakat beírogatni, biztonsági kérdéseket latolgatni és alkalmazkodni a Unix szigorú előírásaihoz. Kétségtelen, hogy a védelem nélkül hagyott gépeket könnyebben kezelheti az átlagember, de időnként ezért a kényelemért nagy árat kell fizetni. Nem véletlen az sem, hogy az újabb Windows-változatok egyre inkább a Unix szellemiségét tükrözik.

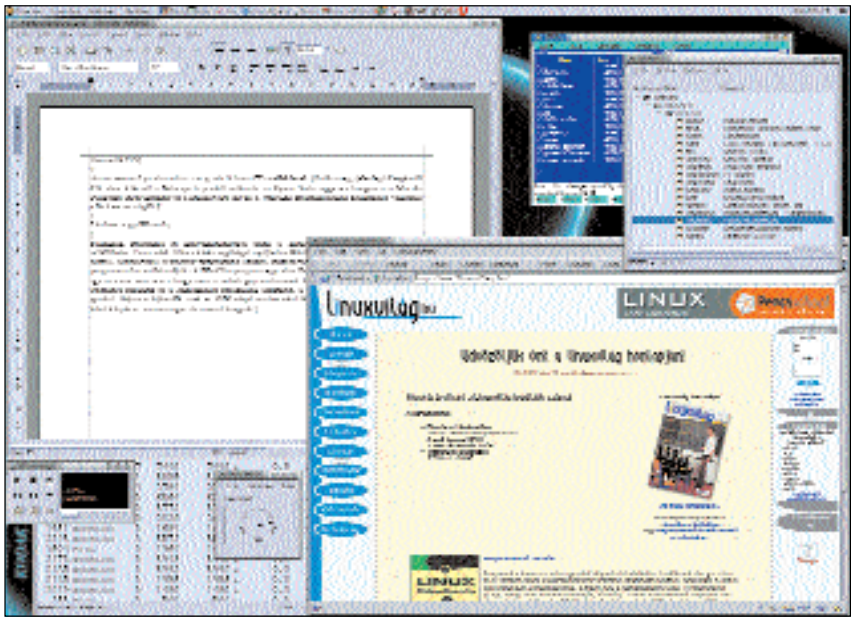
A Sulinet által biztosított iskolai hálózatok költségességük ellenére nem töltik be hivatásukat. A Windows NT kiszolgáló és Windows 98 munkaállomás összeállítások tapasztalatom szerint képtelenek a diákok munkájának védelmét biztosítani, hiszen az általuk létrehozott munkakönyvtárakat bárki megnézheti, sőt onnan bármit le is törölhet. Ilyen öszvér megoldásokkal még a rendszer egységességét sem lehet biztosítani, hiszen a gyerek véletlenül vagy akár szándékosan még a rendszerfájlokat is letörölheti. A világ megváltozott, de a Windows logikája és minősége a régi maradt.

A világ más lett a tekintetben is, hogy megváltoztak a programozási szokások. Manapság nagyon sok fejlesztőkörnyezetet a kezdetektől fogva felületfüggetlenre, azaz olyanra terveznek, hogy több operációs rendszeren is futtathatók legyenek. Példa lehet erre az ismert Opera böngésző, amit több változatban letölthetünk az Internetről (Windows, Linux, EPOC, BeOS, Mac, OS/2), vagy a Blender nevű háromdimenziós tervezőprogram (SGI, Sun, FreeBSD x86, Linux x86, Linux Alpha, PPC Linux, BeOS x86, Windows 95/98/2K/NT). A Windows jelenlegi súlyát mutatja, hogy elsőként rendszerint a windowsos változat készül el, a többit csak némi késéssel kaphatjuk kézhez, és gyakran hiányoznak belőlük olyan kényelmi szolgáltatások, amelyek a nagy testvér programjaiban fellelhetők. Az ilyen felületfüggetlen programfejlesztés első pillantásra nehéznek tűnhet, és valóban nem is könnyű, de nem annyira bonyolult, mint ahogy azt gondolnánk. A kisebb cégek rákényszerülnek erre a többletmunkára, hiszen csak így tudják növelni piaci jelenlétüket. A Microsoft óriási piaci részesedésével nem törekszik a felületfüggetlen fejlesztésre, és ahogy egyes szakértők megjegyzik, a redmondi óriás az egyetlen olyan cég, amelyik legszívesebben csak egy operációs rendszer számára ír programokat: a Microsoft Windowsra.

A felületfüggetlenség iránti igény növekedését mutatja, hogy olyan dokumentumformátumok terjedtek el, amelyeket minden felületen meg tudunk jeleníteni, ha a megfelelő programok a birtokunkban vannak. Ilyen például a Rich Text, az Internet HTML nyelve vagy az Adobe cég PDF fájlformátuma. Az Internetről letölthető GIF vagy JPEG képformátumokat is minden operációs rendszerben meg tudjuk tekinteni. A hálózatba kötött, sokféle operációs rendszert tartalmazó gépek világában természetes igénynek tekinthető, hogy a felhasználó bármilyen formátumú hajlékony- vagy merevlemezt el tudjon olvasni. A Linux sokfajta lemezformátumot ismer, a Windows csak a saját formátumait ismeri fel.

Valóban csak egy operációs rendszer van?

A magyar iskolákban igen. Az Európai Közösség országaiban és Svájcban tett útjaim során azonban meggyőződhettem arról, hogy arrafelé vannak olyan iskolák, amelyek csak Macintoshokat használnak oktatási célokra, valamint az élet sok más területén is lehet ezzel a típussal találkozni. Egy nemrég elküldött MS Winword 6.0 formátumú tömörített dokumentumra kétségbeesett választ kaptam Rómából, mivel ottani ismerősöm nem tudta a levelemhez csatolt .DOC kiterjesztésű függeléket otthoni Macintosh gépén elolvasni. Idehaza Macintoshokat csak az Apple szakboltok kirakataiban vagy a számítástechnikai kiállítások bemutatóin látok hébe-hóba. Ennek anyagi okai vannak, hiszen ezek a minőségi



gépek a törvényesen megvásárolt programokkal együtt jóval többe kerülnek, mint a PC-k. Az a magyar iskola, amelyik ilyen gépeket venne, többé nem háríthatná át a programbeszerzés terhét a tanulókra, hiszen azok nem tudnának Macintoshokra írt programokat hozni közvetlen környezetükből. Egy önmagát megnevezni nem akaró iskolaigazgató felhívta a figyelmemet arra, hogy pályázataikban programvásárlásra irányuló kéréseiket a döntéshozók rendre elutasítják, ugyanakkor gépbeszerzési igényüket eddig minden alkalommal teljesítették.

A számítógépek világa a felületfüggetlenség felé halad. Belátható időn belül megszűnnek a Microsoft-monokultúrák, a következő nemzedékbeli programok esetében kiválaszthatjuk majd, hogy azoknak Windowsra, Macintoshra vagy Linuxra írt változatát kívánjuk-e telepíteni. Mint tudjuk, az átlagfelhasználó nem operációs rendszereket használ, hanem programcsomagokat, amelyek szöveg- és képszerkesztőket vagy irodai alkalmazásokat tartalmaznak. A titkárnő számára édes mindegy, hogy az általa használt és megszeretett szövegszerkesztő milyen operációs rendszeren fut. Az operációs rendszerek kiválasztása az iskolai és a hivatali rendszergazdák feladata lesz, ők pedig vagy szakmai alapon fognak dönteni, vagy egyéb szempontokat vesznek figyelembe. Engedhetnek a gazdasági erőfölénynek és az erőszakos megdolgozásnak, esetleg beszállítókként maguk is abban lesznek érdekeltek, hogy az állami intézmények minél többet költsenek programokra. Az otthoni felhasználók vásárlási szokásait az olcsóság mellett a programok harsány látványossága és a számítógépházak művészi kidolgozása fogja meghatározni. S ez valahol érhető is. Kívülről minden számítógép egyforma, a felhasználók többségét pedig valójában nem érdekli, hogy mi rejtőzik mögötte.

Ha a fenti piaci irányzatok továbbra is érvényesülnek, akkor két következtetést vonhatunk le: az operációs rendszerek és a kiszolgálók piacát a mainál jóval erőteljesebben fogják meghatározni a szakemberek döntései, míg az átlagfelhasználót a formatervezett számítógépek és a felhasználóbarát, több operációs rendszeren is futtatható programok fogják megérinteni. Hamarosan magukat az operációs rendszereket is egyszerre indíthatjuk el gépeinken, és tetszésünk szerint váltogathatunk közöttük. Ez nem holmi vágyálom, hiszen a VMware virtuális felület használatával erre már ma is lehetőség nyílik. Ez a program külön ablakokba hívja be a különböző operációs rendszereket, és még némi adatcserét is engedélyez közöttük. Már ma is számtalan emulátor létezik, ami virtuálisan megteremti azokat a feltételeket, amelyekre egy-egy program futásához szükség van. Például a Macintosh gépeken létezik olyan emulátor, amely PC-t utánoz, így lehetővé teszi a windowsos programok futtatását. Mint legtöbbször, itt is az anyagi lehetőségek szabnak határt az általunk egy időben használható operációs rendszerek számának, hiszen ez ma már nem annyira szakmai, inkább jogi és következésképpen pénzügyi kérdés, hiszen a szerzői jogi előírások miatt minden egyes feltelepített operációs rendszer után fizetni kell, ami igencsak nagy összegre rúghat, különösen, ha a futtatandó programokért is fizetnünk kell!

Jelenleg az általános iskolások és a középiskolát kezdők számára lényegtelen, hogy milyen operációs rendszert használhatnak iskolájuk számítógépein, mert mire megjelennek a munkaerőpiacon, ennek nem lesz akkora jelentősége, mint ma. Másrészt, ha valaki figyelemmel kíséri azt a folyamatos fejlődést, amit például a szövegszerkesztők esetében is tapasztaltunk, levonhatja azt a következtetést, hogy végső soron az sem számít, hogy diákjaink milyen szövegszerkesztőt használnak. A különböző szerkesztési műveleteket jelképező ikonok szinte teljesen ugyanazok, hasonlók az elnevezések, és a menük elrendezésében is felfedezhetünk némi szabványosságot. Tapasztalataim szerint nemcsak egy középiskolás, de még egy egyetemista sem használ olyan egyedi formázásokat, amelyeket Rich Text formátumban ne lehetne elmenteni, ezt a formátumot viszont minden fejlettebb szövegszerkesztő ismeri, fusson az akár Windowson, akár Linuxon. A szövegszerkesztők egyéb szolgáltatásai olyan többletet nyújtanak, amelyeket csak néhány ínyenc ismer és használ. Való igaz, hogy ma hazánkban (majdnem) mindenki törvényesen vagy törvénytelenül Microsoft Office-t telepít a windowsos gépekre, és a tanárok, sőt maguk a tanulók is azt várják el, hogy ez legyen az iskolákban. De mi a biztosíték arra, hogy a Microsoft Office örök és kiválthatatlan? Az 1990-es évek elején a Wordstar gyakoribb volt Magyarországon, és egy évtized múlva lehetséges, hogy a szövegfeldolgozás legelterjedtebb módja nem a gépelés, hanem a tollbamondás lesz. A szöveget a gép helyesírás tekintetében is ellenőrzi, formázza, majd – ha erre felszólítjuk – kinyomtatja. Horribile dictu még az is előfordulhat, hogyha gépelésre kényszerülünk, akkor sem a Microsoft Worddel fogunk szöveget szerkeszteni!

A következő részben többek között a Linux iskolai alkalmazhatóságáról, illetve oktatásáról, a szabad programok ellenzőiről és a különböző országok Linuxhoz fűződő kapcsolatáról fogunk szót ejteni.

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 fekete doboz 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, akkor rendszerint művészettörténész feleségével és kisiskolás lányával „találja szemben” magát.

AVI filmlejátszók és a felvétel

Robin folytatja a linuxos video- és hanglejátszók kipróbálását.

A múlt hónapban kipróbáltunk néhány MPEG-lejátszót, köztük az aKtion, a gmpeg, a gxanim, az MPlayer, a plaympeg, az XAnim, a xine és az Xtheater nevűeket. Megnéztük a Be operációs rendszert is, hogy videós képességeit összehasonlítsuk a Linuxéval.

E hónapban az aKtion, az Aviplay, az MPlayer és az XAnim AVI-lejátszókat vesszük szemügyre, és megkísérlünk felvenni néhány AVI fájlt is a Video4Linux és az XawTV, valamint a QtVidcap segítségével.

Az AVI (Audio Video Interleave) az egyik legelterjedtebb PC-s videofájl-formátum, ezt és a RIFF formátumot (Resource Interchange File Format) a Microsoft és az IBM még együttesen dolgozta ki.

A Microsoft-féle WAV hangformátum a RIFF fájl egy másik változata; utóbbi az Amiga szabványos formátumán, az IFF-en alapul. Ha az AVI-ról szeretnénk adatokat szerezni, jó kiindulópont lehet a John McGowan-féle AVI Overview, mely a ➲ http://www.jmcgowan.com/avi.html vagy a ➲ http://www.faqs.org/ címen érhető el.

A videofájlok képkockák sorozatából és kísérőhangból állnak. Videofelvétel lejátszásakor e képkockákat a lejátszó egymás után beolvassa, és oly módon keveri egymással, hogy végeredményként folyamatosan jelenjenek meg. Általában a hanglejátszást veszik alapul, ugyanis ehhez igazítják a mozgóképet. Ha a hang lejátszási sebessége ingadozik, fülünk a jelenséget egyfajta „nyávogásként” érzékeli, más a helyzet ugyanakkor a mozgókép sebességével: ennek csekély változásait a lejátszásnál nem szükségszerűen vesszük észre.

A mozgóképes fájlformátumokkal kapcsolatban két, egymással könnyen összetéveszthető fogalommal célszerű megismerkednünk: a szállítókkal (transzport) és a kódolókkal (kodekekkel). A szállítóban találhatók egymás után a soron következő hang- és filmdarabok. Az ismertebb szállítóformátumok között említhetjük meg az AVI-t, az MPEG-et, valamint a QuickTime-ot. A kódoló szolgál a szállítón belül tömörített mozgókép- vagy hangrészek beolvasára, ez képes értelmezni egy-egy darab jelentését. A legismertebb mozgóképkódolók a Cinepak és az Indeo, az AVI hangkódolók között pedig a PCM, az ADPCM és a GSM típusok fordulnak elő a leggyakrabban. Ha tökéletesnek látszó filmfájlunkból lejátszónk csak a képet vagy csak a hangot képes lejátszani, valószínűleg egy hiányzó kódoló lesz a hiba oka. A lejátszókat ugyanis úgy tervezték, hogy átugorják az olyan szállítórészeket, amelyeket nem értenek.

A Windows Médialejátszó

A WMP-t (Windows Media Player) a Windowszal szállított filmlejátszóként ismerjük – ez telepíthető kódolókat használ, amelyek bővítményekként, DLL-ek (Dynamic Link Libraries) formájában működnek. Ha új windowsos kódolót fejlesztünk ki, a mozgókép vagy hang WMP-ben is lejátszhatóvá válik. Linuxos kódolófejlesztők számára ez azt jelenti, hogy programjukat bővítményként átvihetik Windowsba, így a nyílt formátumot használó, Linuxban elkészített fájljaik megtekinthetők a Windows saját lejátszóján is. És ez igaz fordítva is: egyes linuxos videolejátszók windowsos bővítményeket használhatnak, valamint a Windows-formátumú fájlokat a nyílt forrású linuxos lejátszókon szintén lejátszhatjuk. Így lehetséges az, hogy egyetlen bővítmény mind Linux, mind pedig Windows alatt működik – legalábbis akkor, ha Intel-processzor dolgozik mindkét operációs rendszer alatt.

Az AVI fájlok beágyazott kodekjeik azonosítására egy bűvös számot, úgynevezett FOURCC kódot használnak. Ez a négybájtos kód teszi lehetővé, hogy az AVI szállító „tudja”, melyik kódolót kell betöltenie egy adatfolyam lejátszásához. Az egyediség szavatolásához a FOURCC-t be kell jegyeztetni a Microsoftnál. A nem hivatalos, de csaknem hivatalos érvényű FOURCC lista a ➲ http://www.webartz.com/fourcc/ címen található meg; itt még számos hasznos adat áll a rendelkezésünkre.

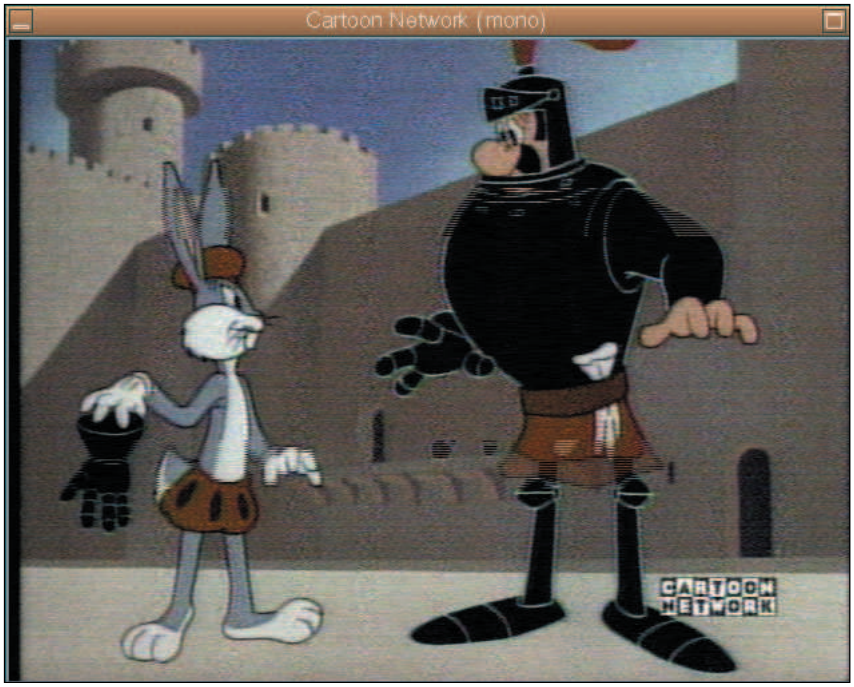
Elterjedt nézet, hogy az AVI fájlok 2 GB-nál nem lehetnek nagyobbak, illetve nem alkalmasak profi felhasználásra – ez azonban egyáltalán nincs így. A fájlméret korlátozásának fő oka az, hogy a RIFF blokk a méret megadásához 32-bites egészeket használ. Gyakran 1 gigás korlátot is emlegetnek, ugyanis a régebbi Microsoft Video for Windows-kód ezt veszi felső határnak (abban az időben már ez is hatalmasnak tűnt). Mivel egy előjel nélküli 32-bites egész 4 gigabájtos fájlméretet tárolhat, így ezt választották felső értékül – 4 GB egyébként a FAT 16 fájlrendszerben tárolható legnagyobb fájlméret is.

Mivel az eredeti AVI fájlszabvány fájlonként csak egy RIFF darabot enged meg, a méretkorlát legyőzésére kézenfekvő lenne a fájlonkénti több RIFF engedélyezése. Az Open Digital Media (OpenDML) Consortium éppen a Matrox ilyen irányú sürgetésére válaszolva megalkotta az OpenDML AVI fájlformátum-kiterjesztéseit. A szabvány az AVI Overview nevű honlapon ➲ http://www.jmcgowan.com/aviauthor.html #VidTrace-ben kereshető. Az OpenDML AVI támogatást a Matrox-féle DigiSuite program, valamint a Microsoft Windows Media Player is tartalmazza, utóbbi az 5.1-es változattól kezdődően.

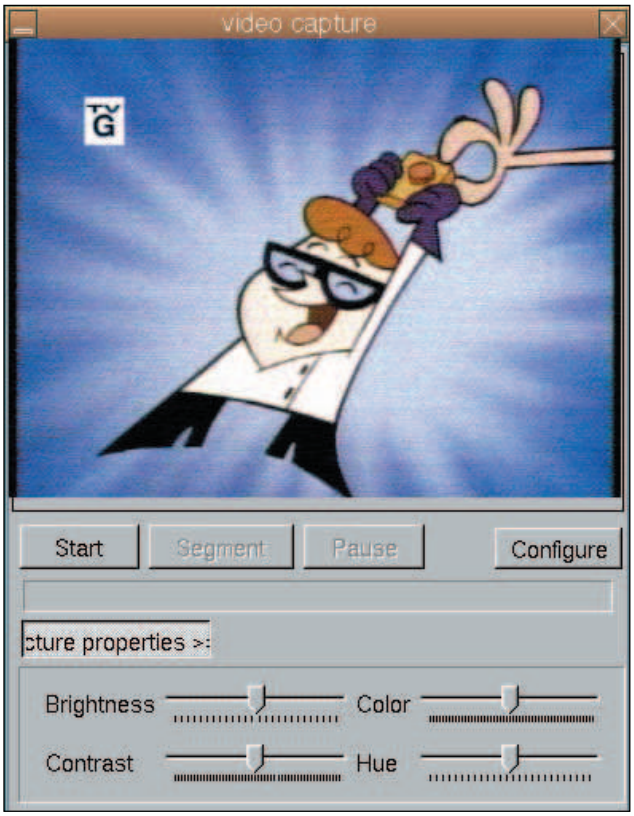
Az AVI mostanában időben veszített népszerűségéből, és ez az új Microsoft-formátumok, többek között az Advanced Streaming Format (ASF) és az újabb Windows Media Format (WMF) megjelenésének köszönhető. Ezeket a Windows Media Player 7-es változatába már beépítették, de a korábbi változatokkal is gond nélkül használhatók. A Microsoft 30 gigás WMF fájlokat próbált ki, de a formátum elméletileg 17 millió terrabájt adat tárolására is elegendő. A hangfájlok kiterjesztése .WMA, a mozgóképeké pedig .WMV.

Néhány linuxos videolejátszó

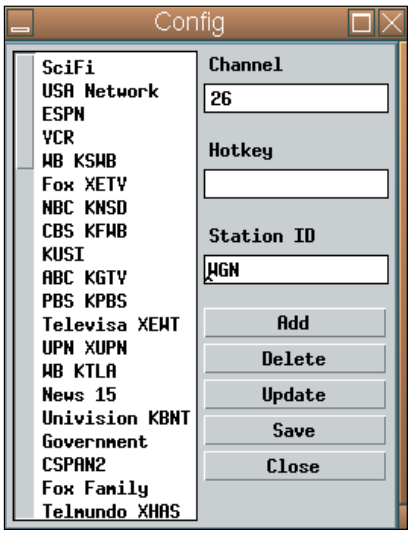
Az XawTV népszerű alkalmazás Video4Linux eszközökön történő filmnézésre. A Video4Linux (V4L) számos mozgóképrögzítő kártya használatát teszi lehetővé, melyeket az alkalmazásokban tetszés szerinti összeállításban használhatunk.
Az egyik legkedveltebb tévékártya a Hauppauge WinTV, a többi ehhez hasonló felépítésű kártya is ugyanarra a Conexant (korábban Rockwell Brooktree) lapkakészletre épül.
Az első Brooktree BT848-alapú tévékártyákhoz tartozó bttv linuxos meghajtókódot *Ralph* és *Marcus Metzler* készítette. Később *Gerd Knorr* vette át a fejlesztést, megalkotva a bttv-0.7.x sorozatú meghajtókat, továbbá ugyancsak az ő munkáját dicséri a XawTV, illetve ennek karbantartása is. *Alan Coxnak* köszönhetjük a Video4Linux névre hallgató egységesített videoszabványt is, melynek kifejlesztésére az OSS hangmeghajtókban található ötletek is hatással voltak: a V4L-et a bttv-meghajtó köré építette ki, majd beillesztette a Linux rendszermagba. Az eredeti bttv-meghajtó támogatása megszűnt és helyébe a *Justin Schoeman* által karbantartott V4L-változat került.
A V4L2 lényegében a Linux bttv-meghajtó újraírása. Justin Schoeman honlapja
➲ http://sourceforge.net/projects/bttv-v4l2/
szerint a bttv-v4l2 célja a Bt848/878-alapú videofelvevő kártyák támogatása nagyteljesítményű képfelvételek készítésében.
A V4L version 2 (V4L2) jelenlegi fejlesztéséért *Bill Dirks* felelős. Ő alkotta meg a felvevő API-t, valamint néhány változtatást is végrehajtott a különféle eszközökhöz történő jobb illesztés érdekében. Honlapja
➲ http://www.thedirks.org/v4l2/ további adatokkal szolgálhat.
Előző cikkünkben (a Linuxvilág 2001. májusi számában) rendszermagot foltoztunk és ebből telepítettük az új i2c illesztőprogramot, ugyanis Gerd Knorr honlapjának tüzetes átvizsgálása után észrevettük, hogy 2.3-as vagy újabb mag szükséges a működéséhez. A foltozás után zökkenőmentesen működött, csak azt nem értettük, miért. Végül a Bill Dirksszel folytatott beszélgetésünk oszlatta el a ködöt; kiderült, hogy mi is a dolga az i2c meghajtónak. A tévékártyák vevőt (tunert) tartalmaznak, ezek pedig saját i2c (vagy iic) nevű adatsínnel rendelkeznek, melyet a gyártó Phillips cég fejlesztett ki. Ez olyan háromhuzalos (órajel, adat, föld), szinkron soros adatsín, amely a vevő és a kártya többi alkotórésze közti kapcsolattartást végzi. Az i2c-vel a PCI adatsín beszélget. A PC alaplapján található egy másik i2c sín is, ez azonban a processzor hőérzékelőjével tartja a kapcsolatot.
Bár a V4L2 fejlesztése még kísérleti szakaszban van, Bill Dirks mégis ennek használatát ajánlja, amennyiben nem okoz gondot. A V4L sokkal inkább a bttv-re épül, de az alkalmazások bármelyik Video4Linux-változatot gond nélkül használhatják. A megfelelő eszközmeghajtót a videodev-meghajtó tölti be a V4L-be. V4L2-ben a meghajtó betöltője a videodev vagy a videodevx is lehet, a V4L2 videodev-csomag pedig csak 2.2.x magokkal működik és kizárólag V4L2- meghajtókat támogat. Az új V4L2 videodevx-meghajtó viszont egyaránt működik 2.2.x és 2.4.x magokkal, és mind a V4L2, mind pedig a V4L eszközmeghajtókat támogatja.
A V4L a magban található, a V4L2-höz telepítésre is szükség van. Ezt meg is tettük, miután a V4L Potatóval szállított változata elég régi. Végül mégis a V4L-et választottuk, mert tartottunk attól, hogy a V4L2 jelenlegi állapotában még nem megbízható. Tehát nem fogadtuk meg Bill tanácsát – valószínű, hogy a videofelvétel kipróbálásakor tapasztalt egyes nehézségeknek is ez



A XtawTV lejátszó



A QtVidcap

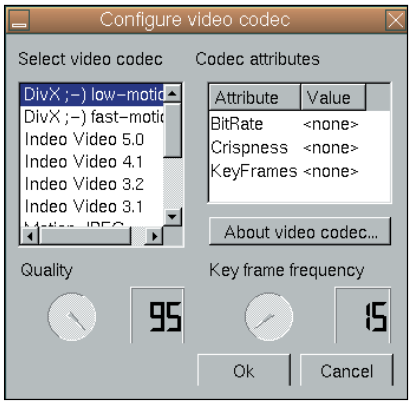


Az XawTV beállítóképernyője

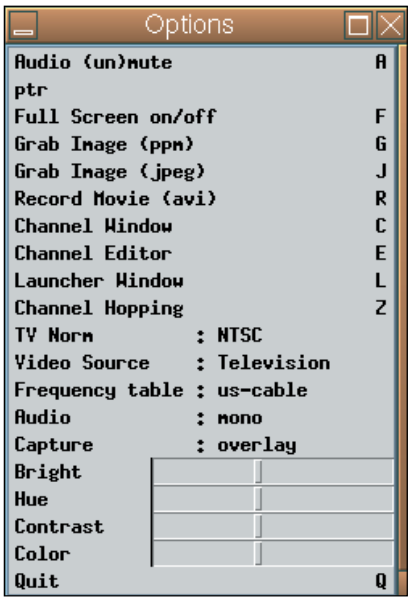
Az XAnim lejátszó

Az XAnim vezérlője

Az Aviplay



A videokódolók beállításai



Az XawTV beállításai

lehetett az oka…

Az AVI-lejátszókat forráskódból építettük fel, két okból is: a legújabb változatot akartuk, és nem állt a rendelkezésünkre .deb csomag.

Az eljárás ez esetben a következő: keressük meg a fejlesztést a ➲ freshmeat.net-en, a kicsomagolásra használjunk `tar xvfz`-t vagy `xvfI`-t (attól függően, hogy a fájl gzip vagy bzip formátumban van-e), majd építsük fel. Utóbbi a GNU-ban megszokott módon történt:

```
./configure
make
make install
```

Múlt havi értékelésünkben a MPlayer bizonyult a legmegbízhatóbb MPEG-lejátszónak, bár a birtokunkban lévő 0.11-es változat az AVI fájlokat nem játszotta le. Az új változat viszont ezekkel is elboldogul. Kiválóan működik, de az AVI-támogatás még nem elég fejlett. Bizonyos fájlokat nem sikerült lejátszanunk, néhány esetében pedig nem volt kép. Sőt, volt egy, amit fejjel lefelé játszott le!

Elméletileg az MPlayer kezeli az összes, a Microsoft Windows Media Player által lejátszható fájlt, hiszen a Linux /usr/lib/win32 könyvtárába bemásolt Windows Media Player kódolóbővítményeket is képes használni. A WMP bővítmények linuxos használatához szükséges kódrészletek viszont még készülőben vannak, az MPlayer kezelése pedig továbbra is kissé körüményes, a spártai környezetet biztosító parancssoros felület miatt.

A lejátszó ablakméretét nem tudjuk változtani, s ez a legtöbb olyan embert idegesíti, aki ahhoz szokott, hogy a kisméretű filmeket kétszeres nagyításban nézheti. Ezenkívül nem képes ismételt lejátszásra, a kilépés parancsra pedig meglehetősen lassan válaszol.

Az MPlayerre nem létezik többé változatszám; letölteni csak egy naponta frissülő CVS pillanatfelvételt lehet. A birtokunkban lévő gcc fordítóval kapcsolatos gondok miatt a program felépítése komoly nehézséget okozott számunkra. A gcc egy rejtélyes üzenetben valamilyen telepítési hibáról adott számot, és nem volt képes végrehajtani cc1plus-t, a g++ felületprogramját. A g++ telepítése látszólag rendben volt. A gcc újabb változatát (2.95.3-5) telepítve a hiba megszűnt.

Az XAnim programot sokan használják MPEG és AVI fájlok megtekintésére. Bővítményeket is használ, de nem a Win32 DLL fájljait, hanem saját, felületfüggetlen bővítményeket kínál a H.261, H.263, Indeo és Cinepak formátumú AVI fájlokhoz. Bár ez a lista nem túl hosszú, a leírás szerint a bővítmények Intel, Alpha, PowerPC, Sun és SGI gépeken is futnak. Az XAnim honlapja szerint a bővítmények saját, NDA alatti fejlesztések.

Az XAnim nálunk rendben működött, de nem volt képes annyi AVI fájl lejátszására, mint az MPlayer. Felhasználói felülete viszont jobb. A XAnim legutolsó változatát nem tudtuk kipróbálni, mert a felépítés során nehézségekbe ütköztünk: előbb ugyanis telepíteni kell az xmkmf nevű Imake-készítőt. Feltételeztük, hogy az X Window fejlesztő könyvtáraiban ez megtalálható, de nem akadtunk a nyomára. Később azonban észrevettük, hogy az MPlayer honlapja szerint ezek a könyvtárak lib6g-dev névvel szerepelnek a Debianban.

Két lejátszó maradt még hátra: az aKtion és a Kmpg. Az aKtion egy régebbi változatát sikerült elindítanunk, de ez hibázik az AVI fájlok lejátszása során. Nem tudtuk felépíteni a Kmpg 0.5.4 -et, ugyanis „nem szerette” a rendszerünk túlságosan új változatú libqt könyvtárát (nekünk 2.2.4 volt, a programnak pedig 1.x kellett volna). Az aKtion 0.4.1 esetében a „nem találom a KDE fejlécállományokat” hibaüzenet jelent meg.

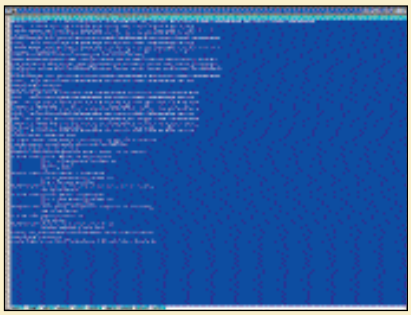
Az Xtheater 0.9.1-t sikerült felépítenünk,

ez viszont az AVI-kat nem játszotta le.
Az Avifile 0.53.5 olyan könyvtár, amelyet AVI fájlok lejátszásához használhatunk. Két példaprogram is található a csomagban: az Aviplay és a QtVidcap. Az Aviplay más felhasználói felületet nyújt, mint a XAnim, de a programmal lényegében hasonló tapasztalatokat szereztünk. A XAnim láthatóan nem használ Avifile-t, ahogyan az MPlayer sem, de mindkettő rendelkezik Win32 DLL-betöltővel, és megbízhatóbbnak tűnnek az Aviplaynél.
A QtVidcap felvevő programot ígéretes volta ellenére sem tudtuk működésre bírni, a fájlfelvétel kezdetekor felmondta a szolgálatot. Reméljük, hogy az XFree86 4.x és a V4L2 telepítése után ez is működni fog.
Az XawTV-vel sikerült AVI fájlokat rögzíteni, de az eredmény gyengébbnek tűnt annál, mintha ugyanezt a feladatot a Windows alatt működő Windows Media Encoderre bíztuk volna. Az XawTV csak 15- és 24-bites RGB- vagy MJPEG- kódoló használatával képes digitalizálni.
Az XAnim volt az egyetlen lejátszó, amely mindhárom XawTV formátumot le tudta játszani. A többi lejátszó a MJPEG-et leszámítva hajlamos az akadozásra, az Aviplaynek viszont egyik sem tetszett.
Gondjaink támadtak a fájlok Windows alatti lejátszásánál – a hiba talán a nem megfelelő felvétel miatt következett be. A linuxos lejátszók jobban működtek Windows alatt rögzített mozgóképpel, de csak akkor, ha a digitalizáláshoz használt kódoló nem volt túlságosan új.
A XawTV használata során két megoldandó feladatunk akadt: a képméret és a hang beállítása. Az ablakot a rögzítéshez használni kívánt felbontásnak megfelelő méretűre kell beállítanunk. Ha 15 képkocka/másodperces sebesség mellett 320x240 képpontnál nagyobb mérettel próbálkoztunk, akkor gondjaink támadtak: nem volt hang addig, amíg rá nem jöttünk, hogy a gmixerben a *line input for record*-ot (vonalbemenet használata a felvételhez) kell választani.
Következő írásunkban Debian Linux telepítésünket Potatóból Woodyra fejlesztjük, amely már a jobb multimédiás képességekkel bíró 2.4 magot és XFree86 4.0 GUI-t is tartalmazza. Eszközmeghajtóként pedig ismét V4L2-t fogunk használni.

Robin Rowe
a MovieEditor.com internetes és televíziós videoalkalmazásokat készítő cég egyik partnere. Írásai a Dr. Dobb's Journalban, a C++ Reportban, a C/C++ Users Journalban, a Data Based Advisorban jelentek meg és számos tanácskozás anyagában megtalálhatók. A Robin által készített programok sorában található többek az a közt kiszolgálóalapú videoszerkesztő rendszer, amit a Manhattan 24 órás televíziós hírcsatorna, a Time Warner New York One, illetve a kapcsolódó honlap ➲ http://www.ny1.com/ is használ. Elérhető a robin.rowe@ movieeditor.com címen.

Szerkesztők háborúja III.

Ki ne ismerné a jó öreg Norton Commander linuxos megfelelőjét, a Midnight Commandert? Az MC népszerűségét többek között beépített szerkesztőjének, az MCEditnek köszönheti.
Az MCEdit a CoolEdit egyik terminálos változata, de azzal ellentétben sajnos nincsen grafikus testvére. A névjegy is mutatja, hogy a CoolEdit „emberbarát" megfelelőjével van dolgunk. Számos különleges szolgáltatása mellett nagyszerűsége épp egyszerűségében rejlik. Bár szöveges alapú és eredetileg fekete-fehér, a -c kapcsolóval ismerős színeket varázsolhatunk a képernyőre. Külön jópont, hogy a funkcióbillentyűk (F1–F10) mind szövegszerkesztés, mind formázás közben segítik munkánkat.



Az F9 gombbal elérhető menük alatt egy sereg hasznos dolog rejtőzik: fájlokat szúrhatunk be tetszőleges helyre, szövegegységeket vághatunk ki, illetve mozgathatunk és másolhatunk egyszerűen. A sokféle választható sablon alapján dokumentumunkat megszépíthetjük, esetleg hasábokra tördelhetjük. Kiváló ötlet, hogy pillanatok alatt levélben is elküldhetjük az egészet. A szerkesztő széleskörűen támogatja a makrókat. Ha olyan, gyakran használt műveleteket szeretnénk elérni, amelyek az alapban nem szerepelnek, megírhatjuk ezeket parancsfájlként, és mint menüpontok beilleszthetővé válnak.
Az MCEdit kiválóan alkalmas C nyelvű programok írására, mivel a jól ismert élénk színekkel emeli ki annak parancsait és sorait. Ez alapján könnyen meg tudjuk különböztetni a nagyobb programrészeket, ciklusokat. Ha programfordításnál hiba történik, akkor megkapjuk a rakoncátlan sor számát: az `ugrás sorra` parancsot pont erre találták ki. Többek között ez a kis segédlet is nagymértékben hozzájárul ahhoz, hogy az MCEditet sokan használjuk mindennapi munkánk során. A különleges karakterek beszúrásán és az `ispell` helyesírás-ellenőrzésen már meg sem lepődünk. Ha az írógép híve vagy, a szerkesztő képes a sortöréseket hasonló módon megoldani.
Valamit véletlenül kitöröltünk vagy elgépeltünk? Aggodalomra semmi ok, erre is akad gyógyír: az `Undo` parancs (CTRL+U billentyű), amellyel előző lépéseinket visszavonhatjuk, ez pedig különösen hasznos lehet a szövegrészek kivágása és másolása során, ha néhány részlet nem oda került, mint ahova tenni szerettük volna.
Az MCEdit hiányosságaként az róható fel, hogy legfeljebb 16 MB méretű dokumentum megnyitására alkalmas, és ez néha kevésnek bizonyulhat a számunkra, de erre is létezik egyszerű megoldás: csak daraboljuk fel az egészet körülbelül 2-3 részre, mondjuk a `dd` parancs segítségével – és máris otthonos környezetben folytathatjuk a gépelést! Munkánk végén a `cat`-tel a darabokat egységessé fűzhetjük össze.
Összegzésül: legyél akár profi, akár kezdő, az MCEditet neked találták ki, mivel rendkívül alkalmas sokrétű feladatok megoldására.
Reszkess vi, itt a vetélytárs!

Kintera Vilmos (wolfbyte@freemail.hu)

Irány a mozi!

Linux a NAB-on

A következőkben a Las Vegasban megtartott National Association of Broadcasters (Műsorsugárzók Nemzeti Szövetsége) összejövetelén bemutatott linuxos videoalkalmazásokról számolunk be. Minden évben a tévétársaságok vezetői, az adásvezetők, a mérnökök és mindenki más, akit érdekel a profi videózás, Las Vegasba utazik a NAB-ra. A NAB a világ egyik legnagyobb árubemutatója, ahol több mint 115 ezer látogató és 1700 kiállító vesz részt. A számítógép-rajongók ennek ellenére elképzelhető, hogy jobban ismerik a CES-t (122 ezer látogató és ezer kiállító), illetve a COMDEX-et (225 000 látogató és 2300 kiállító). Területre mindhárom kiállítás közel azonos méretű: 108900 négyzetméter – aki nekivág, annak túrabakancs kötelező! A NAB-on a Linux a tavalyihoz képest sokkal nagyobb mértékben képviseltette magát. Számos új képhatásrendszert mutattak be, többek között láthattuk a RAYZ-t a Silicon Grailtől, a Shake-et a Nothing Realtől és a Mayát az Alias/Wavefronttól. A Linux Media Arts a Cinelerra Quicktime videoszerkesztőt és a Kino DV-szerkesztőt állította ki. Az AMD rövid bemutatót tartott videofejlesztők számára arról, miként lehet az Athlonból a legtöbbet kihozni. A BOXX Technologies a 3DBOXX grafikus munkaállomást állította ki. A videokártya-gyártók a Hauppauge-et és a DVS-t ismertették meg velünk. A linuxos asztali készülékek frontján pedig az ATI, az OpenTV és a Phillips TiVo volt jelen. A Khronos és a ProMpeg Forum is jelentős Linux média-API-kkal jelentkezett, ezenkívül a Radio Free Asia Linux és MP3-bemutatót is tartott a NAB-gyűlésen.

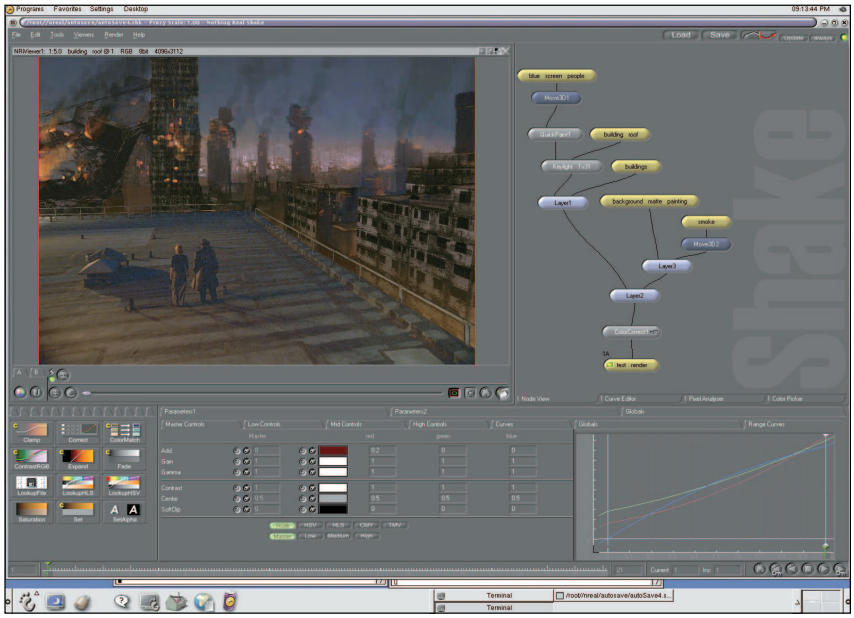
„A legtöbb profi filmgyártóstúdió a következő 18 hónap során átáll Linuxra” – vélte *Ray Feeney*, a Visual Effects Society műszaki bizottságának ➲ http://www.visual-effects-society.org/ négyszeres Academy Award-nyertese. A VES tagjai között találhatók az ILM, a DreamWorks, a Pixar, a PDI, a Disney’s, a Secret Lab és a Sony cégek különleges kép- és hanghatásokkal foglalkozó művészei. A VES-tagok most azt vitatták meg, hogyan állhatnának át Linuxra. Éveken keresztül ugyanis az SGI IRIX volt a stúdiófilmek első számú operációs rendszere, és amióta áttértek Windowsra, sokan elégedetlenek a támogatottsággal. Minthogy a Linux nyílt forrású rendszer, a filmipar igényeinek megfelelő programokat fejleszthetnek rá, és mivel mindenütt jelen van, sokkal könnyebb olyan felhasználókat találni, akik jobban ismerik, mint az IRIX-et. „Itt és most senki sem vásárol” – mondta Feeney a NAB-on – „de ha egyszer véget ér a forgatókönyvírók sztrájkja, komoly átrendeződésre számíthatunk. Az emberek jelenleg Linuxot használnak a leképezéshez, így a dolgoknak ezt a részét megoldottuk és kézben tartjuk” – hangsúlyozta Feeney. Összehasonlította a linuxos grafikus munkaállomások jelenlegi szintjét a két évvel korábbival, amikor először használták fel őket nagyobb mennyiségben a Titanic című film leképezési munkálataihoz. Szerinte két éven belül a Linux nemcsak a kiszolgálók, hanem a grafikus munkaállomások területén is egyeduralkodóvá válik. Az ILM-nél már Linuxot használnak a munkaállomásokon és a leképezéshez egyaránt.

Feeney egyébként a Silicon Grail alapítója is. Ez a társaság mutatta be legújabb nemzedékbeli RAYZ szerkesztőjét a NAB Compaq-pavilonjában. A RAYZ, melyet különleges hatások készítésére terveztek, Linuxon, IRIX-en és Windowson egyaránt használható. Az eredetileg filmen rögzített felvételeket előbb filmletapogató, például Kodak Cinenon segítségével digitalizálják. Bár minden egyes kép külön képfájlban kerül tárolásra (például: Targa formátumban), a RAYZ felhasználója számára ezek végeredményükben mozgóképpé állnak össze. A hatásszűrők a könnyű kezelhetőség céljából balról jobbra sorakoznak a grafikonablakban, míg az eredmények a videolejátszó ablakban láthatók. A kész filmek visszaírhatók film- vagy videoszalagra.

A Maya linuxos változata

A RAYZ a Silicon Grail legújabb interaktív szerkesztője, mely a cég Chalice nevű programján alapul. A Chalice használatával készült a Háborgó mélység, az Egyiptom hercege, a Star Trek: Űrlázadás, a Titanic, a Fantasia 2000, a Sötét zsaruk (Men in black) – és a lista folytatható. Rendelkezik különleges hatásokkal és színjavító eszközökkel is. Egy jellegzetes összeállításban a RAYZ ára 9900 dollár és a program májusban került piacra ➲ http://www.silicongrail.com/.

Az SGI-csoporthoz tartozó Alias/Wavefront által kiadott Maya széles körben használt profi térbeli animációs és képhatások létrehozására alkalmas programcsomag



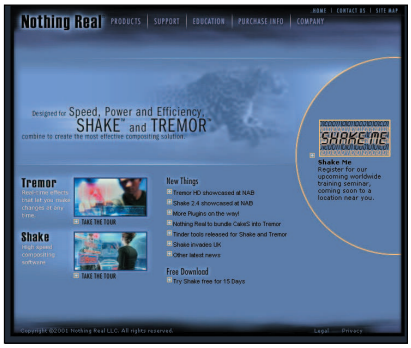
A Shake 2.4 linuxos változata

(lásd bővebben a 108. oldalon). A NAB-on az Alias/Wavefront bejelentette a Maya 4-et, melyben a leképezéssel, a szereplőanimációval, az ecset- és festőeszközökkel, valamint a játékkészítéshez használható eszközökkel kapcsolatos számos újítás található. A Maya nemlineáris szerkesztőeljárását is bővítették: most már időtorzítást (time warping), karakterösszefűzést, húzd-és-dobd-rendszert, valamint karakterkészlet-szerkesztést is találhatunk benne. Az új szereplőanimációs lehetőségek között megemlíthetjük az előre, illetve hátra mozgás közötti átkapcsolást, a mozgásnyomjelzést, a szellemhatást és a szereplők izmainak remegtetésére szolgáló „reszketőtorzítót” (jiggle-deformer).

„A fejlesztés menetét a felhasználói igények, kérések befolyásolták” – fogalmazott *Bob Bennett*, az Alias/Wavefront szórakoztatóipari üzletágának igazgatója. A döntéseket emellett a Visual Effects Society műszaki bizottságától érkező javaslatok is befolyásolták. *Linus Torvalds* a linuxos változat 2000 nyarán történő bejelentésekor a Maya 3-at „a Linuxon valaha is futott legösszetettebb és legnagyobb teljesítményű térbeli grafikus alkalmazásnak” nevezte.

A Mayát használta az ILM a Viharzónához, a Secret Lab a Mars mentőakcióhoz, a Sony Pictures Imageworks az Árnyék nélkülhöz és sok más alkotáshoz. Az Alias/Wavefront bemutatta a NAB-on a Maya 4 IRIX-os és windowsos változatát, a linuxos változat azonban később készült el. Az árak 7500 dollárnál indulnak (lásd ➲ http://www.aliaswavefront.com/).

A kaliforniai Nothing Real vállalat Shake-je gyors szerkesztőprogram, amely leginkább különleges képhatások létrehozására alkalmas. A Shake 2.4 új lehetőségei között szerepel a vektoralapú festés, a fejlett színjavító eszköztár, egy új rotoszkóp, valamint a használatot egyszerűsítő fejlesztések is. A program felbontásfüggetlen és önműködően kezeli a különböző bitmélységeket (8, 16, 32) és felbontásokat (Web, 601, HDTV, film, IMAX).



➲ http://www.nothingreal.com/

A Shake-et több mint hatvan filmben használták, bemutatkozása óta pedig minden olyan film elkészítésében fontos szerepet játszott, amely a különleges hatásokért járó Oscar-díjat elnyerte (Titanic, Csodás álmok jönnek, Mátrix, Gladiátor). A Shake fut Linuxon, IRIX-on és Windowson is.

A 2.4-es változatot februárban próbálták ki és röviddel a NAB előtt került piacra. Az ára 9900 dollárnál kezdődik (lásd ➲ http://www.nothingreal.com/).

Az ugyancsak kaliforniai Linux Media Arts kulcsrakész videoszerkesztő- és médiafolyam-rendszereket készít filmes és internetes felhasználásra. *Mike Collins*, az LMA elnöke szerint: „Célunk, hogy a Linuxot a világ első számú multimédiás szerkesztő- és gyártófelületévé tegyük, főként nyílt forrású programok felhasználásával”. Collins szerint célkitűzéseik között elsőként az szerepel, hogy kiszolgálókat és szerkesztőket készítsenek egy új, csúcsminőségű SDDI kártya számára, melyet a NAB-on jelentettek be. Eddig M-JPEG és DV-rendszereket kínáltak. Az LMA DV- és Quicktime-alapú szerkesztőrendszereit mutatta be. A csomagban a Cinelerra és Kino videoszerkesztőket, a Blender 3D-t, a Gimpet, a CorelDraw-t és a RedHat Linuxot találjuk. A rendszerek AMD, Intel és Compaq Alpha processzorokra épülnek. Az árak 1395 dollártól indulnak, 1 GHz-es AMD processzorral ➲ http://www.linuxmediaarts.com/.

A Kino egyszerű, csak vágásra szolgáló DV

➲ http://www.linuxmediaarts.com/

➲ http://www.aliaswavefront.com/

videoszerkesztő. A DV formátumot használja a legtöbb amatőr digitális kamera. Szerzője, a német *Arne Schirmacher* szerint „a Kinóval DV-kamerával rögzített filmeket lehet felvenni, létrehozni, elmenteni, szerkeszteni és lejátszani. Bár vannak ablakai és menüi, valójában billentyűzetről vezérelt program. Számos billentyűzetes parancsot használ, melyek a vi szövegszerkesztő parancsaihoz hasonlóak.”

➲ http://www.schirmacher.de/arne/kino/0.)

Jason Howard, az LMA munkatársa a Kino 0.4 bemutatásakor arról számolt be, hogy ez

➲ http://www.tivo.com/

a videoszerkesztő a DV-ről DVD-re történő átírásra szolgáló eszközök egy részét képezi. Howard a dvcont nevű, DV-kamerát vezérlő eszköz fejlesztőjeként ismert. A szintén Schirmacher által írt dvgrab az 1394 protokollt használva a filmet a kameráról a számítógépre viszi át, a dvsend pedig a szerkesztett anyagot írja vissza a kamera szalagjára. Howard egyúttal a Spectsoft munkatársa is. A jelenleg vegyes Windows/Linux-rendszerekkel rendelkező filmes cég munkatársa, *Ramona Howard* (Jason Howard édesanyja) lelkesen szervezi a teljes átállást Linuxra, hogy így jobban ellenőrizhesse az alkotófolyamatot. Filmproducerként Jasont bíztatja arra, hogy még több nyílt forrású programot fejlesszen. Jason Howard szerint „ehhez mindössze egy olyan ember kell, akinek Linux a szenvedélye, továbbá nem árt néhány láda üdítő sem!" Howard jelenleg egy olyan programon dolgozik, amellyel a mozgókép átalakíthatóvá válik az AVI és a Quicktime formátumok között, így téve lehetővé az anyagoknak a Kino és a Cinelerra közti egyszerű cseréjét. Egy olyan eszközön is dolgozik, amely az anyagokat egy szerkesztési lista alapján válogatja ki a szalagról és csak a szükséges részleteket viszi át a merevlemezre ➲ http://www.spectsoft.com/.

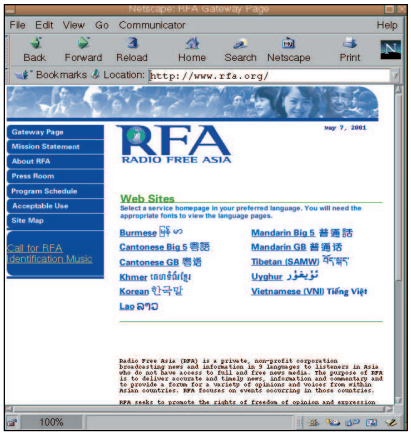
A Cinelerra a Heroine Virtual Quicktime-alapú videoszerkesztője. A cég jól ismert a linuxos közösségen belül a Broadcast 2000 szerkesztő és az Xmovie lejátszó révén. A Cinelerrát a NAB-on jelentették be és bemutatására is itt került sor. Ez a felbontásfüggetlen szerkesztő több mint száz valós idejű kép- és hangszűrővel támogatja az ötcsatornás hangot, és kihasználja a többprocesszoros gépek lehetőségeit is. A Heroine Virtual a programot a Linux Media Arts ➲ http://heroines.sourceforge.net/ által gyártott rendszerekhez adja.

Az AMD processzorgyár a NAB-on a videofejlesztőket tájékoztatta arról, hogyan használhatják ki az általuk gyártott Athlon és Duron processzorok előnyeit. *Jim Bovenzi* filmes fejlesztőmérnök szerint „a többprocesszoros rendszer a nagy teljesítményű videoalkalmazások kulcsa. A két Atthlon processzorral rendelkező gépek hihetetlen teljesítményt nyújtanak." Az AMD továbbá a Sasem-féle OnAir DTV 1080i HDTV kártyát is bemutatta, mely Windows XP-vel futott (lásd ➲ http://www.amd.com/ és ➲ http://www.sasem.com/).

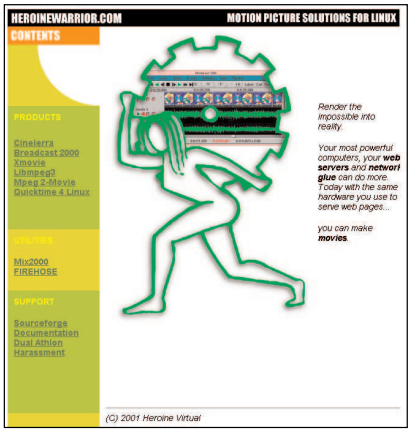
A WinTV-HD Hauppauge gyártmányú HDTV kártya. A Hauppauge WinTV-kártya nagyon népszerű a Linux-felhasználók körében, de egy Hauppage-nél dolgozó mérnök szerint sajnos a WinTV-HD felépítése teljesen más. A linuxos közösség így ismét egy fordított tervezési feladat előtt áll: a NAB-on bemutatott WinTV-HD kártya meghajtóit ugyanis szükséges lesz elkészíteni, mivel a Hauppage nem tervez a termékhez linuxos támogatást. A WinTV-HD kártya 399 dollárba kerül ➲ http://www.hauppauge.com/.

Egy a NAB-on bemutatott hasonló kártya volt az accessDTV, 479 dollárért ➲ http://www.accessdtv.com/.

A HDStationOEM tömörítés nélkül működő, HDTV valós idejű kimeneti, illetve bemeneti kártya, profi célokra. A gyártó DVS Digital Video linuxos meghajtóprogramokat is kínál, ezek azonban a 4000



A Radio Free Asia honlapja



➲ http://www.heroines.sourceforge.net/

➲ http://www.khronos.org/

dolláros árcímke miatt kifejezetten a profi piacra valók. A kártya az 1080, sőt, a 2000 képpontos felbontást is támogatja ➲ http://www.digitalvideosystems.com/.

A Phillips kínálta linuxos TiVo, valamint egy új, az ATI Technologies cégtől származó asztali eszköz is kiállításra került, utóbbi magvát az All-In-Wonder képezi. Bemutattak egy Linux PC-n futó tervezőrendszert is. Az ATI linuxos meghajtóprogramokat is kínál új, nagy teljesítményű, 1995 dollárba kerülő Fire GL4 grafikus kártyájához.

Az asztali eszközök fejlesztésével foglalkozó OpenTV bejelentette azon szándékát, hogy a jövőben támogatni kívánja a Linuxot (lásd ➲ http://www.tivo.com/, ➲ http://www.ati.com/ és ➲ http://www.opentv.com/).

A 3DBOXX nagy teljesítményű windowsos

A 3DBOXX

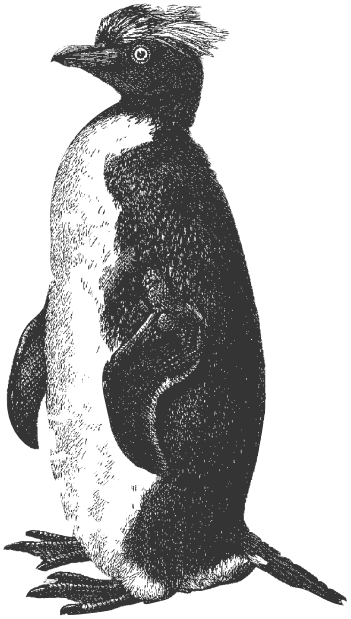
és linuxos munkaállomáscsalád, melyet a gyártó BOXX Technologies olyan digitális alkalmazásokhoz ajánl, mint a film-, HDTV-, videoszerkesztés és a játékfejlesztés. A 3DBOXX linuxos gépeinek egyik felhasználója a Blur – egy különleges képhatásokkal, animációval és tervezéssel foglalkozó stúdió. A cég a Paramount Parks részére készíttetett egy négyperces, térbeli, sztereó 70 mm-es rövidfilmet, a „7th Portal"-t.
A 3DBOXX rendszerek 2309 dollártól indulnak, a RenderBOXX kiszolgálórendszerek pedig 2692 dollártól (➲ http://www.boxxtech.com/).
Az OpenML egy új API, melytől azt remélik, hogy ugyanazt nyújtja a videofejlesztőknek, mint az OpenGL a grafikus fejlesztőknek. A Khronos csoport élenjáró, grafikával és digitális médiával foglalkozó társaságokból áll, így például a 3Dlabs, az ATI, a Discreet, az Evans & Sutherland, az Intel, az nVidia, az SGI és a Sun Microsystems is a tagjai sorába tartozik. A csoport a NAB-on beharangozta, hogy befejezték az OpenML 1.0 szabvány kialakítását. Az OpenML a dmSDK-n alapul, melyet az SGI nemrég tett nyílt forrásúvá, továbbá az MLdc-n – egy megjelenítő eszközökhöz készült közvetítőrétegen. A képhatások létrehozására szolgáló programokat gyártó Discreet segít a követelmények megállapításában. Az újdonságot bejelentő *Neil Trevett* (3Dlabs) sokat vár a rendszer nyílt forrású, linuxos megvalósításától (lásd ➲ http://www.khronos.org/ és ➲ http://oss.sgi.com/.)
A NAB Pro-MPEG fórumán jelentették be az AAF (Advanced Authoring Format) programfejlesztő-készlet 1.0-s változatának kibocsátását. A programot mozgókép-, hang- és metaadatok cseréjére szánják, utómunkálati célokra. (lásd ➲ http://www.aafassociation.org/ és ➲ http://sourceforge.net/projects/aaf/).
A kiállítókon kívül a NAB másik érdekessége a „Linux és az MP3 az adattárolásban" címet viselő ülés volt. A Radio Free Asia (➲ www.rfa.org) rövidhullámon és a Weben sugároz adásokat tibeti, kantoni, ujgur, burmai, vietnami, laoszi, khmer (Kambodzsa számára) és koreai nyelven (Észak-Korea számára). Az RFA magántársaság, melyet az Egyesült Államok kongresszusa alapított abból a célból, hogy a szabad sajtót nélkülöző országok lakosságát hírekkel és adatokkal lássa el. Első adásuk 1996 szeptemberében közvetítették Kína felé. Az RFA napi 24 órán át sugároz, kilenc nyelven.
A. J. Janitschek, az RFA gyártástámogatási igazgatója és *Tom Hallewell* vezető mérnök ismertette az RFA adattárolási eljárásainak történetét. „A Radio Free Asiánál minden adásunk tárolására WAV fájlokat használtunk, 4 GB DAT szalagra rögzítve" – mondta Janitschek. Ez azonban drága volt, munkaigényes és a visszakeresés is kényelmetlenségekkel járt. „Jelenlegi rendszerünkkel 100 órás műsort tudunk tárolni egyetlen CD-ROM-on, és bárki számára elérhetővé tehetjük, akinek a PC-jén MP3-lejátszója van" – tette hozzá Hallewell.
Az RFA kb. 15 CD-ROM-ot használ fel egyhavi hanganyag (1020 óra) rögzítéséhez. Három különböző kódolási beállítást használnak:
1.: 32 kHz, 48 kb/mp (20 MB/óra) a rövidhullámú rádiózásra;
2.: 16 kHz, 16 kb/mp (7 MB/óra) a hosszúidejű tárolásra;
3.: 12 kHz, 14 kb/mp (6 MB/óra) az internetes adásokra.
Az RFA nem PC-n kódolja az MP3-as adatfolyamokat, hanem a Telos Audioactive Realtime MPEG Internet Audio Encoder nevű kódolóját használja. Ez a szekrénybe építhető egység MP3-fájlokat sugároz, melyeket linuxos tárolórendszerük rögzít és CD-re ír. Janitschek szerint a gépi kódoló előnye a valósidejű működés és az a tény, hogy rendelkezik az MP3 szabadalmi engedélyekkel (a Fraunhofer és Thomson részéről). Az egység ára 2800 dollár (lásd ➲ http://www.audioactive.com/).
Az RFA nemcsak mentésre használja a Linuxot, hanem nyílt forrású programmal hozzá is járul fejlődéséhez. Az R-BOSS (Radio-Broadcast Open Source System) rendszer digitális adásanyag-kezelő alkalmazások gyűjteménye, melyet Pythonban írtak. Beszerezhető a 3D-Project nevű ingyenes csomagjuk is, melyben a műsorsugárzáshoz kapcsolódó térbeli rajzok, anyagok és mintázatok találhatók. További információ, letöltés a ➲ http://www.techweb.rfa.org-ról lehetséges.
A fentieken kívül a NAB-on még számos linuxos kiállítóval találkozhattunk. A Real Networks például a RealMedia formátum linuxos sugárzó- és lejátszórendszerét mutatta be. A Kasenna a Linuxot támogató vagyonkezelő rendszerével érkezett.
A Thomcast Communications a DCX Millennium Digital Transmitterrel jelentkezett, mely Linuxszal vezérelhető. Ezek mellett még felsorolni is nehéz, hány egyéb Linux-alkalmazást láthattunk a kiállításon.

A jövő hónapban visszatérünk a hagyományos kerékvágásba: Debian Linuxunkat frissítjük majd a 2.4-es rendszermagra és XFree86 4.x-re, emellett pedig telepítünk egy ATI All-In-Wonder Radeon grafikus kártyát is.

Robin Rowe
a MovieEditor.com internetes és televíziós videoalkalmazásokat készítő cég egyik partnere. Írásai a Dr. Dobb's Journalban, a C++ Reportban, a C/C++ Users Journalban, a Data Based Advisorban jelentek meg és számos tanácskozás anyagában megtalálhatók. A Robin által készített programok sorában található többek az a közt kiszolgálóalapú videoszerkesztő rendszer, amit a Manhattan 24 órás televíziós hírcsatorna, a Time Warner New York One, illetve a kapcsolódó honlap ➲ http://www.ny1.com/ is használ. Elérhető a robin.rowe@ movieeditor.com címen.

Ellenőrzés pásztázókkal: Nessus (2. rész)

Nessus: a magasabb szintre emelt biztonsági elemzés.



A múlt hónapban megkezdtük érdekes és veszélyes utunkat a pásztázás világába. Akkor az Nmap nevű sokoldalú kapupásztázó program állt írásunk középpontjában. Az Nmap segítségével a rendszergazdák, a biztonsági szakemberek (és bizony-bizony a leendő betörők is) megállapíthatják, hogy egy adott gépen milyen szolgáltatásokhoz lehet kapcsolódni.

A gép biztonsági elemzéséhez jó kiindulópontot nyújt, ha megvizsgáljuk, hogy milyen belépési pontokat kínál fel a rendszer. Vajon miként értelmezzük az Nmap által átadott ismereteket? A múlt hónapban például az egyik kipróbált pásztázás a *listán* látható kódot adta. Mit jelent ez? Rendben, azt már tudjuk, hogy a gép futtat webkiszolgálót (TCP 80), bizonyos RPC-szolgáltatásokat (UDP 111, UDP 1026), és valószínűleg Windows-megosztásokat is (UDP 137, TCP 138-139). Ezek közül a szolgáltatások közül melyik támadható? Ezen a ponton lépnek be a biztonsági pásztázók. Vállalva azt is, hogy a témában előreszaladunk, nézzük meg célpontunk Nessus-pásztázásának kimenetét (lásd *1. kép*).

Helyhiány miatt nem mutatjuk be az egész jelentést, de még ebből a rövidített változatból is jól látható, hogy a Nessus hét biztonsági lyukat, illetve kihasználható támadási pontot észlelt a célrendszerben. Ezenkívül négy további figyelmeztetést és két biztonsággal kapcsolatos megjegyzést is megjelenített.

A Nessus egyebek között megállapította, hogy ez a gép a Sambar webkiszolgálót futtatta, amelyen nem volt beállítva a rendszergazda jelszava, és a veszélyes mailit.pl CGI programot is el lehetett érni (az 1. kép nem mutat minden részletet, a hiányzó részeket el kell hinnünk). Az egész C:\ meghajtó a jelszó beállítása nélkül volt megosztva. Továbbá a Nessus még azt is felfedte, hogy a rendszer sebezhető, ugyanis jelszavas védelem esetén is védtelen maradt volna a *Null session* kapcsolattal és a *first-letter* jelszótámadásokkal szemben. Futott még egy FTP-kiszolgáló a TCP 1432-es kapun (ezt azonban az Nmap tévesen blueberry-lm szolgáltatásként értelmezte), és a TCP/IP-verem kiszámítható TCP-sorozatszámokat használt. Ezt sokféle módon ki lehet használni: például TCP-eltérítésre és IP-hamisításra.

Ez a rendszer megérett a feltörésre! Mi lehet ez a Nessus nevű halálos varázslat? És miért táncolja körbe az Nmapot a rendszer elemzése közben?

Biztonsági pásztázók

Amíg az Nmaphoz hasonló kapupásztázók feltárják, hogy mi figyel, a Nessus-szerű biztonsági pásztázók elárulják, hogy mi sebezhető. Mivel előbb érdemes megtudni, hogy mi figyel, és csak ezt követően lehet a gyenge pontokat kipuhatolni, a biztonsági pásztázók általában tartalmaznak kapupásztázót vagy kapcsolódnak egyhez.

A Nessus minden egyes pásztázás első lépéseként az Nmapot hívja meg, ezért volt félrevezető az előbb azt sugallni, hogy a Nessus sokkal jobb elemzést adott, mint az Nmap – valójában a Nessus az Nmaptól függ.

Miután a biztonsági pásztázó megállapította, hogy milyen szolgáltatások működnek, különféle ellenőrzéseket végez: meghatározza, hogy melyik programcsomag melyik változata fut, és ez sebezhető-e ismert módszerrel. Természetesen a nyomozásnak ebben a szakaszában szükséges egy jó adatbázis, amely a biztonsági lyukak leírását tartalmazza, és ezt az újabb hibák felbukkanásával egy időben frissítik. Eszményi esetben az adatbázis kézzel is szerkeszthető, azaz a felhasználó saját sebezhetőségi próbákat építhet be – az adott igényeknek és környezetnek megfelelően. Ezzel a szolgáltatással megvalósulhat az az igény, hogy a felhasználó maga is frissíthesse adatbázisát, ha a pásztázó fejlesztője ráérősen adja ki a frissítést. A Nessus magas szinten testre szabható, ellentétben számos más biztonsági pásztázóval.

A biztonsági másoló a megadott gépeken megtalálja, azonosítja és elemzi a figyelő szolgáltatásokat, ezután jelentést készít. A jobb pásztázók nemcsak a sebezhetőség tényét rögzítik, hanem részletesen elmagyarázzák az esetet és a javításra is javaslatot tesznek.

A jó biztonsági pásztázó olyan terjedelmes jelentést hoz létre, hogy esetenként a súlyos pénzeket kereső tanácsadók is ezt nyújtják be teljes körű biztonsági ellenőrzésük fő végeredményeként. Ez a gyakorlat megkérdőjelezhető ugyan, de hangsúlyozza a tényt, hogy a jó pásztázás sok adatot szolgáltat.

A Nessuson kívül akad még néhány ingyenes biztonsági pásztázó, például a VLAD és a SAINT. A Nessus azonban kiemelkedik a sorból, mert kiváltható vele néhány nagy tudású kereskedelmi termék: az ISS Internet Scanner és a NAI CyberCop Scanner.

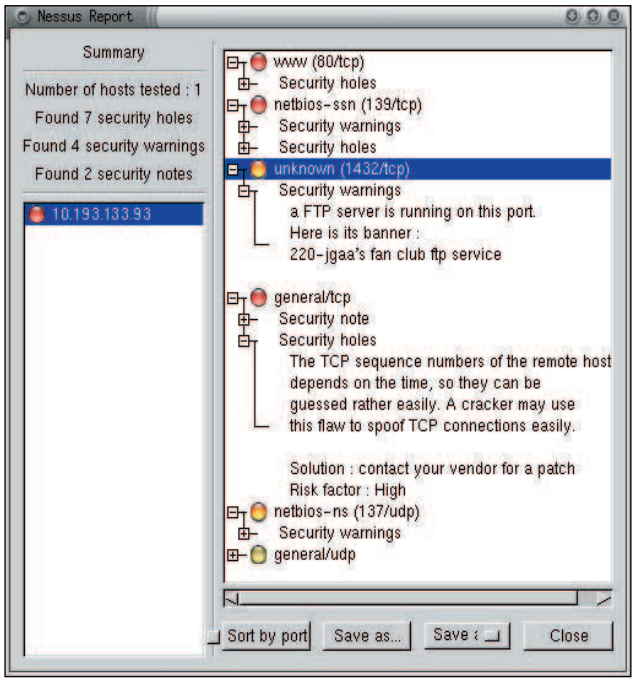
Az elsődlegesen *Renaud Deraison* és *Jordan Hrycaj* által fejlesztett Nessus a Gimp és az Apache programokkal áll egy szinten, olyan értelemben, hogy ezek az eszközök ugyanannyira vagy még jobban használhatók és rugalmasabbak, mint kereskedelmi megfelelőjük.

Mielőtt továbblépnénk, megismételjük előző havi figyelmeztetésünket: a tudás hatalom – mindent felelősségünk teljes tudatában használjunk! Az Nmap, a Nessus és a hozzájuk hasonló eszközök csak olyan rendszeren és hálózaton futtathatók, amelyek pásztázását engedélyezték számunkra. A kapupásztázás általában nem törvénytelen tevékenység, az engedély nélküli biztonsági pásztázás azonban komoly bajba sodorhat minket.

A Nessus felépítése

A Nessus két fő részből áll: a kiszolgáló végzi az összes pásztázást, az ügyféllel pedig vezérelhető a pásztázás és megnézhető a jelentés. Ez az osztott felépítés rugalmassá teszi a Nessust, így nem szükséges munkaállomásunk teljes processzoridejét a pásztázásra fordítani. Szintén lehetőség nyílik a felületek keverésére, például használhatjuk a kiszolgáló unixos változatát, és ehhez választhatjuk az X-, a Java- vagy az MS-Windows-ügyfelet. (A hirek szerint a Java-ügyfelet nem fejlesztik már tovább.)

A `nessusd` a TCP 3001-es és a TCP 1241-es kapukon várja a kapcsolódó ügyfeleket (az 1241-es kaput az Internet Assigned Numbers Authority nemrég rendelte a Nessushoz, ugyanis a 3001-es



1. kép Egy Windows 98 Nessus-pásztázása

előbb-utóbb „kimegy a divatból”.) Az ügyfélkapcsolatokat El Gamal-alapú nyilvános kulcsú eljárás alapján hitelesítik, és a titkosítást olyan titkosító kulccsal végzik, amelyet minden kapcsolatnál különkülön újra kicserél a két fél. E tekintetben a Nessus titkosító rétege (Jordan Hrycaj valósította meg a libpeks könyvtára segítségével) hasonlóan viselkedik az SSL-hez.

A Nessus ügyfélrésze, a `nessus` beállítható úgy is, hogy titkos kulcsunk jelszóval védett legyen, de akár azt is megadhatjuk, hogy ne kapcsolódjon hozzá jelszó. Előbbi esetben a nem engedélyezett felhasználók nem tudnak munkaállomásunkról csatlakozni a Nessus-kiszolgálóhoz.

Miután csatlakoztunk a kiszolgálóhoz, az felkínálja az általa támogatott bővítmények listáját (sebezhetőségi próbákat) és néhány egyéb beállítási lehetőséget. Ha belefordítottuk a Nessusba, az a lehetőség is megjelenik, hogy a pásztázás azután is folytatódjon, miután a kapcsolat megszakadt az ügyféllel (ez az úgynevezett leválasztott pásztázás – Detached Scan). Egy teljes oldalnyi lehetőségünk nyílik tudásbázis létrehozására és fenntartására, amely szintén belefordítható a programba, így ennek segítségével a pásztázások eredményei elrakhatók, valamint a gépek biztonsági állapota pásztázásról pásztázásra nyomon követhető (például lehet különbségi pásztázást futtatni).

Fontosnak tartjuk megjegyezni, hogy az utóbbi két lehetőség csak kísérleti állapotban van, ezért kisebb üzemelési gondok miatt nem fordítódnak bele a programba, csak ha kifejezetten akarjuk. A Nessus 1.2 megjelenéséig ezeket a hibákat ki fogják javítani, és ez a két módozat is teljesértékű válik a programnak. Azért említjük meg ezeket itt, mert a leválasztott pásztázás (Detached Scan) lehetősége különösen jó példát szolgáltat a Nessus kiszolgálóalapú felépítésének értékes mivoltára.

Miután mindent beállítottunk és elkezdtük a pásztázást, a Nessus meghívja a megadott és/vagy szükséges modult vagy bővítményt, egy Nmap pásztázással bevezetve. Az egyes bővítmények futásának eredményétől függ, hogy kell-e további próbákat futtatni – a Nessus ebben meglehetősen okos. A pásztázás végeztével az eredményeket elküldi az ügyfélnek. Ha a munkafolyamat mentése (session-saving) be van állítva, az eredmények a kiszolgálón is tárolhatók.

A Nessus beszerzése és telepítése

A legtöbb nyílt forráskódú csomaghoz hasonlóan a Nessus is elérhető forráskódban és bináris formában egyaránt. A RedHat 7.0-hoz készített Nessus-csomag, melynek változatszáma 1.0.7a (a legfrissebb a cikk írásának idején), csak a ➲ http://redhat.aldil.org/rpm.html?id=73 címről tölthető le, *Matthias Saou*-nak köszönhetően (ezekbe a bináris okba nem fordították bele a kísérleti forráskódrészleteket).

Ha nem RedHat 7.0-n dolgozunk és az általunk használt terjesztésben nincs Nessus-csomag (a Debian 2.2-ben van), vagy a kísérleti lehetőségek érdekelnek, akkor a forrásból kell a Nessust lefordítani. Aggodalomra semmi ok: ha előbb telepítünk néhány szükséges dolgot és elolvassuk a Nessus telepítési leírását, a fordítás simán fog menni. A Nessus FAQ (➲ http://www.nessus.org/doc/faq.html) és a Nessus levelezőlista (http://list.nessus.org) bőséges ismeretanyagot szolgáltat a Nessus fordításáról és telepítéséről.

A Nessus a következő csomagok meglétét követeli: Nmap, a múlt hónapban ismertetett kapupásztázó, gtk, a Gimp eszközkészlet – beleértve a gtk+, gtk+-devel, glib+devel és XFree86-devel csomagokat, és az m4 parancsfájl-értelmező környezet, vagy a libgmp (ennek csomagját egyszerűen gmp-nek hívják).

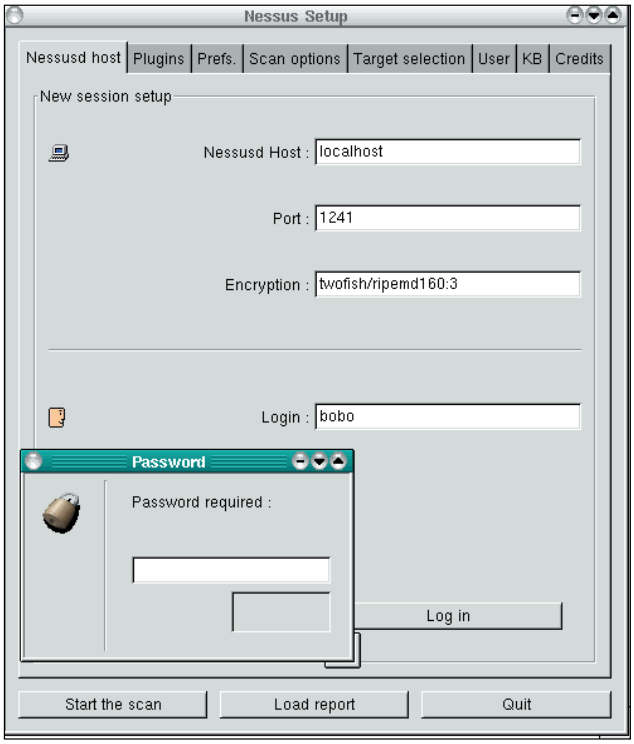
Miután telepítettük ezeket, terjesztésünknek további szükségletei is lehetnek. Két ilyen esetről tudunk. Először, a gmp-2.0 szükséges a RedHat 7.0-hoz (amelyben általában a gmp-3.0 található meg és nem a 2.0; az RPM `--force` kapcsolóját kell használni, amennyiben a 3.0 már telepítve van a 2.0 telepítésekor). Ez a csomag letölthető a ➲ http://www.redhat.com/swr/i686/gmp-2.0.2-5.i686.html címről. Másodszor, a Nessus telepítéséhez vagy lefordításához SuSE Linux alatt szükség van a bison, flex, gtkdev és glibdev csomagokra. További részletekért olvassuk el a ➲ http://www.nessus.org/doc/faq.html oldalt.

Ha már minden szükséges összetevő a helyére került, fordíthatjuk és telepíthetjük a Nessus-csomagokat. A fordítás egyszerű, mind a négy csomagnál a következő lépéseket kell végrehajtani: 1. kicsomagolás, 2. a csomag könyvtárában a `./configure` parancs kiadása, 3. `make`, majd 4. `make install`. A csomagokat a következő sorrendben fordítsuk le és telepítsük: nessus-libraries, libnasl, nessus-core és nessus-plugins.

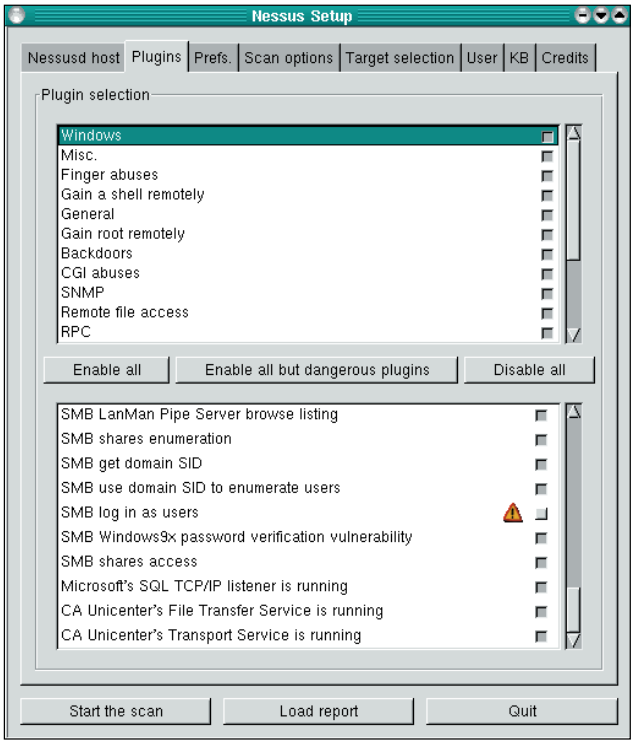
Mielőtt lefuttatnánk a `configure` parancsfájlt a nessus-core-hoz, fontoljuk meg, hogy akarjuk-e használni munkafolyamat mentése illetve a tudásbázis lehetőségeket. A munkafolyamat mentése

Nmap pásztázás TCP-kapcsolat, UDP- és RPC-modulokkal

```
Starting nmap V. 2.53 by fyodor@insecure.org
(www.insecure.org/nmap/)
Interesting ports on (10.193.133.93):
(The 3075 ports scanned but not shown below
are in
state: closed)
Port           State          Service (RPC)
80/tcp         open           http
111/udp        open           sunrpc (rpcbind
V2)
137/udp        open           netbios-ns
138/udp        open           netbios-dgm
139/tcp        open           netbios-ssn
1026/udp       open           (rpcbind V2)
1432/tcp       open           blueberry-lm

Nmap run completed-1 IP address (1 host up)
scanned
in 14 seconds
```



2. kép A „bobo” felhasználó első bejelentkezése a Nessus-kiszolgálóra



3. kép Bővítmények képernyője (A Windows-család látható)

lehetővé teszi a megszakított műveletek folytatását (például: a pásztázás folytatható az operációs rendszer vagy egy alkalmazás lefagyása után) és a leválasztott pásztázást (lásd fent). Ezt a lehetőséget a configure parancsfájl `--enable-save-session` kapcsolójával engedélyezhetjük.

A tudásbázis lehetővé teszi, hogy a pásztázások eredményeit a kiszolgálón tároljuk egy adatbázisban, amelynek felhasználásával később különbségi pásztázásra nyílik lehetőség. A tudásbázist engedélyező configure-kapcsoló a `--enable-save-kb`.

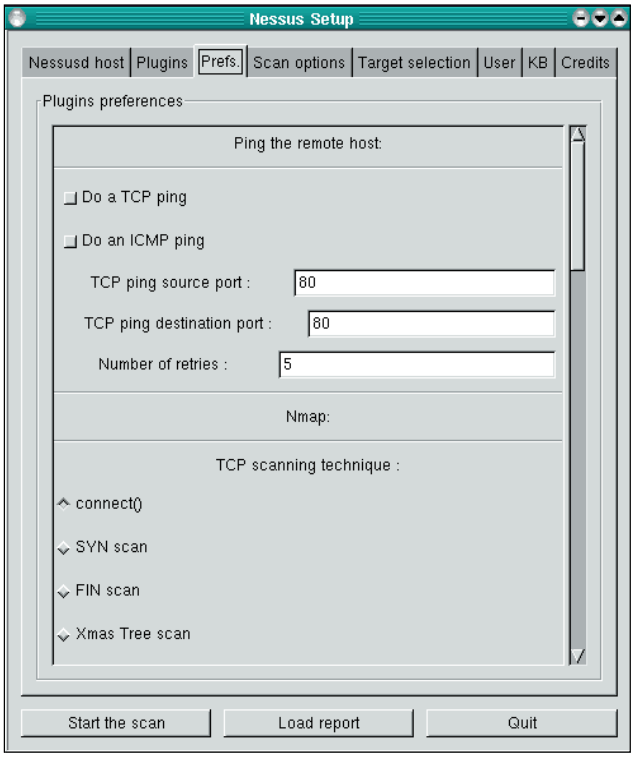
Ezek szerint ha a Nessus mindkét kísérleti lehetőségét ki akarjuk használni, a nessus-core fordítása előtt a következőképpen hívjuk meg a `configure`-t:

```
./configure --enable-save-sessions --enable-save-kb
```

A ➲ http://www.nessus.org/documentation.html címen részletesen olvashatunk ezeknek a lehetőségeknek a lefordításáról és használatáról. Mivel kísérleti kódról van szó, ebben a cikkben már nem ejtünk több szót róluk.

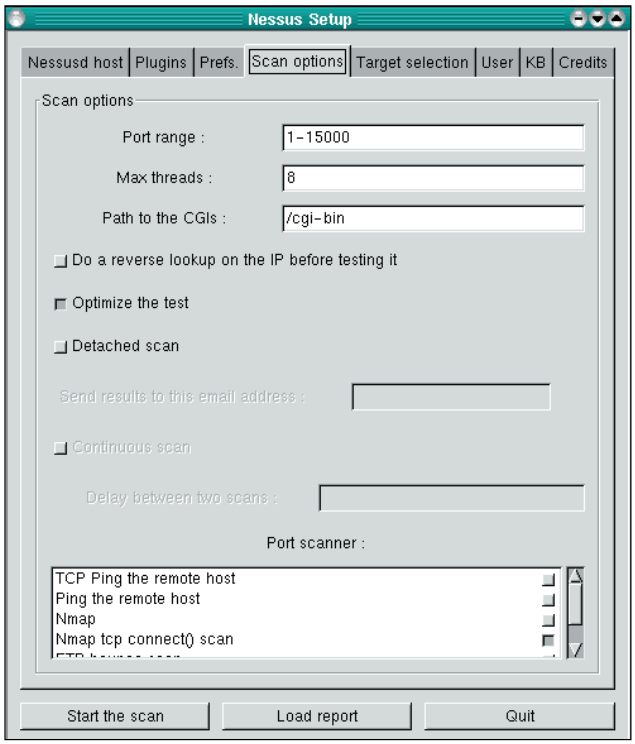
Miután mind a négy csomagot lefordítottuk és telepítettük, győződjünk meg róla, hogy a /etc/ld.so.conf fájl tartalmazza a /usr/local/lib bejegyzést. (Ha nem így lenne, adjuk hozzá kedvenc szövegszerkesztőnkkel.) Ezután futtassuk az `ldconfig` parancsot, hogy az ld (dinamikus csatoló) gyorstára frissüljön.

Végül, mivel a Nessus egyik erőssége, hogy fejlesztői rendszeresen új sebezhetőségi parancsfájlokat adnak ki, jó ötlet a munkát egy teljes sebezhetőségi adatbázissal kezdeni. A `nessus-update-plugins` parancsfájl segítségével az összes olyan bővítmény letöltődik a lynx program segítségével, amelyek a Nessus kiadása óta jelentek meg. Javasoljuk a `nessus-update-plugins -v` használatát, mert a `-v` kapcsoló nélkül a parancsfájl nem írja ki, hogy milyen bővítményeket telepít. Az új parancsfájlok letöltése, kicsomagolása és mentése után a `nessus-update-plugins` újraindítja a `nessusd`-t, hogy a rendszer az új bővítményeket használni tudja



4. kép Beállítások képernyő

(feltéve, hogy a nessus démon fut). Jelenleg ez a parancsfájl nem készít MD5 vagy másféle ellenőrzőösszeget, ezért a folyamat sokféleképpen befolyásolható. Ha ez zavar minket, a bővítményeket kézzel egyenként is letölthetjük a ➲ http://www.nessus.org/scripts.html weboldalról, de még ekkor sem lehetünk teljesen biztosak abban,



5. kép A Pásztázás beállításai képernyő

hogy minden rendben van – hacsak nem olvassuk el az összes parancsfájlt (a /usr/local/lib/nessus/plugins könyvtárban vannak), mielőtt a pásztázást megkezdenénk.

Nessus-ügyfelek

Hacsak nem egy gépen futtatjuk a Nessus-kiszolgálót és az ügyfelet, további telepítéseket kell elvégeznünk azon a gépen, amelyet ügyfélgépként kívánunk használni. A Nessus-kiszolgálóval (amelyen a nessusd fut) ellentétben, amely csak Unix-alapú gépen működhet, az ügyfelek Unixon és MS Windowson egyaránt futhatnak. A Nessus lefordítása és telepítése Unix-ügyfélgépen nem különbözik a kiszolgálótól (lásd fent).
A windowsos ügyfelek telepítése (WinNessus, NessusW és NessusWX) még ennél is egyszerűbb, mindhárom megtalálható bináris formában. A WinNessus hasonlít legjobban a Unix-ügyfél felhasználói felületére, aki ehhez van hozzászokva, annak ez a legjobb választás. Mindhárom Windows-ügyfél letölthető a ➲ http://www.nessus.org/win32.html címről.
Mielőtt a Nessus-ügyfelek használatáról beszélnénk, indítsuk el a démont.

A nessusd futtatása és karbantartása

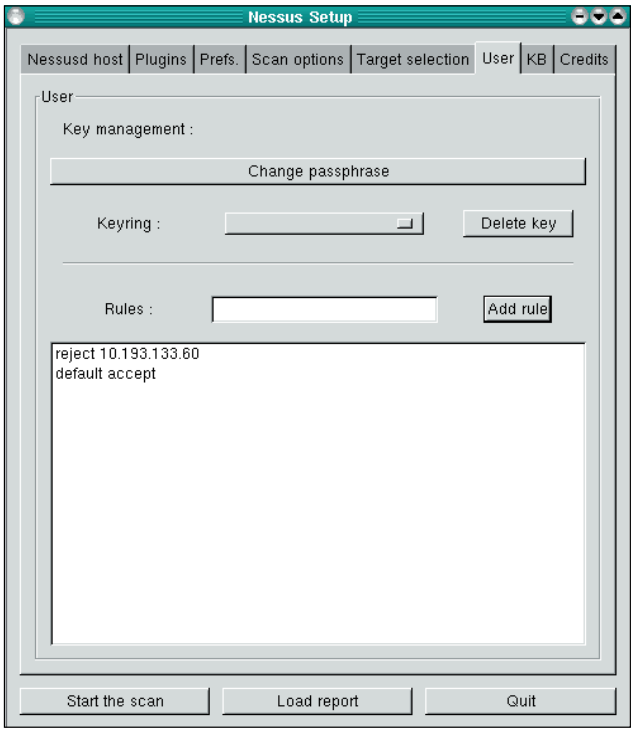
Rendben, térjünk vissza a Nessus-kiszolgáló konzoljához és készüljünk fel az első indításra. (Remélem, mindenki izgatottan várja ezt a percet és csakis jóban sántikál!) A nessusd sok más démontól különbözik abban, hogy elindítható démonként (azaz a háttérben) fut, valamint kapcsolóiban is, amelyek befolyásolják a viselkedését.
A démonmódban való indításhoz adjuk ki a `nessusd -D` parancsot. Ahogy az egy kiszolgálóalapú alkalmazástól elvárható, szükségünk lesz néhány Nessus-felhasználói fiókra a kiszolgálón. Ezek függetlenek a kiszolgáló helyi Unix-felhasználói fiókjaitól. A Nessus-fiókok két különböző módon hozhatók létre. Az első módszer a nessusd meghívása a `-P` kapcsolóval, amelyet közvetlenül a felhasználónév és az egyszer használatos jelszó követ. Ez nem zavarja a nessus démon működését, és nem is indít egy újat; valójában a Nessus felhasználói adatbázisát frissíti, és észrevétlenül újraindítja a démont. Például a „bobo” felhasználót „scuz00DL” jelszóval így adjuk hozzá:

```
nessusd -P bobo,scuz00DL
```

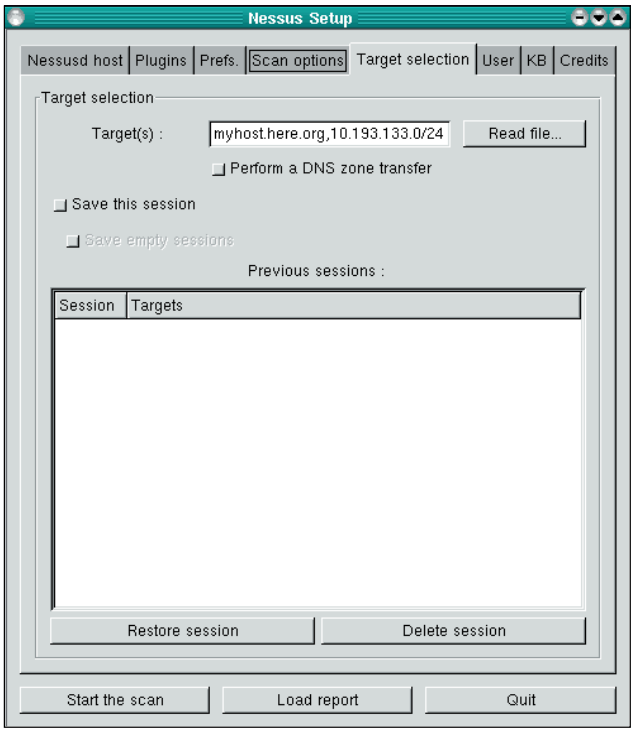
A jelszót egyszer használatosnak neveztük, mert alapesetben miután bobo először bejelentkezett és megadta a jelszavát, nyilvános kulcsa feljegyzésre kerül a Nessus-kiszolgálón, így a további bejelentkezések alkalmával nem kell újra beírnia a jelszót (a hitelesítés észrevétlenül történik egy SSL-szerű felszólítás illetve válasz művelettel).
A második és sokkal hatékonyabb módszer, hogy új felhasználói fiókokat hozzunk létre a kiszolgálón: a `nessus-adduser` parancs. Ez a parancsfájl gyakorlatilag minden varázslatát a nessusd hívogatásával végzi el, de kényelmes felületet ad a felhasználók kezelésére, ha a `nessusd -D` parancsnál nagyobb részletességgel szeretnénk őket szabályozni. Nemcsak a felhasználónevet és a jelszót kéri el, hanem azt az IP-címet is, ahonnan a felhasználó kapcsolatot kezdeményezhet, illetve azokat a szabályokat is, amelyek meghatározzák, hogy a felhasználó mely gépeket pásztázhatja a Nessusszal.
Ha ilyen mélységben érdeklődünk a felhasználói fiókok kezelése iránt, érdemes végigolvasnunk a nessus-adduser leírását. A cikk hátralevő részét most inkább arra szánjuk, hogy megbeszéljük a Nessus-pásztázás felépítését, futtatását és értelmezését.
Mielőtt azonban elhagynánk a hitelesítés témakörét, meg kell említeni egy másik hitelesítési fajtát is, amelyet a Nessus használ: az ügyfél helyben is végez hitelesítést minden egyes ügyfélkapcsolathoz. A Nessus-ügyfél első elindításakor egy jelszót kérdez. Ez a jelszó védi a titkos kulcsot a Unix-fiók `home` könyvtárában, amelyre a Nessus indításakor bejelentkezünk. Ezt mindig meg fogja kérdezni. Ezután ha a Nessus-kiszolgálóhoz csatlakozunk, a titkos kulcs segítségével jön létre az előbb említett észrevétlen felszólítás/válasz tranzakció, amely tulajdonképpen a távoli nessusd folyamat számára hitelesít minket. Ha ez most homályosnak tűnik is, semmi baj, a lényeg, hogy ne feledjük: ennek a jelszónak, amelyet mindig elkér az ügyfélprogram, semmi köze ahhoz a jelszóhoz, amelyet a Nessus-kiszolgálóhoz való első csatlakozásunk alkalmával használtunk.

Biztonsági pásztázások a Nessusszal

Most kezdődik az igazi móka! Miután a Nessust telepítettük és legalább egy felhasználót is beállítottunk, kezdhetünk pásztázni.
Először indítsuk el az ügyfelet és írjuk be az ügyfél titkos kulcsának jelszavát (mellesleg ezt bármikor megváltoztathatjuk a `nessus -C` paranccsal, amely bekéri a jelenlegi jelszót és az újat, amire módosítani szeretnénk).
Ezután írjuk be annak a nessusd kiszolgálót futtató gép nevét vagy IP-címét, amelyhez csatlakozni szeretnénk; a kaput, amelyen a kiszolgáló figyel; a használni kívánt titkosítási módszert és a Nessus felhasználónév-jelszó párost (*2. kép*). A kapu és titkosítás (*Port és Encryption*) alapértelmezett értékei általában megfelelők.
Ha elkészültünk, kattintsunk a *Log in* (Bejelentkezés) gombra.
Ha ez az első alkalom, hogy a kiszolgálóhoz kapcsolódunk, akkor meg kell még adnunk az egyszer használatos jelszót is (legközelebb már nem kéri a program). Ezután létrejön a kapcsolat és elkezdhetjük a pásztázást.
Ha a *Plugins* (Bővítmények) fülre kattintunk, megjelenik az összes a Nessus-kiszolgálón elérhető sebezhetőségi teszt listája, családok szerint csoportosítva (*3. kép*). Kattintsunk a család nevére (ezek az ablak felső felében jelennek meg), hogy megnézhessük, milyen bővítmények érhetők el az adott családhoz. A család melletti jelölőnégyzet használatával választhatjuk ki az összes bővítményt.
Ha nem tudjuk, hogy egy adott bővítmény mit csinál, kattintsunk



7. kép Felhasználó képernyő



6. kép Célpont kiválasztása képernyő

a nevére, ilyenkor egy tájékoztató ablak bukkan fel. Ha az egérmutatót a bővítmény neve fölé húzzuk, egy buborék jelenik meg a bővítmény rövid ismertetésével. Az olyan bővítmények, amelyek jelölőnégyzete mellett sárga háromszög található, különösen veszélyesek: az általuk végrehajtott tesztek megszakíthatják vagy akár le is fagyaszthatják a célponton (áldozaton) futó szolgáltatásokat. Csakis nagy körültekintéssel használjuk ezeket!

Gond nélkül kiválaszthatunk nagyszámú bővítményt, akár az összeset is. A Nessus elég okos ahhoz, hogy például a Windows-próbákat átugorja a nem Windowst futtató gépeken. Általában a Nessus hatékonyan eldönti, hogy milyen próbát kell futtatnia az adott körülmények között.

A következő képernyő a *Prefs* (Beállítások – *4. kép*). Ellentétben azzal, amit ennek láttán elsőre gondolnánk, ez a képernyő mégsem az általános, hanem a bővítményekkel kapcsolatos beállításokat tartalmazza, némelyik kötelező az adott bővítmény helyes működéséhez. Menjünk végig a listán, és adjunk meg annyi adatot, amennyit csak tudunk.

Figyeljünk a *Ping* részre (a legtetején), elég sokszor előfordul, hogy ha bármelyik pingmódszert (TCP vagy ICMP) választjuk, a Nessus arra a hibás következtetésre jut, hogy a célpont nincs bekapcsolva. A Nessus nem hajt végre semmilyen próbát olyan gépen, amely nem válaszol a pingre, ezért ha nem vagyunk biztosak benne, ne pingeljünk! Figyelem: az Nmap részben a Linux-felhasználók csak a `tcp connect()` lehetőséget válasszák, az összes többit a libpcap hibája miatt ne, ugyanis az befolyásolja a Nessus kapupásztázási tevékenységét.

A Prefs után jön a Scan Options (Pásztázás beállításai – *5. kép*). Vegyük észre, hogy az 5. képen szereplő Nessus-program a munkafolyamat mentése lehetőséggel lett fordítva, ennek bizonyítéka a *Detached Scan* és a *Continuous Scan*, amelyek egyébként nem jelentek volna meg. Akárcsak a Prefs képernyőn, itt is csak az Nmap `tcp connect()` lehetőséget válasszuk ki a *Port scanner* beállításnál, a fent említett hiba miatt.

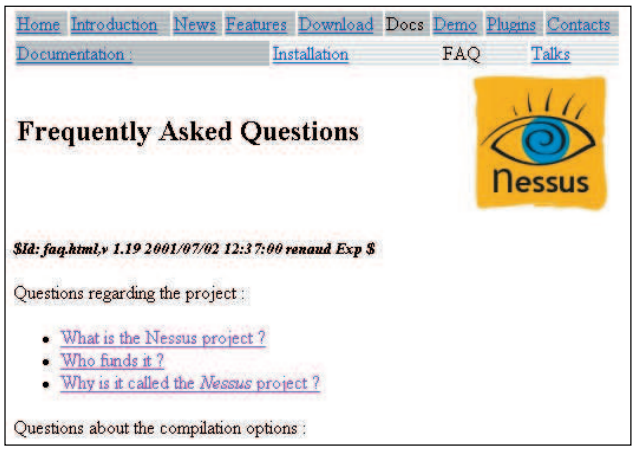
Az *Optimize the test* lehetőség választásával elkerülhetjük a látszólag felesleges próbákat, de ez hamis negatív eredmény keletkezéséhez vezethet, legalábbis elméletileg. Gondoljuk meg, mennyire aggaszt ez minket ahhoz viszonyítva, amilyen gyorsan megkapjuk az eredményt. Ha már a sebességről beszéltünk: amennyiben gyorsan szeretnénk eredményt kapni, kerüljük a *Do a reverse (DNS) lookup…* használatát, ez ugyanis minden pásztázott IP-címhez megpróbálja megtalálni a gépnevet.

Itt az ideje megnevezni „áldozatainkat”, vagyis a célpontokat. Ezeket a *Target(s):* mezőben adhatjuk meg a *Target Selection* (célpont kiválasztása) képernyőn (*6. kép*). Ide írhatjuk be vesszővel elválasztva a gépneveket, IP-címeket és hálózati címeket x.x.x.x/y alakban (ahol az x.x.x.x a hálózat száma és y az alhálózati maszk bitjeinek a száma, pl. 192.168.1.0/24).

A *Perform a DNS zone transfer* bejelölésével a Nessust arra utasítjuk, hogy minden elérhető DNS-információt megpróbáljon megszerezni a Target(s): mezőben megadott valamennyi tartománynévről és altartománynévről. Megjegyzendő, hogy a legtöbb internetes DNS-kiszolgáló úgy van beállítva, hogy megtagadja az ismeretlen gépektől érkező zónaátviteli kéréseket. A többi beállítás ezen a képernyőn a fentebb említett kísérleti munkafolyamat mentésének lehetőségével van összefüggésben. További tájékoztatásért olvassuk el a ➲ http://www.nessus.org/documentation.html weboldalt a kísérleti lehetőségek használatáról.

Végezetül még egy képernyőn kell túljutnunk a pásztázás megkezdése előtt, ez a *User* (Felhasználó) képernyő (*7. kép*). (Átugrottuk a KB képernyőt, amely csak akkor van jelen, ha a tudásbázis-lehetőséget is használni akarjuk, és ezért azt is lefordítottuk.) Ezen a képernyőn adhatjuk meg az ügyfél jelszavát (ugyanaz a hatása, mint a `nessus -C` parancsnak), és a Target Selection képernyőn beírt célpontok közül jelölhetünk meg kivételeket (valójában a célpontlista finomhangolása ez).

Ezeket a kivételeket szabályoknak hívják és egyszerű alakban felírhatók: cím elfogadása, cím elvetése, alapértelmezett elfogadás vagy visszautasítás. A 7. képen a felsorolt szabályok azt jelentik, hogy ne pásztázza a 10.192.133.60-at, de pásztázzon minden mást



➲ http://www.nessus.org/doc/faq.html

az előző képernyőn megadottak közül.
Lássuk az eredményt! Kattintsunk a *Start the Scan* (Pásztázás megkezdése) gombra a képernyő alján. A pásztázás időtartama eltérő hosszúságú lehet, főként a kiválasztott gépek és a végrehajtandó próbák számától függ. A végeredmény hasonló az 1. képen láthatóhoz. A *Report* (Jelentés) ablakból menthetjük is a jelentést egy fájlba azon kívül, hogy nézegethetjük és különböző részleteit megjeleníthetjük. A támogatott formátumok között szerepel a HTML, az ASCII, a LaTeX és természetesen a Nessus saját jelentésfájl-formátuma, az NSR. Ha valaha is jelentést szeretnénk visszatölteni a Nessusba, az NSR formátumot kell használnunk.

Olvassuk el alaposan a jelentést, minden + jelre kattintsunk rá, és javítsuk a Nessus által jelzett hibákat. A Nessus megtalálja a hibát, sokszor megoldást is javasol, de nem javítja ki helyettünk. Az is igaz, hogy a Nessus nem feltétlenül talál meg minden biztonsági rést a rendszerünkön.
Ez a helyzet a biztonsági pásztázókkal: ennyit képesek megtenni, ráadásul nem minden bővítmény hatékony egyforma mértékben a hibák megtalálásában. Még ha hatékonyak is lennének, a Nessus érthető okokból nem találhatja meg azokat a biztonsági réseket, amelyekre nincs bővítménye, ezért bővítményeinket rendszeresen frissítsük.

Néhány záró gondolat

A Nessus sokoldalú, rugalmas, kereskedelmi szintű, de teljesen ingyenes és szabad biztonsági pásztázó. A helyesen létrehozott és értelmezett Nessus-jelentések segíthetnek minket abban, hogy a jól ismert hibákat elkerüljük. Nem beszéltünk a saját bővítmények írásáról, amelyek lehetővé teszik, hogy ne csak a közismert biztonsági réseket ellenőrizhessük, hanem új, ez idáig ismeretlen hibákat fedezzünk fel. Még egyszer kérjük, hogy mindenki felelősen használja ezt az eszközt!
Remélve, hogy így lesz, jó szórakozást kívánunk!

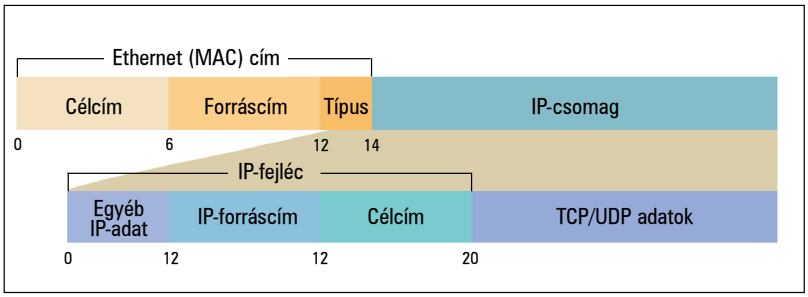
Mick Bauer (mick@visi.com) hálózati biztonsággal foglalkozó szaktanácsadó. 1995 óta a Linux elkötelezett híve, 1997 óta pedig OpenBSD prófétaként tevékenykedik. Mick szívesen fogad minden kérdést, és megjegyzést.

Csomagszűrő: bájtok leszippantása a hálózatról

A 2.2-es rendszermagban megjelent LSF segítségével a rendszermag beprogramozható, hogy melyik csomagot engedje be. Most megtudhatjuk, hogyan.

Abban az esetben, ha hálózati rendszergazda vagy, esetleg biztonsági kérdésekkel foglalkozol, illetve ha egyszerűen csak érdekel, hogy a helyi hálózatodon mi megy keresztül, néhány csomag leszippantása a hálózati kártyáról hasznos gyakorlat lehet a számodra is. Egy kis C programozással és alapvető hálózatos ismeretekkel felvértezve könnyen elfoghatod azokat az adatokat is, amelyek nem a te gépedre érkeznek. Ebben a cikkben ethernethálózatokról lesz szó, mert messze ez a legelterjedtebb a helyi hálózatok között. Később megvilágított okból azt is feltesszük, hogy a forrás- és a célgép ugyanahhoz a helyi hálózathoz tartozik.

Először is elevenítsük fel, hogyan működik egy átlagos ethernethálózati kártya. Akik már jártasak ezen a területen, azok nyugodtan ugorjanak a következő bekezdéshez. A felhasználói programokból származó IP-csomagok ethernetkeretekbe ágyazódnak be (így hívják az ethernetszakaszon átküldött csomagokat). Ezek egyszerűen nagyobb méretű, alacsonyabb szintű csomagok, amelyek tartalmazzák az eredeti IP-csomagot, valamint további adatokat is a célba juttatáshoz (lásd *1. ábra*). Ebben az esetben a cél IP-címe a cél 6 bájtos ethernetcímére képeződik le (ezt gyakran MAC-címnek hívják) az ARP-nak hívott átalakítás segítségével. Így a csomagot tartalmazó keret áthalad a forrást és a célt összekötő vezetéken. A keret valószínűleg hálózati eszközökön, például elosztókon vagy kapcsolókon is keresztülmegy, de mivel feltettük, hogy nem lépi át a helyi hálózatot, útválasztó vagy átjáró nincs benne.

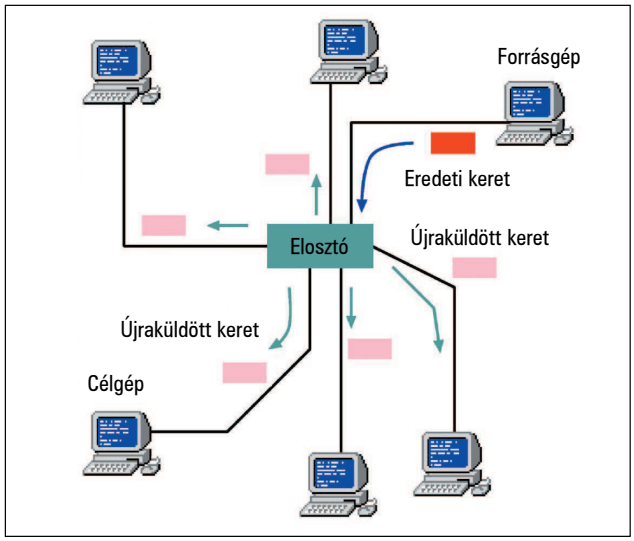


1. ábra IP-csomagok mint ethernetkeretek

Az ethernet szintjén nincs útválasztás. Más szavakkal: a forrásgép által küldött keret nem fog közvetlenül a célgépre jutni, ehelyett az a helyi hálózatot alkotó összes vezetéken megjelenik, és minden hálózati kártya látni fogja, amint elhalad (lásd *2. ábra*). Minden hálózati kártya elolvassa a keret első hat bájtját (amely az imént említett MAC-címet tartalmazza), de csak az a kártya olvassa be az egész keretet, amelyik a célcímben felismeri a saját címét. Ezen a ponton a keretet átveszi a hálózati meghajtó, visszaállítja az eredeti IP-csomagot, és továbbítja a fogadó alkalmazásnak a hálózati protokollvermen keresztül.

Pontosabban a hálózati meghajtó ellenőrzi a protokolltípus-mezőt az ethernetkeret belsejében (lásd 1. ábra), és ettől az értéktől teszi függővé, hogy melyik protokollfogadó függvénynek továbbítsa a csomagot. Az esetek többségében ez az IP protokoll lesz, a fogadó függvény kiveszi az IP-fejlécet, és a maradékot az UDP-t vagy TCP-t fogadó függvényeknek továbbítja. Ezek a protokollok azután továbbadják a csomagot a foglalatkezelő függvényeknek, amelyek végezetül a csomag adatait a felhasználói térben futó alkalmazásnak továbbítják. Az utazás során a csomag minden hálózattal kapcsolatos ismeretet elveszt, például a forrás címeit (IP és MAC), TCP-jellemzőket stb. Továbbá, ha a célgépen nincs nyitva a megfelelő kapu, a rendszermag eldobja a csomagot, amely így nem ér el az alkalmazások szintjére.

Ennek következtében a hálózaton utazó csomagok leszippantásához két különálló feladatot kell megoldanunk. Az első az ethernetcímzéssel kapcsolatos: nem tudjuk azokat a csomagokat elolvasni, amelyeket nem a mi gépünknek címeztek. A másik a protokollverem feldolgozásával függ össze: ha nem akarjuk elveszíteni a csomagot, minden egyes kaput nyitva kell tartani és figyelni kell. Ráadásul a csomag adatainak egy része a protokollverem feldolgozása során el is vész. Az első gond nem alapvető, ugyanis nem biztos, hogy érdekelnek minket a többi gépre küldött csomagok, lehet hogy csak a saját gépünkre küldötteket akarjuk leszippantani. A másodikat ellenben meg kell oldani. Megnézzük, hogyan célszerű kezelni ezt a két feladatot külön-külön, most kezdjük az utóbbival.



2. ábra Ethernetkeretek küldése helyi hálózaton keresztül

A PF_PACKET protokoll

Amikor egy foglalatot a szabványos `sock = socket (domain, type, protocol)` hívással megnyitsz, meg kell adnod, hogy melyik tartományt (vagy protokollcsaládot) fogod azzal a foglalattal használni. Gyakran használt család a PF_UNIX, amely a helyi gépen belüli párbeszédre használatos, és a PF_INET, amely az IPv4 protokollokon alapuló párbeszédekre jó. Meg kell továbbá adnod a foglalat típusát, melynek lehetséges értékei a megadott családtól függenek. A PF_INET család leggyakrabban használt típusai:

1. lista A leszippantott csomagok protokollveremszerű feldolgozása

```
#include <stdio.h>
#include <errno.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <linux/in.h>
#include <linux/if_ether.h>

int main(int argc, char **argv) {
  int sock, n;
  char buffer[2048];
  unsigned char *iphead, *ethhead;

  if ( (sock=socket(PF_PACKET, SOCK_RAW,
                    htons(ETH_P_IP)))<0) {
    perror("socket");
    exit(1);
  }

  while (1) {
    printf("----------\n");
    n = recvfrom(sock,buffer,2048,0,NULL,NULL);
    printf("%d bájt beolvasva\n",n);

    /* Ellenőrzi, hogy a csomag tartalmazza-e
    legalább
     * a teljes ethernet (14), IP (20) és
     * TCP/UDP (8) fejléceket.
     */
    if (n<42) {
      perror("recvfrom():");
      printf("Nem teljes csomag (errno is %d)\n",
             errno);
      close(sock);
      exit(0);
    }

    ethhead = buffer;
    printf("A forrás MAC-címe: "
           "%02x:%02x:%02x:%02x:%02x:%02x\n",
           ethhead[0],ethhead[1],ethhead[2],
           ethhead[3],ethhead[4],ethhead[5]);
    printf("A cél MAC-címe: "
           "%02x:%02x:%02x:%02x:%02x:%02x\n",
           ethhead[6],ethhead[7],ethhead[8],
           ethhead[9],ethhead[10],ethhead[11]);

    iphead = buffer+14; /* Az ethernetfejléc
                                    átugrása */
    if (*iphead==0x45) { /* Ellenőrzi az IPv4
                                    meglétét */
      printf("Forrásgép %d.%d.%d.%d\n",
             iphead[12],iphead[13],
             iphead[14],iphead[15]);
      printf("Célgép %d.%d.%d.%d\n",
             iphead[16],iphead[17],
             iphead[18],iphead[19]);
      printf("Forrás- és célkapuk %d,%d\n",
             (iphead[20]<<8)+iphead[21],
             (iphead[22]<<8)+iphead[23]);
      printf("Layer-4 protokoll
            %d\n",iphead[9]);
    }
  }

}
```

a SOCK_STREAM (ez a TCP-nek feleltethető meg) és a SOCK_DGRAM (az UDP megfelelője). A foglalat típusa befolyásolja a rendszermagot a csomagok kezelésében, mielőtt még azok az alkalmazáshoz kerülnének. Végül meg kell adnod a foglalaton átáramló csomagot kezelő protokollt (további részletek a socket(3) kézikönyvoldalon olvashatók).

A Linux-rendszermag újabb változataiban (a 2.0 utáni kiadásokban) új protokollcsaládot vezettek be, a PF_PACKET-et. Ez a család lehetővé teszi az alkalmazásoknak, hogy közvetlenül a hálózati kártya meghajtójával tárgyalva küldjenek és fogadjanak csomagokat, így elkerülik a szokásos protokollkezelést (például: TCP- és IP-, UDP-feldolgozást). Ez azt jelenti, hogy minden a foglalatra küldött csomag közvetlenül az ethernetcsatolóra kerül, és minden a csatolóra érkező csomag közvetlenül az alkalmazáshoz továbbítódik.

A PF_PACKET család két eléggé eltérő foglalattípust támogat, a SOCK_DGRAM és a SOCK_RAW típusokat. Előbbi a rendszermagra hagyja az ethernetszintű fejlécek hozzáadását és eltávolítását, utóbbi pedig teljesen az alkalmazásra. A protokollmező a socket() hívásban meg kell egyezzen a /usr/include/linux/if_ether.h-ban megadott ethernetazonosítók egyikével, amely az ethernetkeretben szállítható bejegyzett protokollokat adja meg. Hacsak nem nagyon sajátos protokollokról van szó, általában az ETH_P_IP-t kell használnod, amely magába foglalja az összes IP-protokollt (például: TCP, UDP, ICMP, nyers IP stb.).

Mivel ezeknek igen súlyos biztonsági következményeik lehetnek (például szerkeszthetsz egy hamis MAC-címmel rendelkező keretet), a PF_PACKET családot csak a rendszergazda használhatja.

A PF_PACKET család könnyedén megoldja a leszippantott csomagokkal kapcsolatos feladatainkat. Nézzük meg az *1. listán*, hogyan működik mindez. Megnyitunk egy PF_PACKET családhoz tartozó foglalatot, megadjuk a SOCK_RAW típust és az IP-vel kapcsolatos protokolltípust. Ezután elkezdünk olvasni a foglalatról, és néhány ellenőrzés után kinyomtatjuk az ethernetszintű adatokat, valamint az IP-fejlécet. A kinyomtatott címek és az *1. ábra* összevetéséből jól látható, milyen könnyű volt az alkalmazásból elérni a hálózati szintű adatokat.

Feltéve, hogy a géped ethernetes helyi hálózatba csatlakozik, kísérletezhetsz ezzel a kis példaprogrammal. Küldj csomagokat a gépedre egy másik gépről (pingelheted a gépedet vagy jelentkezz be Telnettel)! Látni fogod a neked szánt csomagokat, de nem fogod látni a többi gépre küldötteket.

Lehallgató és nem lehallgató mód

A PF_PACKET család lehetővé teszi, hogy az alkalmazások olyan formában szerezzék meg az adatcsomagokat, ahogy azok a hálózati kártyára érkeznek, de még ez sem engedi olyan csomagok elolvasását, amelyeket nem a gépnek címeztek. Ahogy az előbb láthattuk, ez amiatt történik, mert a hálózati kártya eldobja azokat a csomagokat, amelyek nem a saját MAC-címére érkeznek. Ezt a működési módot *nem lehallgató módnak* (nonpromiscuous) nevezik, ami egyszerűen azt jelenti, hogy az összes hálózati kártya csak a saját dolgával törődik és csakis a neki szóló kereteket olvassa el. Három

3. lista tcpdump -d eredménye

```
echer:~# tcpdump -d host 192.168.9.10
(000) ldh      [12]
(001) jeq      #0x800 jt 2   jf 6
(002) ld       [26]
(003) jeq      #0xc0a8090a   jt 12  jf 4
(004) ld       [30]
(005) jeq      #0xc0a8090a   jt 12  jf 13
(006) jeq      #0x806 jt 8   jf 7
(007) jeq      #0x8035jt 8   jf 13
(008) ld       [28]
(009) jeq      #0xc0a8090a   jt 12  jf 10
(010) ld       [38]
(011) jeq      #0xc0a8090a   jt 12  jf 13
(012) ret      #68
(013) ret      #0
```

4. lista tcpdump a -dd kapcsolóval

```
escher:~# tcpdump -dd host 192.168.9.01
{ 0x28, 0, 0, 0x0000000c },
{ 0x15, 0, 4, 0x00000800 },
{ 0x20, 0, 0, 0x0000001a },
{ 0x15, 8, 0, 0xc0a80901 },
{ 0x20, 0, 0, 0x0000001e },
{ 0x15, 6, 7, 0xc0a80901 },
{ 0x15, 1, 0, 0x00000806 },
{ 0x15, 0, 5, 0x00008035 },
{ 0x20, 0, 0, 0x0000001c },
{ 0x15, 2, 0, 0xc0a80901 },
{ 0x20, 0, 0, 0x00000026 },
{ 0x15, 0, 1, 0xc0a80901 },
{ 0x6, 0, 0, 0x00000044 },
{ 0x6, 0, 0, 0x00000000 },
```

kivétel van ez alól a szabály alól: ha a keret célcíme a különleges FF:FF:FF:FF:FF:FF MAC-cím – ezt minden kártya felveszi; ha a keret célcíme többszörös cím – ezt azok a kártyák veszik fel, amelyeken a többszörös cím fogadása engedélyezett; végül amelyik kártya lehallgató módban van – az összes csomagot felveszi, amit csak hall.

Természetesen céljaink elérése szempontjából az utóbbi a legérdekesebb. A hálózati kártyát egy nyitott foglalatára irányított megfelelő ioctl() hívással állíthatjuk lehallgató módba. Mivel ez a művelet biztonsági kockázatot hordoz magában, a hívást csak rendszergazdai jogosultságú felhasználó adhatja ki. Feltéve, hogy a „sock” egy már nyitott foglalat, a következő utasítás végzi el a feladatot:

```
strncpy(ethreq.ifr_name,"eth0",IFNAMSIZ);
ioctl(sock, SIOCGIFFLAGS, &ethreq);
ethreq.ifr_flags |= IFF_PROMISC;
ioctl(sock, SIOCSIFFLAGS, &ethreq);
```

(Az ethreq egy ifreq-szerkezet, amely a /usr/include/net/if.h-ban van megadva.) Az első ioctl kiolvassa az ethernetkártya pillanatnyi állapotjelzőit; ezt az értéket az IFF_PROMISC állandóval VAGY-oljuk, majd a második ioctl kiírja az új állapotjelzőt a kártyára. Nézzük meg ezt egy teljesebb példán (lásd a 2. listát a ➲ ftp://ftp.ssc.com/pub/lj/listings/issue86/ címen)! Lefordítás után rendszergazdaként futtatva egy helyi hálózathoz kapcsolt gépen az összes csomagot megláthatod, amely a vezetéken áramlik, akkor is, ha nem a te gépednek szánták őket. Ez azért lehetséges, mert hálózati kártyád lehallgató módban dolgozik. Könnyen meggyőződhetsz erről, ha kiadod az `ifconfig` parancsot és megfigyeled a kimenet harmadik sorát.

A Linux csomagszűrő

Mindkét leszippantással kapcsolatos gondunk mostanra megoldódott, van azonban még egy lényeges dolog, amelyet meg kell fontolnunk. Ha a valóságban is kipróbáltad a 2. listán bemutatott példát, mégha helyi hálózatodon nincs is túl nagy forgalom (néhány windowsos gép már elég ahhoz, hogy a sok NETBIOS csomagtól zsúfolt legyen a vezeték), észreveheted, hogy a szippantó túl sok adatot ír ki. Ahogy a hálózati forgalom nő, a szippantó kezdi elveszíteni a csomagokat, mert a számítógép nem tudja elég gyorsan feldolgozni őket.

A megoldás az lehetne, hogy szűröd az érkező csomagokat és csak azokról íratsz ki adatokat, amelyek érdekelnek. Az első ötlet ehhez az lehet, hogy *if* utasításokat szúrunk be a szippantó forrásába. Ez azonban noha szebbé teszi a szippantó kimenetét, a teljesítmény szempontjából nem lesz túl hatékony. A rendszermag ekkor is beolvassa a hálózaton áramló csomagokat, ami időpocsékolás, továbbá a szippantó ekkor is megvizsgálná az összes csomag fejlécét és döntene a csomaggal kapcsolatos adatok kinyomtatásáról.

Az eszményi megoldás az lenne, ha a szűrőt a lehető leghamarabb iktatnánk be a csomagfeldolgozó láncba (ez a hálózati meghajtó szintjén kezdődik és az alkalmazás szintjén ér véget, lásd a *3. ábrát*). A Linux-rendszermag lehetővé teszi, hogy berakjunk egy szűrőt (LPF a neve) közvetlenül a PF_PACKET protokollfeldolgozó műveletek belsejébe, amelyek röviddel azután futnak, hogy a hálókártya megszakítása kiszolgálásra került. A szűrő dönti el, hogy melyik csomag mehet tovább az alkalmazás felé, és melyik nem.

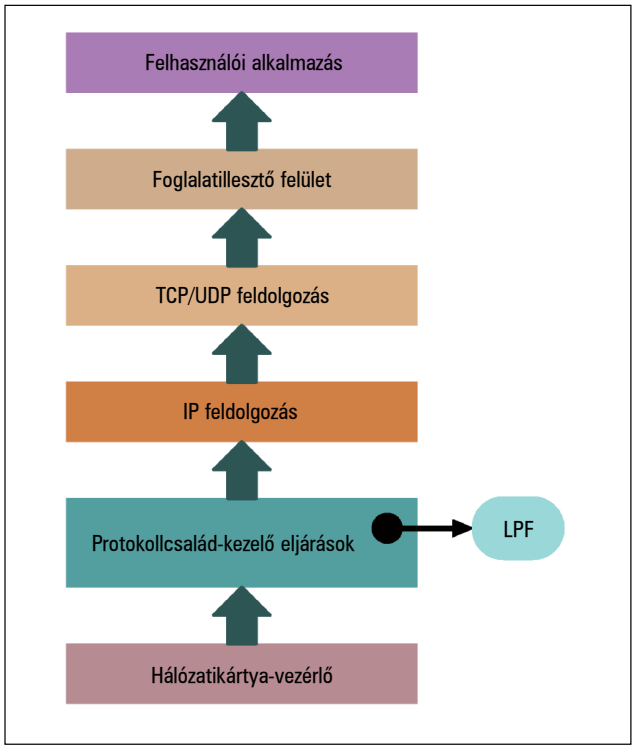
A lehető legnagyobb rugalmasság érdekében, és hogy a programozó keze ne legyen előre megadott feltételekkel megkötve, a csomagszűrőmotor számítógépszerűen megvalósított, amelyet a felhasználó programozhat. A programot egy különleges, gépi kódra emlékeztető nyelven kell megírni, amelyet BPF-nek hívnak (a Berkeley Packet Filter rövidítése), és amelyhez *Steve McCanne* és *Van Jacobson* egyik régi cikke adta az ötletet (lásd a *Kapcsolódó címeket*). A BPF tényleg Assembly nyelvnek látszik, egy pár regiszterrel és néhány utasítással, amelyek betöltik és tárolják az értékeket, számtani műveleteket és feltételes elágazást hajtanak végre.

A szűrőkód minden vizsgálandó csomagra lefut, és a BPF-feldolgozó a csomag adatait tartalmazó bájtokból álló tárterületen fejti ki működését. A szűrő eredménye egy egész szám, amely azt adja meg, hogy hány bájtot kell a csomagból a foglalatról az alkalmazás felé továbbítani. Ez további előnyökkel jár, mivel sokszor csak a csomag első néhány bájtja érdekes, így feldolgozási időt takaríthatunk meg, ha a többi másolását elkerüljük.

A szűrő programozása

Bár a BPF nyelv elég egyszerű és könnyű megtanulni, a legtöbben közülünk valószínűleg jobban szeretik az ember számára olvasható alakban megadott szűrőkifejezéseket. Ezért a BPF nyelv részleteinek és utasításkészletének bemutatása helyett (amelyet a fent említett cikkben megtalálhatsz) inkább azt fogjuk megtárgyalni, hogyan lehet egy működő szűrő kódját egy logikai kifejezésből előállítani.

Először is telepítened kell a tcpdump programot az LBL-től (lásd a *Kapcsolódó címeket*) – bár ha ezt a cikket olvasod, valószínűleg már ismered és használod a tcpdumpot. Az első változatokat ugyan-



3. ábra A csomagfeldolgozó lánc

azok írták, akik a BPF-cikket is, valamint a nyelv első megvalósítását. Valójában a tcpdump a BPF-et használja – a libpcap programkönyvtáron keresztül – a csomagok elkapására és szűrésére. A programkönyvtár operációs rendszertől független csomagoló a BPF-motor körül. Linuxon használva a BPF utasításait a Linux csomagszűrő hajtja végre.

A libpcap egyik leghasznosabb függvénye a pcap_compile(), amely logikai kifejezést tartalmazó karakterláncot alakít BPF szűrőkóddá. A tcpdump ezt a függvényt használja a felhasználó által a parancssorban megadott kifejezés működő BPF-szűrővé alakításához. A mi szempontunkból most a tcpdump egyik ritkán használt kapcsolója, a -d lesz érdekes, amely a szűrő kódját nyomtatja ki. Például a `tcpdump host 192.168.9.10` csak azokat a csomagokat szippantja le, amelyek forrás- vagy célcíme 192.168.9.10. A `tcpdump -d host 192.168.9.10` kiírja a szűrő BPF-kódját, ahogy az a *3. listán* látszik.

Fűzzünk rövid megjegyzéseket a kódhoz: a 0–1. és 6–7. sorok ellenőrzik a protokollazonosító alapján, hogy az elfogott keret IP, ARP vagy RARP protokollt hordoz-e (lásd /usr/include/linux/if_ether.h). A protokollazonosító a keret 12. bájtjánál van (lásd 1. ábra). Ha a próba sikertelen, a csomagot eldobjuk (13. sor).

A 2–5. és a 8–11. sorok összehasonlítják a forrás és a cél IP-címét 192.168.9.10-zel. A protokolltól függően a címek elhelyezkedése más, ha a protokoll az IP – ekkor 26 és 30, egyébként 28 és 38. Ha az egyik cím egyezik, akkor a szűrő elfogadja a csomagot, és az első 68 bájtot elküldi az alkalmazásnak (12. sor).

A szűrőkód nem mindig a legelőnyösebb, mivel egy általános BPF-gépre hozták létre, és nem igazították rá arra a rendszerre, amely a szűrőmotort futtatja. Az LPF esetében a szűrőt a PF_PACKET feldolgozófüggvények futtatják, amelyek esetleg már előbb ellenőrizték az ethernetprotokollt. Ez a kezdeti socket() hívásban megadott protokollmezőtől függ: ha az nem ETH_P_ALL (ami azt jelenti, hogy az összes ethernetkeretet el kell kapni), akkor csak azok a keretek érkeznek meg a szűrőhöz, amelyek itt vannak meghatározva. Például egy ETH_P_IP foglalat esetén így írhatjuk át a szűrőt, hogy gyorsabb és rövidebb legyen:

```
(000)   ld    [26]
(001)   jeq   #0xc0a8090a   jt 4    jf 2
(002)   ld    [30]
(003)   jeq   #0xc0a8090a   jt 4    jf 5
(004)   ret   #68
(005)   ret   #0
```

A szűrő telepítése

Egy LPF telepítése egyszerű művelet: csak annyit kell tenned, hogy létrehozol egy sock_filter szerkezetet, amely a szűrőt tartalmazza és csatolod egy nyitott foglalathoz. A szűrő szerkezete könnyen megkapható, ha a tcpdump -d kapcsolóját -dd-re cseréled. A szűrő C tömbként nyomtatódik ki, amelyet beilleszthetsz a saját kódodba, ahogy a *4. lista* mutatja. Utána a setsockopt() hívással csatolod a szűrőt a foglalathoz.

Egy teljes példa

Cikkünket egy teljes példával zárjuk (lásd az 5. listát a ➲ ftp://ftp.ssc.com/pub/lj/listings/issue86/ címen). Ez teljesen hasonló az első két példához, csak hozzá van adva az LSF-kód és a setsockopt() hívás. A szűrő csak azokat az UDP-csomagokat engedi át, amelyek forrás- vagy célcíme a 192.168.9.10 és az UDP-forráskapu 5000.

Ha ki akarod próbálni ezt a példát, szükséged lesz egy egyszerű módszerre, amellyel tetszőleges UDP-csomagot tudsz előállítani (ilyen a sendip vagy az apsend, megtalálhatók a ➲ http://freshmeat.net/ címen). Valószínűleg az IP-címet is át szeretnéd állítani egy olyanra, amely a helyi hálózatodon előfordul. Ehhez a szűrő kódjában a 0xc0a8090a számot le kell cserélned az IP-cím tizenhatos számrendszerbeli alakjára.

Az utolsó megjegyzés az ethernetkártya akkori állapotára vonatkozik, amikor kilépsz a programból. Mivel az ethernet-állapotjelzőket nem állítottuk alaphelyzetbe, a kártya lehallgató módban marad. A megoldás az, hogy meg kell valósítanod a Control-C (SIGINT) jel kezelőjét, amely az ethernet-állapotjelzőket megelőző értékükre állítja vissza (ezeket közvetlenül azelőtt kell elmentened, hogy a VAGY-ot végrehajtanád az IFF_PROMISC-kal), mielőtt a program befejezi működését.

Összegzés

A csomagok helyi hálózatról történő leszippantása hálózati gondok vagy mérések esetén felbecsülhetetlen értékű ismereteket adhat. Néha a közismert eszközök – például a tcpdump vagy az ethereal – nem illeszkednek pontosan az igényeinkhez. Ilyenkor nagy segítség, ha képesek vagyunk saját szippantót írni. Az LPF-nek köszönhetően ez egyszerűen és hatékonyan megtehető.

Gianluca Insolvibile (g.insolvibile@cpr.it.) a 0.99p14-es rendszermag-változat óta lelkes Linux-rajongó. Jelenleg hálózatokkal és digitális videókutatással, valamint -fejlesztéssel foglalkozik.

Kapcsolódó címek

➲ http://www.linuxpapers.org/
➲ http://www-nrg.ee.lbl.gov/nrg.html címen.
➲ http://www-nrg.ee.lbl.gov/nrg.html

Könnyű álom (7. rész)

A 2.4-es rendszermag védelmi lehetőségei (2. rész)

Cikkünk a Linuxvilág múlt havi számában megjelent írásunk folytatása. Elolvasása előtt javasoljuk az előző rész újraolvasását. A korábbi cikk sokadszori (sajnos már elkésett) átolvasása után ismertük fel (köszönet *Kadlecsik Józsefnek* és *Kis Szabó Andrásnak*), hogy a magyarítás kissé zavaróra sikeredett. A magyar *szűrő* kifejezés olyan sok értelemben tűnt föl a cikkben, hogy avatatlan olvasó elbizonytalanodhatott. E hibát igyekszünk most kijavítani. Idézzük fel röviden az előző cikk tartalmát (most már a helyes szóhasználattal)!

A Netfilter a Linux-rendszermag része. Keretrendszer, amelyet a hálózaton közlekedő csomagok kezelésére hoztak létre. Olyan egységekből áll, ezek a hálózaton közlekedő csomagok jellemzői alapján szükség esetén annak továbbítását megtagadják, vagy fejlécében módosításokat végeznek.

Az IP Tables a Netfilter-rendszer része, segítségével az IP és az arra épülő protokollok csomagjainak módosítására nyílik lehetőség. Több részből áll, melyek feladatai bizonyos jellemzőkkel rendelkező csomagok eltávolítása, így ezzel egyfajta tűzfalszolgáltatást adva szükség esetén a csomagok forrás- vagy célcímét módosítják.

Az egyes modulok (table) végzik a csomagok tulajdonságainak tulajdonképpeni figyelését és módosítását. Minden alapmodul tartalmaz beépített szűrőláncokat, de szükség esetén a felhasználó is létrehozhat ilyet. A szűrőláncok szabályokat tartalmaznak. Egy szabály két részből áll: feltételből (ezeket az előző alkalommal áttekintettük) és műveletből. A feltételek több elemi feltételből is állhatnak, melyek logikai ÉS kapcsolatban állnak egymással. Ha minden elemi feltétel teljesül, akkor a rendszer végrehajtja azokat a megadott műveletet, amelyekről a továbbiakban szó lesz.

A rendszernek szóló utasítások

Amennyiben az adott csomag minden feltételnek megfelel, akkor a szabályban megadott művelet hajtódik végre. A rendszerben négy egyszerű beépített művelet van: a DROP, az ACCEPT, a RETURN és a QUEUE.

ACCEPT esetén az adott csomag további vizsgálatok nélkül továbbításra kerül, DROP esetén a rendszer az adott csomagot visszajelzés és újabb vizsgálatok nélkül eldobja. A RETURN művelet a csomag vizsgálatát visszaadja a hívó szűrőláncnak, a QUEUE pedig a csomagot egy felhasználói programnak küldi további feldolgozásra.

A fenti beépített műveleteken kívül két dolgot tehetünk: használhatunk kiterjesztett műveleteket, amelyek külön egységként tölthetők be, vagy az ellenőrzést egy, a felhasználó által létrehozott szűrőláncra léptethetjük. A kiterjesztett műveletek nagy részének hatását további értékekkel lehet finomítani. A további alcímek alatt a használható műveleteket mutatjuk be.

LOG-művelet

A feltételeknek megfelelő csomag megadott jellemzőit a rendszermag naplózó alrendszerén keresztül naplózza. Nekem az a véleményem, hogy az ipchains-rendszerben szintén megoldás nagyon jól használható volt, így célszerű lenne azt is támogatni. Az ipchains-rendszerben a naplózás független volt az adott szabály műveletétől, így bármely végrehajtott művelet naplózható volt. Mivel az IP Tables-rendszerben ilyen jelenleg nem létezik, az ember kénytelen először naplózni, hogy mit szándékozik tenni az adott csomaggal, és csak azután térhet rá a tényleges műveletre. Ez több szempontból is kellemetlen. Mi történik akkor, ha a naplózás és a végrehajtott művelet feltételrendszere egymástól elcsúszik (élő rendszereknél ez könnyen előfordulhat)? Így naplózzuk, hogy egy adott csomagot kitiltunk, de valójában nem tesszük meg… További felmerülő gond a beállítás átláthatóságának csökkenése. Amíg az ipchains-rendszeren a naplózás csak egy beállítás volt egy szabály bevitelénél, addig itt egy további szabály. Ez magától értetődően csökkenti az átláthatóságot. A kellemetlenségek mellett van azonban egy vitathatatlan jó tulajdonsága az új rendszernek: a finomhangolás lehetősége. Így a szokványos események más naplószinten (`log-level`) naplózhatók, mint az azonnali beavatkozást igénylők. Az esetleges hibák felderítésére a naplók kiegészíthetők a felhasználó által megadott szöveggel – így annak keletkezési helyére akár a naplóban is utalhatunk. Szükség szerint a csomagok alacsony szintű jellemzőit is naplóztathatjuk.

A lehetséges értékek az alábbiak:

`--log-level` *szint*

Ezzel a naplózási szint állítható be vele. A naplózás szintjeiről bővebb tájékoztatás kapható a naplózó alrendszer leírásánál (például *man 5 syslog.conf*).

`--log-prefix` *előtag*

A naplózott sorok előtagja határozható meg. Az előtag legfeljebb 29 betűs lehet. Nagyszerűen használható a fontosabb sorok kiemelésére és a nyomkövetésnél is jó hasznát vehetjük.

`--log-tcp-sequence`

Naplózza a TCP-sorozatszámot. Ha a felhasználók olvashatják a naplóállományt, akkor ez komoly biztonsági kockázattal jár. A naplóállományok felhasználók általi olvashatóságát nem javasoljuk, íly módon ez a gond elkerülhető.

`--log-tcp-options`

Naplózza a TCP-csomag *Options* (Tulajdonságok) mezőjét.

`--log-ip-options`

Naplózza az IP-csomag *Options* (Tulajdonságok) mezőjét.

REJECT-művelet

Mivel a beépített DROP minden visszajelzés nélkül dobja a padlóra a csomagokat, annak küldője nem tudja, hogy valami gond van az adatátvitellel. E nehézség megoldására született a REJECT-művelet, amely hasonló az ipchains-rendszerben megismertéhez, de annál jóval okosabb. Itt meghatározható ugyanis, hogyan jelezzen hibát a csomag küldőjének. Ha az ipchains-rendszer REJECT-műveletét használtuk, akkor az `port unreachable` ICMP-csomaggal utasította el a kérést. Ebből TCP-kapcsolat esetén nyilvánvalóan látszott, hogy valójában a csomagszűrő utasította el a kapcsolatot, és nem arról van szó, hogy az adott kapun nem figyel semmi. Az IP Tables REJECT-moduljával ez a gond megoldható, mindössze annyit kell tennünk, hogy a TCP-kapcsolatot RESET TCP-csomaggal utasítjuk el.

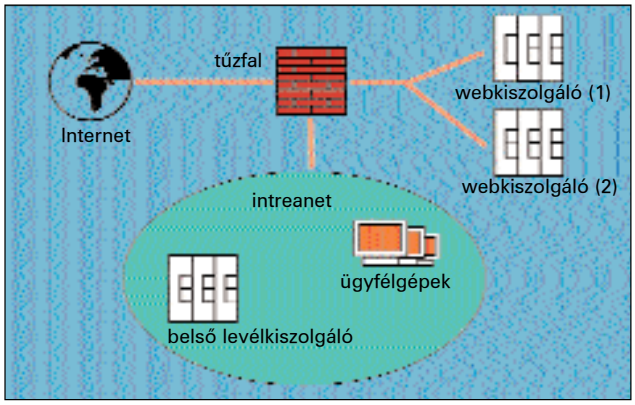
Alkalmazására csak az INPUT, FORWARD és az OUTPUT szűrőláncban vagy kizárólag az előbb említettekből hívott felhasználói szűrőláncban van lehetőség.

A válasz típusát az alábbi kapcsolóval tudjuk befolyásolni:

`--reject-with` *típus*
A lehetséges választípusok:
`icmp-net-unreachable`
`icmp-host-unreachable`
`icmp-port-unreachable` (ez az alapértelmezett)
`icmp-proto-unreachable`
`icmp-net-prohibited`
`icmp-host-prohibited`
ICMP echo request csomag esetén válaszul adható:
`echo-reply`
Csak TCP-kapcsolat esetén lehetséges válasz:
`tcp-reset`

MARK-művelet

Segítségével beállítható a csomaghoz kapcsolódó jelzés. Ezt a jelzést a későbbiekben további szűrésre használhatjuk (lásd mark feltétel) vagy `policy routing` használata esetén a csomagok előjelölésére. Ez utóbbi lehetőségről olvashatunk bővebben itt [1]. A MARK-művelet csak a mangle-modulban alkalmazható.
`--set-mark` *jelzés*



TOS-művelet

Az IP-fejléc 8-bites *Type of Service* (ToS) mezőjét állíthatjuk be a segítségével.
Használata lehetővé teszi a `policy routing` használatát összetett, több Linux-alapú útválasztóból vagy tűzfalból álló rendszereken. Csak a mangle-modulban használható.
`--set-tos` *tos*
Használhatók számok vagy a következő nevek:

- Minimize-Delay 16 (0x10),
- Maximize-Throughput 8 (0x08),
- Maximize-Reliability 4 (0x04),
- Minimize-Cost 2 (0x02),
- Normal-Service 0 (0x00).

SNAT-művelet

A forráscím átírására használható. Kizárólag a nat-modulban alkalmazható, a POSTROUTING szűrőláncban. Egyetlen kapcsolója van:
`--to-source` *IP-cím*`[-`*IP-cím*`][:`*kapu-kapu*`]`
Meghatározható vele egy IP-cím, IP-címek egy zárt tartománya, illetve lehetőség van egy kaputartomány megadására (ezt csak TCP- vagy UDP-protokoll esetén lehet használni). Ha a kaputartomány nincs megadva, akkor a rendszer az 512-es alatti kapukat lehetőség szerint a továbbítás alatt is 512 alatt tartja, az 513 és 1023 közötti kapukat 1024 alatt, az efölöttieket pedig 1023 fölött továbbítja. Amennyiben lehetséges, a kapuk a továbbítás alatt nem változnak meg.
Ha IP-címtartományt adunk meg, akkor minden kapcsolat azzal az IP-címmel tart kapcsolatot a továbbiakban, amivel a kapcsolat felépült.

DNAT-művelet

A csomagok célcímének átírását teszi lehetővé. Csak a nat-modulban használható, a PREROUTING és az OUTPUT szűrőláncokban vagy csak a belőlük hívott felhasználói szűrőláncokban. Egyetlen kapcsolója hasonló a SNAT-hoz:
`--to-destination` *IP-cím*`[-`*IP-cím*`][:`*kapu-kapu*`]`
Megadható vele egy új cél-IP-cím, egy zárt címtartomány (itt is követi a rendszer a már felépült kapcsolatokat, és a továbbiakban a kezdeti jellemzőkkel tart kapcsolatot), és lehetőség van a célkapu megváltoztatására is.
Amennyiben a célkapu nincs megadva, akkor nem változik meg.

MASQUERADE-művelet

Hasonló feladatot lát el, mint a SNAT, csak itt nem kell a kimenő IP-címnek állandónak lennie. Ez a művelet használandó tehát dinamikus IP-címkiosztású rendszereknél (például modemes behívás, ISDN, ADSL) a rendszer mögött lévő hálózat forgalmának továbbítására. A lehetőség sokak előtt ismert az ipchains-rendszerből.
A hálózati szolgáltatók őszinte bánatára itt már gyerekjáték átállítani a forráskapu-tartományt, így az ilyen címfordítást használó rendszert nem olyan egyszerű felismerni. (A szolgáltatók gyakran adnak el egygépes széles sávú szolgáltatást olcsóbban. A felhasználók szeretik olcsón megvenni az egygépes szolgáltatást, majd több gép hálózati forgalmát azon keresztül bonyolítani. Mivel az ipchains alapértelmezés szerint a 61000-es és a 65096-os kapu közül indította az ilyen kapcsolatokat, ami általában nem szokása semmilyen operációs rendszernek, viszonylag könnyű volt a turpisságot érzékelni. Természetesen a rendszermag kis módosításával a dolog áthidalható volt, de ezt nem mindenki ismerte, illetve hibát is okozhatott.
Egyébként más, kifinomultabb módszerek is léteznek az ilyen rendszer felfedezésére – ez esetleg egy későbbi cikk tárgya is lehet.
A MASQUERADE annyival több az adott kimenő IP-címen történő SNAT-nál, hogy a hálózati csatoló lekapcsolásakor önműködően elfelejti addig rögzített kapcsolatait. Így egy következő behívás nem okozhat gondot. Kapcsolója:
`--to-ports` *kapu*`[-`*kapu*`]`
Ez határozza meg a fent említett kaputartományt, felülbírálva a SNAT-nál már ismertetett eljárást. Csak TCP- és UDP-protokolloknál alkalmazható.

REDIRECT-művelet

A bejövő vagy kimenő kapcsolatok helyi kapuba való irányítását teszi lehetővé. Mi ennek az értelme? Ez teszi lehetővé átlátszó tűzfalak és proxyk létrehozását. Megadhatjuk, hogy az áthaladó webkérések legyenek egy squidba irányítva. Kapcsolója:
`--to-ports` *kapu*`[-`*kapu*`]`
A célkaput vagy kaputartományt határozza meg. Alapértelmezésben a célkapu nem változik.

Csomagszűrés a gyakorlatban

Az eddig említett lehetőségek szemléltetésére létrehoztunk egy példahálózatot. Az *ábrán* látható az elképzelt rendszer felépítése. A közepes cég logikai hálózata három alapvető részből áll: az Internetből, a cég belső hálózatából (intranet) és egy kevésbé védett, úgynevezett szabad területből (DMZ), amelyben két webkiszolgáló található. A webkiszolgálókat azért célszerű leválasztott hálózatrészbe tenni, mert ezek a legerősebben támadott rendszerek, gyakran változó tartalommal. A változó tartalmat kiszolgáló programok tervezésénél általában nem a biztonság a legfőbb szempont, így ezek sajnos gyakran válnak a betörők áldozataivá. Ha egy webkiszolgáló külön hálózatban van, akkor esetleges feltörése esetén sem különösebben veszélyezteti a belső hálózat biztonságát.

iptables.conf

```
# a nat-modul beállításai
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]

# 1. példa
-A PREROUTING -p tcp -i eth0
              ➥-m tcp --destination-port www
              ➥-j REDIRECT --to-ports 3128
-A PREROUTING -p tcp -i eth0 -m
              ➥tcp --destination-port https -j
              ➥REDIRECT --to-ports 3128
# 2. példa
#-A PREROUTING -p tcp -d <interIP> -m
               ➥tcp --destination-port www -j
               ➥DNAT --to-destination
               ➥10.0.1.11-10.0.1.12
# 3. példa
-A POSTROUTING -p tcp -o comx0 -m
               ➥tcp --destination-port telnet
               ➥-j MASQUERADE

COMMIT

# a mangle-modul beállításai
*mangle
:PREROUTING ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
COMMIT

# a filter-modul beállításai
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
:spoof - [0:0]
:interIN - [0:0]
:interLO - [0:0]
:intraIN - [0:0]
:intraLO - [0:0]
:dmzIN - [0:0]
:dmzLO - [0:0]

-A INPUT -i lo -j ACCEPT
-A INPUT -j spoof
# a saját lábunkra érkező forgalom
-A INPUT -i eth0 -d 10.0.0.1 -j intraLO
-A INPUT -i eth1 -d 10.0.1.1 -j dmzLO
-A INPUT -i comx0 -d <interIP> -j interLO
-A INPUT -j LOG --log-prefix "input catch all "
-A INPUT -j DROP

-A FORWARD -j spoof
# az átmenő forgalom
-A FORWARD -i eth0 -j intraIN
-A FORWARD -i eth1 -j dmzIN
-A FORWARD -i comx0 -j interIN
-A FORWARD -j LOG --log-prefix "forward catch all "
-A FORWARD -j DROP

################## SPOOF szűrőlánc a címhamisítás
##megakadályozására ##############################
################## Embereknek, akik nem bíznak a
##magfejlesztőkben :) ############################

-A spoof -i eth0 -s 10.0.0.0/24 -j RETURN
-A spoof -i eth0 -j LOG --log-prefix
       ➥"spoof cache all eth0 "
-A spoof -i eth0 -j DROP

-A spoof -i eth1 -s 10.0.1.0/24 -j RETURN
-A spoof -i eth1 -j LOG --log-prefix
       ➥"spoof cache all eth1 "
-A spoof -i eth1 -j DROP

# Internetről jönne nem szabályos forráscímmel
-A spoof -i comx0 -s 10.0.0.0/8 -j DROP
-A spoof -i comx0 -s 172.16.0.0/12 -j DROP
-A spoof -i comx0 -s 192.168.0.0/16 -j DROP
-A spoof -i comx0 -s 224.0.0.0/4 -j DROP
-A spoof -i comx0 -s 240.0.0.0/5 -j DROP
-A spoof -i comx0 -j RETURN
-A spoof -j LOG --log-prefix "spoof catch all "
-A spoof -j DROP
COMMIT

############### A lábakra bejövő és az áthaladó
##forgalom irányítása ###########################

# if_intranet helyi lábra érkező forgalma
# 4. példa
-A intraLO -p tcp -s <in_mail> -m
           ➥tcp --source-port 1023:65535
           ➥--destination-port smtp -m mac
           ➥--mac-source <in_mail:mac_addr>
           ➥-j ACCEPT
-A intraLO -p tcp -m tcp --source-port
           ➥1023:65535 --destination-port
           ➥3128 -j ACCEPT
-A intraLO -p udp -m udp --source-port
           ➥1023:65535 --destination-port
           ➥domain -j ACCEPT
-A intraLO -p udp -m udp --source-port ntp
           ➥--destination-port ntp -j ACCEPT
-A intraLO -j LOG --log-prefix
           ➥"intraLO catch all "
-A intraLO -j DROP

# if_intranet átmenő forgalma
-A intraIN -p tcp -m tcp --destination-port ftp
           ➥-j ACCEPT
-A intraIN -j LOG --log-prefix
           ➥"intraIN catch all "
-A intraIN -j DROP

# if_internet helyi lábra érkező forgalma
-A interLO -p tcp -m tcp --source-port
           ➥1023:65535 --destination-port
           ➥smtp -j ACCEPT
```

A rendszer határvédelmi tervét a Linuxvilág február-márciusi számának 58. oldalán találjuk. A táblázat a korábban tárgyalt [4.] módszerrel írja le, hogy a rendszer egyes részei milyen erőforrásokhoz férhetnek hozzá. Ne felejtsük el: ennek a példarendszernek egyetlen feladata a csomagszűrés lehetőségeinek bemutatása – a gyakorlatban kissé finomabban hangolt védelmi rendszerekre van szükség.

A csomagszűrő betöltése

A csomagszűrő beállításait kétféle módszerrel érdemes az operációs rendszer elindulása után visszatölteni. Az egyik módszer szerint írunk egy parancsfájlt, amely `iptables` parancsok kiadásával egyenként beviszi az előre elkészített csomagszűrő-beállításokat. Egy ilyen parancsállomány tartalmazhat további beállításokat (a szükséges rendszermagmodulok betöltése, `sysctl`-beállítások stb.). Több ilyen letölthető az Internetről, ahol csak a csomagszűrő hangolását kell elvégeznünk.

Elegánsabb módszer az `iptables-save` és az `iptables-resore` programok használata. Ezek segítségével egy memóriában beállított csomagszűrő egy állományba menthető, illetve onnan visszatölthető. Mivel formátuma meglehetősen egyszerű, akár eleve ebben a formátumban megírhatjuk a csomagszűrő beállítóállományát, így pedig az iptables-restore programmal könnyen betölthető. Mi az utóbbi lehetőség mellett döntöttünk. Beállítóállományunk a *listán* látható.

Egy dolgot érdemes megemlíteni ezekkel a parancsokkal kapcsolatban. Képesek mind a szűrőláncokhoz, mind az egyes szabályokhoz tartozó csomag- és bájtszámlálók mentésére és visszaállítására.

A rendszer szűrési lehetőségei

A 4. példa összetettebb szűrést mutat be. A cél az, hogy az intranetről közvetlenül kizárólag a belső levelezőlkiszolgáló tudjon levelet továbbítani a rendszeren keresztül. Ennek érdekében a rendszer MAC-címét is felvettük a feltételek közé. A megoldással kapcsolatban két gond merülhet fel. Egyrészt a hálózati kártyák MAC-címeit programból át lehet állítani, így a levelezőkiszolgáló leállása esetén a helyére állhat valaki. A másik gond az, hogy ez a megoldás csak a helyi ethernethálózaton működik. Tetszőlegesen bonyolult szűrések állíthatók össze a beépített és a kiterjesztett feltételek alkalmazásával.

Példaműveletek

A levéltovábbító protokoll (SMTP) egy sajátosságán keresztül mutatjuk be az egyik legjobban használható műveletet, a REJECT-et. Levél továbbításakor a levelezőkiszolgálók egy része a feladó gép *auth*-kapujához fordul. Erről korábban már volt szó [4.], ott azonban a rendszer a *port-unreachable* ICMP-üzenettel tért vissza, mivel az ipchains REJECT művelete nem volt hangolható. Az üzenet miatt a figyelmes szemlélő azonnal rájöhetett, hogy csomagszűrő utasította el, és az adott szolgáltatás bizonyos feltételek mellett akár el is érhető. Mivel az IP Tables által adott REJECT-műveletnek meg lehet mondani, hogy mit adjon vissza, a rendszer tökéletesen tud szimulálni egy olyan gépet, amelyen az adott kiszolgálóprogram valóban nem figyel. Ezt láthatjuk az 5. példában. (➲ http://www.linuxvilag.hu/konnyualom)

Mint azt már korábban is említettük, a naplózás továbbfejlődött az ipchainsben megszokotthoz képest, ez azonban néhány kellemetlenséget is magával hozott.

Minthogy a LOG művelet nem köthető más valóban érdemi műveletekhez, a naplózni kívánt csomagok két szabály bevezetését kívánják meg az IP Tables-rendszerben (6. példa ➲ http://www.linuxvilag.hu/konnyualom). Ha az ilyen érdemi változást nem okozó műveletek összevonhatók lennének a valóban tiltó vagy engedélyező műveletekkel, akkor a beállítások kissé átláthatóbbak lennének.

A címátírás használata

Első három példánk olyan címfordítási helyzeteket mutat be, amelyek gyakran előfordulnak a gyakorlatban. Az 1. példa egy átlátszó webgyorstár csomagszűrő részét mutatja be. A csomag a belső hálózat felől érkezik és egy Interneten lévő webkiszolgálóhoz megy. Mivel a hálózat terheltségét erősen csökkentheti egy webgyorstár, de minden gépen beállítani sok munkát jelentene, célszerű valamilyen átlátszó megoldást létrehozni. Az IP Tables nat-modulja a PREROUTING szűrőláncban átírja a csomagok célcímét a helyi gép címére, a célkaput pedig 3128-ra. Így a kapcsolatot a helyi webgyorstár szolgálja ki.

A 2. példa terheléselosztást mutat be a DMZ-területen lévő két webkiszolgáló között. Az Internetről bejövő kapcsolatot a rendszer az egyik webkiszolgálónak adja és követi a kapcsolatot annak bontásáig. Így az egyszer felépült TCP-kapcsolathoz tartozó csomagokat minden esetben a megfelelő kiszolgáló kapja meg. A harmadik példaként kiemelt sor pedig az IP-álcázás (masquerading) használatát mutatja be.

Miközben ismerkedtünk a csomagszűrő rendszer sajátosságaival, néhány alul- vagy nem dokumentált dolog is kiderült. A nat-modul használata során a csomagszűrésnél az alábbiak igazak: DNAT és REDIRECT használata esetén a filter-modul szűrőláncai a módosított célcímet kapják meg, az SNAT és MASQUERADE műveleteknél azonban a filter-modul szűrőláncaiban az eredeti forrás látszik.

A címátírás használata esetén a tcpdump (bővebben lásd a következő számban) használatával érdekes dolgokat tapasztalhatunk… Az átírt csomagok kimenő és bejövő jellemzői nem minden esetben egyeznek meg, így előfordulhat, hogy a célcím-átírás után a tcpdump nem tudja a kapcsolatot egységként látni.

Összegzés

Mindent egybevetve az IP Tables rendkívül hatékony csomagszűrő rendszer. Hely hiányában csak egy kis ízelítőt tudtunk nyújtani lehetőségeiből, de nem is a kimerítő tárgyalás volt a cél, hanem ezeknek a puszta bemutatása. Az IP Tables és a 2.4-es rendszermag más hálózati szolgáltatásainak segítségével szinte kimeríthetetlen lehetőségek állnak a rendelkezésünkre.

Hivatkozásjegyzék

[1.] *Bert Hubert*, *Gregory Maxwell*, *Remco van Mook*, *Martin van Oosterhout*, *Paul B. Schroeder* és mások: Linux 2.4 Advanced Routing HOWTO

[2.] *Paul 'Rusty' Russel*: Linux 2.4 Packet Filtering HOWTO ➲ http://netfilter.samba.org/

[3.] *Paul 'Rusty' Russel*: Linux 2.4 NAT HOWTO ➲ http://netfilter.samba.org/

[4.] Linuxvilág (II évf. 2-3. szám, 54. oldal): Könnyű álmok (4. rész) A hálózati határvédelem alapjai

Mátó Péter (atya@andrews.hu), informatikus mérnök és tanár. Biztonsági rendszerek ellenőrzésével és telepítésével, valamint oktatással foglalkozik. 1995-ben találkozott először linuxos rendszerrel. Ha teheti, kirándul vagy olvas.

Borbély Zoltán (bozo@andrews.hu), okleveles mérnök-informatikus. Főként Linuxon futó számítógépes biztonsági rendszerek tervezésével és fejlesztésével foglalkozik. A 1.0.9-es rendszermag ideje óta linuxozik. Szabadidejét barátaival tölti.

Utazás a Postfix körül (1. rész)

Kezdetben vala a Sendmail. Majd érkezett a Qmail. És lám, itt van a Postfix...

A két korábbi rendszer már régen ismert, kiforrott, a Postfixre azonban most kezdenek felfigyelni. A cikksorozat első részében bemutatjuk az írót és a program felépítését.

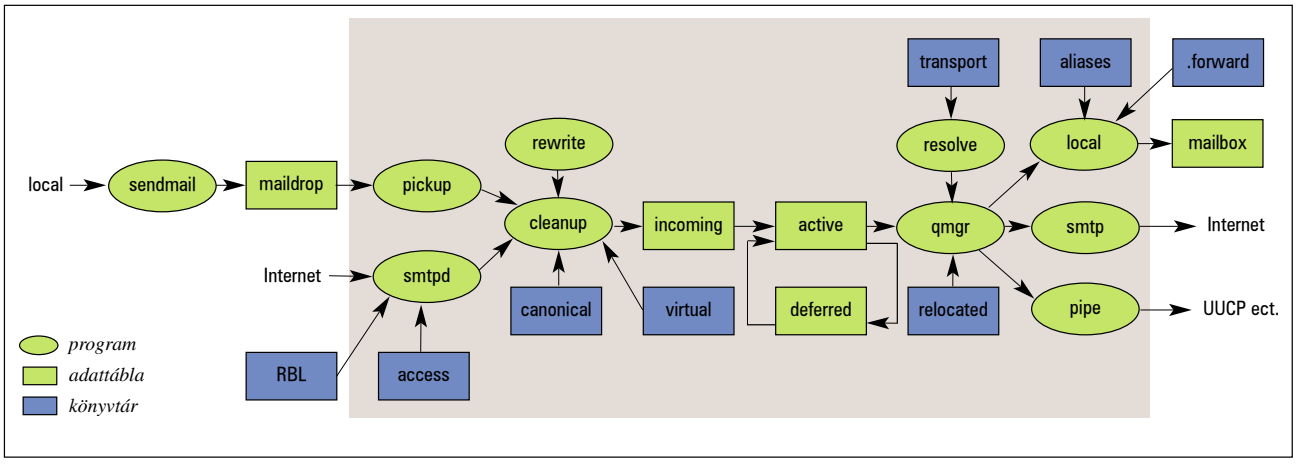
Wietse Venema: a Programozó

Mikor Isten a programozót teremtette, valószínűleg Venemára gondolt. A kódok tiszták, a programok felépítése átgondolt és logikus. Tekintsük át, milyen programok fűződnek a nevéhez. A TCT, azaz a The Coroner's Toolkit. E programon *Dan Farmer*-rel együtt dolgoztak. A programmal betörés után kísérelhetjük meg felderíteni annak menetét. Használata nem egyszerű, de megéri megtanulni. A következő a Satan rendszerátvizsgáló eszköz, mely nemcsak a hibát mutatja meg nekünk, hanem elhárítására megoldást is javasol. Ma már kicsit idejétmúlt, a Nessus nagyobb támogatottságot élvez ➲ http://www.nessus.org. Azután a TCP Wrapper, mely minden Linux-változatban megtalálható program, segítségével szabályozhatjuk a hálózati szolgáltatások elérését csomagszűrő vagy alkalmazásszintű tűzfal nélkül. És a rengeteg kis – főleg Solaris-rendszerre írt – programon kívül a `postfix`.
A felsorolt programok természetéből látható, hogy az író nem sorolható a kezdők közé, és rendkívül foglalkoztatja a biztonság témaköre.

A Postfix gondolatvilága

Amikor Venema elhatározta, hogy létrehozza a szerinte lehető legbiztonságosabb és leggyorsabb levelezőkiszolgálót, végignézte az addig megírt kiszolgálóprogramokat, majd összesítette azok előnyeit és hátrányait. A következő célt tűzte ki: az új kiszolgálónak legyen olyan sok szolgáltatása, mint a Sendmailnek, de túlbonyolított beállítások nélkül. Továbbá bizonyuljon legalább olyan biztonságosnak, mint a Qmail. És végül legyen a leggyorsabb. Elmondhatjuk, hogy egyedülálló mű született. Felépítését tekintve a Postfix moduláris, minden szolgáltatást külön programrész hajt végre, a jogosultságok pedig a lehető legalacsonyabb szintre kerültek. A rendszert két fő beállítófájlon keresztül tudjuk módosítani, felépítésüket nagyon könnyű megérteni. A programváltozat a következő állapotokat ismeri: a pillanatfelvételt (snapshot) és a teljes kiadást (release). A pillanatfelvételek olyan programrészleteket és programokat tartalmaznak, melyeket a szerző még nem tart véglegesnek, míg a kiadás már a hivatalos változatot, ebből hiányoznak a kísérleti jellegű vagy a nem százszázalékosan kidolgozott programok. A pillanatfelvétel-változatokról egy rövid megjegyzés: Venema a saját levelezőkiszolgálójára is ezeket telepíti. A programok ezen állapotbeli üzembiztonságát sok másik program megirigyelhetné.
A rendszert úgy tervezték, hogy a távoli hálózatokkal a lehető legkevesebb modul érintkezzen, azok pedig a legkisebb jogosultsági szinttel rendelkezzenek. Aki megpróbál egy Postfix-alapú levelezőrendszert feltörni, annak több szinten kell keresztülverekednie magát, mire egy olyan programhoz jut, ami rendszergazdai jogosultságokkal fut. A programokat és a rendszer felépítését részletesen az *1. ábra* mutatja be, ennek segítségével kövessük egy levél fogadását.

- `sendmail`
  Ez a kis program gyakorlatilag egy burkoló (wrapper), a sendmailt helyettesíti, hogy az ahhoz megírt programok helyi kéréseit átfordítsa a `postfix` által is értelmezhető hívásokká, és letegye azt az előírt könyvtárba. Ha a könyvtár mindenki által írható (world writeable), akkor egyedül is meg tudja ezt tenni, ha azonban csak egy adott csoport számára elérhető – az alapértelmezett csoport a `maildrop` – akkor a `postdrop` programot használja erre a célra, ami szintén a `postfix` rendszer része.
- `pickup`
  Begyűjti a leveleket a `maildrop` könyvtárból és átadja őket a `cleanup` programnak (magyarázatát lásd később).
- `smtpd`
  Fogadja a belső hálózaton vagy az Interneten keresztül érkező leveleket és különböző ellenőrzések után átadja azt a `cleanup` programnak. Itt tudjuk előírni, hogy vizsgálja meg: a küldő fél nyitott továbbító-e (`open-relay` – mindenki tud rajta keresztül levelet küldeni akárhova, a levélszemetelők kedvenc célpontja), vagy sem.



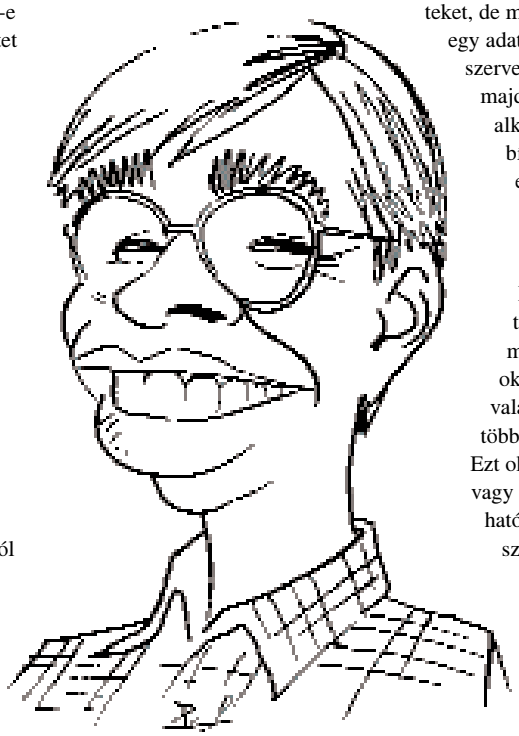
Bejövő levelek ellenőrzése

Ellenőrzi, hogy nem tiltottuk-e ki a küldőt más módon, például olyan táblázatokat használva, amelyek a küldőre vonatkoznak, akár hálózati cím, egész tartománynév vagy küldő személy szerint. De itt akár engedélyezhetjük azt is, hogy elfogadjuk a levelet egy olyan levelezőkiszolgálótól, amelyik benne van a nyitott továbbítókat nyilvántartó adatbázisban.

- `cleanup`
  Ellenőrzi a levél fejlécét, hogy megfelel-e az előírásoknak: a címzett megtalálható-e a rendszeren, kell-e valamilyen műveletet végrehajtani a levél fejlécén – például a címzett nevének átírását –, esetleg meg kell-e hívnia a `trivial-rewrite` programot. Ha végzett, akkor „lerakja” – fájlba menti – az üzenetet az `incoming` sorba (queue). Ebbe a sorba kerülnek tehát azok a levelek, melyeket a kiszolgáló elfogadott.
- `trivial-rewrite`
  Szükség esetén átírja a levél fejlécét, és visszaadja azt a `cleanup` programnak. A programot a `qmgr`, azaz a „sorkezelő” is meg tudja hívni.
- `qmgr`
  Az `incoming` sorból áthelyezi az üzeneteket az `activ` sorba, valamint felügyeli a levél sorsát. Az `activ` sorból kerülnek elküldésre a levelek helyileg vagy távolra. A sor mérete korlátozott, nehogy túlterheljék a kiszolgálót, olyan módon például, hogy túl sok levelet próbálnak vele elküldetni, és így nem marad elég memória az egyéb tevékenységekre, vagy éppen csak a levelek fogadására. A `qmgr` arra is felügyel, hogy azok a levelek, melyeket először próbál elküldeni – akár helyileg, akár távolra –, elsőbbséget élvezzenek azokkal szemben, melyeket nem sikerült először elküldeni, és ezért későbbi továbbításra várnak. A levelek innen ahhoz a programhoz kerülnek, amit a továbbítás megkövetel. Ezek a `local`, `smtp` és a `pipe` programok. A Postfix pillanatfelvételeiben elérhető még a `virtual` is, erre azonban a cikksorozat későbbi részében térek ki.
- `local`
  A helyi felhasználók számára fogadja a leveleket, ellenőrzi a `.forward` fájlok meglétét, valamint az `alias` adatbázist. A `.forward` fájlokat használjuk arra, ha egy programnak akarjuk elküldeni a levelet, vagy másik címre próbáljuk azt továbbítani. A rendszer az alábbi üzenettároló formátumokat ismeri: mbox és Maildir. A kettő közötti különbségről egy következő alkalommal írok.

A `.forward` fájlon keresztül a vezérlést más programoknak tudjuk adni, például a `procmail`-nek, bár a `local` önmagában is képes erre. Az `alias` adatbázist arra használjuk, amire a neve is utal: neveket leképezünk más nevekre. Jó példa erre a Kis.Ferenc kisf névre alakítása, amit már akármelyik unixos rendszer elfogad – gyanúm szerint a windowsos rendszerek is hasonló névleképezéssel oldják meg ezt a helyzetet. A cikksorozat következő részében ezt is részletesebben ismertetem majd.

- `smtp`
  A nem helyi felhasználók számára továbbítja az üzeneteket. Tehát ha én az ago@lsc.hu címről akarok levelet küldeni az info@linuxvilag.hu címre, akkor az én kiszolgálóm ezzel kapcsolódik a Linuxvilág kiszolgálójához. A távoli rendszerekkel csak smtp protokollon keresztül tartja a kapcsolatot (figyelem: nem az egész rendszer, csak ez a részprogram!).
- `pipe`
  Más vagy eltérő típusú rendszernek továbbítja az üzeneteket, de még a helyi rendszeren. Tehát én például egy adatbázisba továbbítom a leveleket, melynek szerver része fogadja a leveleket, letárolja azt, majd később elküldi azokat. Leggyakoribb alkalmazása, ha UUCP-n keresztül továbbítjuk a leveleket, használhatjuk azonban ezen keresztül az `lmtp` protokollt ismerő programokat is. Az `lmtp` protokollt használja például a `Cyrus-IMAP` rendszere is.



Wietse Venema, a Programozó

Ezek után a levél remélhetőleg kézbesítésre került. Ha ez mégsem történt volna meg, akkor visszapattan. Ennek kétféle oka lehet: elsőként a levél nem felelt meg valamilyen feltételnek és Postfixünk soha többé nem fog próbálkozni az elküldésével. Ezt okozhatja a levél tartalma, a küldő személye vagy más is, például a címzett már nem található meg a fogadó kiszolgálón. A második szintén nem túl szerencsés eset, ha a levelet csak átmenetileg nem tudjuk elküldeni, például a fogadó levelezőkiszolgáló időleges hibája miatt. Ekkor a levél `deferred` jelzőt kap, és a rendszer adott időközönként, illetve időtartamig megpróbálja elküldeni. Azokról a levelekről, amelyeket visszautasítottak, egy bejegyzést készít a `bounce` program a bounce nevű sorba, majd a levél eredetijét vagy annak egy részét visszaküldi a feladónak, kiegészítve azt a hiba okával. A bounce sor nem tárolja el a leveleket! Ha a levelet csak időlegesen utasította vissza a fogadó fél, akkor egy bejegyzés kerül a defer sorba, a levelet magát pedig a deferred sorban helyezi el. Ezt a helyzetet szintén a `bounce` program kezeli le.

Van egy program, amihez hasonlót eddig egyetlen más levelezőrendszernél sem láttam, a `master` démon, amely mindig ott fut a háttérben. Ez felügyeli az összes többi démont, a rendszer állapotát, és nem engedi, hogy a rendszert alkotó programok „elvaduljanak” és megállásra vagy szolgáltatásmegtagadásra kényszerítsék a rendszert. Ha túl messzire merészkednének, akkor visszafogja őket. Mivel ez vezérli az összes többit, és a Postfix modularitása megengedi, könnyű olyan kiszolgálót készíteni, amelyik csak küldeni tud.

A következő részben olyan rendszert állítunk be, melynek teljes a szolgáltatáskínálata, visszautasítja a levélszemetelők küldeményeit, és nem engedi, hogy kiszolgálónkat nyitott továbbítóként használják.

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.

Webfejlesztés PHP 4.0 és FastTemplate 1.1.0 segítségével

Ismerkedjünk meg egy időt megtakarító megoldással: a FastTemplate 1.1.0-val!

A webfejlesztésről közismert, hogy nagyon nehéz lépést tartani az általunk használt programeszközök hihetetlen gyors fejlődésével. Egy évvel ezelőtt még csak nem is hallottam a PHP nevű parancsnyelvről, ma pedig teljes munkaidőmben PHP-programozással foglalkozom.

Két csapatunk van: az egyik kizárólag PHP-programokat ír RedHat Linuxos munkaállomásokon, a másik csak DreamWeaverrel dolgozik NT-alapú PC-ken. A marketingrészleg meghatározza az oldalak célcsoportját; a grafikuscsapat pedig az adatok alapján kialakítja a HTML-lap látványvilágát. Ezután a programozóké a főszerep, hogy a PHP képességeit kihasználva egységessé kovácsolják a HTML-lapokat, és az aktív adatokat beillesszék a lapvázlatokba.

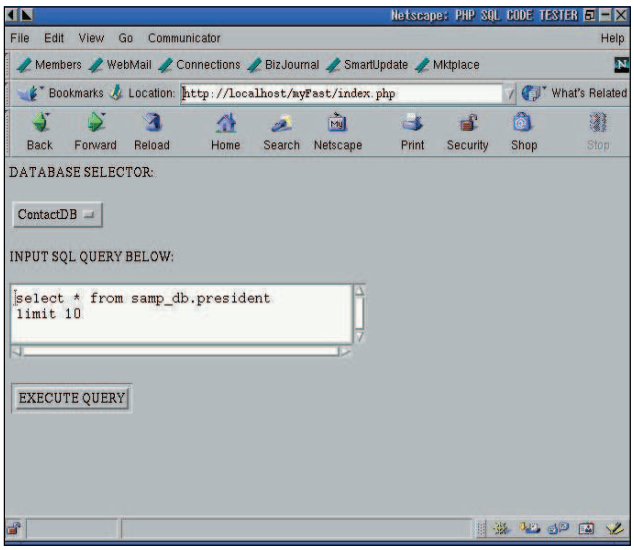
Mindez ragyogóan működik – egészen addig, amíg valaki mondjuk, meg nem akarja változtatni egy bizonyos gomb feliratát 46 különböző oldalon. Ez ugyanis azt jelenti, hogy 46 fájlban 46 változtatás szükséges csak azért, hogy egyetlen kis gomb másképpen nézzen ki! De hála az égnek, létezik a grep!

Az előbbi gondot azonban el is kerülhetjük, ebben kívánunk segítséget nyújtani ezzel az írással. A HTML-sablonok révén a fent említett gomb egyetlen fájl megváltoztatásával átalakítható lenne, ugyanis a sablonok használata lehetővé teszi, hogy a PHP-kódot elválasszuk a HTML-től. A PHP-kód módosítása nem lesz hatással a HTML-re, és fordítva.

A minták a következőképpen működnek: a PHP-program külső fájlokat nyit meg, és értelmez, amelyek a HTML-laphoz tartozó kódot tartalmazzák. A HTML fájlok az adatok helyén helyőrző jeleket (placeholders) tartalmaznak, amelyek futásidőben helyettesítődnek be. Ezek a tagok többnyire valahogy így néznek ki: `{ITEM}`. Ha a PHP-program felfedez egy tagot, behelyettesíti a megfelelő adatot, ezáltal a PHP kódfájlok és HTML fájlok teljesen elválasztva tárolhatók, így a programozók és a tervezők nem zavarják egymás munkáját.

Több nyílt forráskódú mintacsomag is elérhető, ezek között akadnak kicsit bonyolultabbak és hatékonyabbak is. Létezik azonban egy csomag, amelyben a végrehajtási sebesség és a használhatóság szerintem megfelelő egyensúlyban van, nevezetesen a CDI FastTemplate 1.1.0 programja. Szerzője, *Joe Harris* a General Artistic License felhasználási szerződés alapján adta ki művét. Joe a leírásban azt is közzéteszi, hogy a FastTemplate 1.1.0 tulajdonképpen egy azonos nevű Perl-modul PHP-változata. Az eredeti Perl-modul *Jason Moore* munkája.

A FastTemplate alapötlete: a honlapot szétbontjuk alapelemekre – egyedi gombokra, jelölőnégyzetekre vagy akár szövegsorokra. A táblákat fejlécre és sorokra kell törni, a sorokat pedig cellákra bontjuk. Minden elemhez saját HTML fájlt rendelünk, amelyben statikus (állandó) kód és egy, a változó adatot jelképező különleges tag található. A honlap ezekből az egyszerű elemekből épül fel. Végül egy olyan fájlt kapunk, mint amilyen a példában szereplő main.tpl, amely egyetlen `{BODY}` tagot tartalmaz. Néhányan ugyan jobban kedvelik a `{CONTENT}` elnevezést; de tulajdonképpen nem a név számít. Ezt a legfelső szintű tagot a teljes lap adataival fogjuk helyettesíteni. Ha szükséges, a honlap minden egyes lapjához létrehozhatunk egy-egy legfelsőbb szintű fájlt, melynek módosításával a teljes honlap kinézetét meg lehet változtatni. A példaprogramban is így fogunk eljárni.



1. kép Próbaoldal SQL lekérdezéshez

Lássuk, milyen előnyök származnak az eddig elmondottakból! Mivel a lapon található összes elemhez (widgethez) mintafájlokat szeretnénk készíteni, a minták használatával minden egyes lap tartalmának az utolsó bitig a helyén kell lennie. Ez jó, hiszen rákényszerít arra, hogy mérnökként tervezzünk, a meggondolatlan programozók életét viszont megkeseríti. Egy újabb lehetőség hozzáadása ezután nemcsak egyetlenegy, hanem legalább három fájl megváltoztatását igényli. Honlapunk ezáltal jobbá válik, hiszen tervezése átgondoltabb, részletesebb és finomabb lesz.

Elsőre talán úgy tűnhet, hogy rengeteg apró mintafájlt fogunk felhalmozni. Ez igaz is, de a mintafájlok széles körben újrafelhasználhatók. Ahogy az alkalmazás mérete növekszik, úgy egyre kevesebbet kell felhasználni. Akár több adattag is elhelyezhető egyetlen mintafájlban, ez csökkenti ugyan ezek számát, de egyszersmind kisebb mértékben lesznek újra felhasználhatóak.

A FastTemplate értékeléséhez látnunk kell működés közben, valamint azt is, hogy mire képes a weblap karbantartása során. Tételezzük fel, hogy van elérésünk egy olyan linuxos számítógéphez, amelyen már fut az Apache, a PHP és valamilyen relációs adatbázis-kezelő (én MySQL-t használtam), és rendszergazdai jogosultsággal rendelkezünk.

A FastTemplate telepítése pontosan olyan könnyű, mint amilyennek látszik. A honlap a ➲ http://www.thewebmasters.net/php/FastTemplate.phtml címen érhető el. Töltsük le a FastTemplate-1_1_0.tar.gz fájlt amennyiben PHP 4.0-t használunk, egyúttal a diff fájlt is (php4.diff). Mozgassuk a letöltéseket a dokumentumgyökérbe és az `untar` parancs segítségével csomagoljuk ki.

Ez létrehoz egy FastTemplate/ nevű könyvtárat. Ha PHP 4.0-t használunk, helyezzük a php4.diff fájlt a FastTemplate/-be és futtassuk le a következő parancsokat:

1. lista mysql.php

```
<%/////////////// index.php ///////////////////////

// ha valaki a EXIT PROGRAM-ra kattint,
// megmutatjuk a kilépő képernyőt, és kilépünk

if (isset($goodbye)) {
   include ("./goodbye.php");
   }

include ("class.FastTemplate.php");
$tpl = new FastTemplate(".");

$tpl->define (array(main     => "main.tpl",
                    form     => "form.tpl",
                    select   => "select.tpl",
                    option   => "option.tpl",
                    submit   => "submit.tpl",
                    textarea => "textarea.tpl"));

$tpl->assign (array(TITLE    =>
             ➥"PHP SQL CODE TESTER",
              FORM_ACTION    => "mysql.php",
              FORM_METHOD    => "post",
              STRING1        =>
             ➥"DATABASE SELECTOR: ",
              STRING2        =>
             ➥"INPUT SQL QUERY BELOW: ",
              SELECT_NAME    => "database",
              SELECT_SIZE    => "1",
              SUBMIT_VAL     => "EXECUTE QUERY",
              TEXTAREA_NAME => "query",
              TEXTAREA_COLS => "40",
              TEXTAREA_ROWS => "3" ));

                    //Feltételezzük a MySQL
                    //használatát.
$host="localhost"; //Módosítsuk megfelelően
$user="bill";      // e három sor
$password="megan"; // beállításait.

mysql_connect($host, $user, $password);
             ➥$db_table = mysql_list_dbs();

//szerezzük meg a helyi adatbázisok nevét, és
// csatoljuk a selector lehetőséghez.

for ($i = 0; $i < mysql_num_rows($db_table);
$i++) {
    $optionval = mysql_tablename($db_table, $i);
    $tpl->assign(array(OPTION_TAG
                ➥=>"$optionval"));
    $tpl->parse(OPTIONS, ".option");
    }

$tpl->parse(SELECT, "select");
$tpl->parse(TEXTAREA, "textarea");
$tpl->parse(SUBMIT, "submit");
$tpl->parse(BODY, "form");
$tpl->parse(MAIN, "main");

$tpl->FastPrint();
exit;
%>
```

```
patch class.FastTemplate.php3 php4.diff
mv class.FastTemplate.php3 class.FastTemplate.php
```

Ezután mozgassuk át a class.FastTemplate.php fájlt a PHP include könyvtárunkba. Ha nem vagyunk biztosak benne, hogy hol található, ellenőrizzük a php.ini fájlt (alapértelmezett esetben a /usr/local/lib/-ben helyezkedik el). Keressük az `include path` beállítást!
A class.FastTemplate.php fájlt közvetlen módon, az `include( )` paranccsal is csatolhatjuk a PHP-programokhoz, ez azonban többletmunkával jár, én is többnyire megfeledkezem róla.
Készítsünk valahol a dokumentumgyökérben egy másik könyvtárat a példafájloknak! Ehhez szükségünk lesz az *1.* a *2.* és a *3. listában* található programokra. A 4. listát egyedi mintafájlokká kell széttördelni (lásd ➲ ftp://ssc.com/pub/lj/listings/issue86/).
A példa rövidsége és egyszerűsége ellenére jól mutatja, hogy a FastTemplate könnyedén képes olyan táblákat is kezelni, amelyeknek sorai és oszlopai egyaránt dinamikusan változhatnak. Az oldal igazából csak annyira lesz bonyolult, amennyire az oldalon lévő elemek bonyolultak.
Vessünk egy pillantást az első listára! Ahhoz, hogy használhassuk a FastTemplate-et, egyszerűen csak csatoltuk a class.FastTemplate.php fájlt, és létrehoztunk egy példányt. A példányosítóban látható `(".")` azt jelöli, hogy mintafájljaink a jelenlegi könyvtárban találhatók, ezek azonban akárhol lehetnének. A FastTemplate osztály négy fő eljárással bír: `define( )`, `assign( )`, `parse( )` és `print( )`.
A `define( )` eljárás a programunk által használt külső fájlneveket címkékhez csatolja. Megjegyzendő, a mintafájloknak nem kötelező tpl-re végződniük, és az is elképzelhető, hogy egy programban mindössze egyetlen `define( )` hívásra van szükség.
Az `assign( )` eljárás a `{}` jel nélküli adattagokat rendeli ahhoz az adathoz, amivel helyettesíteni akarjuk. Egy programban több `assign( )` hívás is lehet. Kezdjük mindig a legkisebb, legalapvetőbb összetevővel, és építsünk belőle egyre nagyobbat, bonyolultabbat. Én a `parse( )` eljárást értettem meg a legnehezebben. Ugyanis ez az eljárás értelmezi a mintát, végrehajtja az adattag-behelyettesítést, és az eredményt egy helyi változóban tárolja. Ez a helyi változó a `parse( )` első értéke. A második érték az a fájlkezelő (file handle), amit a `define( )` eljárással megadtunk. Ha a `parse( )` második értéke `"."` jellel kezdődik, akkor az új érték az előző végéhez kapcsolódik (amennyiben létezik előző érték). Ebben a példában ezt a módszert használjuk arra, hogy létrehozzuk az adatbázis-tartalomjegyzéket. A PHP-program igazi erősségét használjuk fel arra, hogy a megjelenítendő adatot betöltsük vagy kiszámítsuk.
Végül a `print( )` alapértelmezés szerint kiírja a legutolsó parse-eljárás eredményét. Természetesen egy korábban értelmezett eredményt is átadhatunk értékként. A lapon egynél több `print( )` is lehet, sőt néha ez az egyszerűbb módja a HTML-oldal megjelenítésének. Általában az oldal létrehozására több lehetőségünk is van. Amelyik működik és hatékony, az jó.
Akkor lássuk a példát! Ha a böngészővel betöltjük az első lista példáit, az *1. képhez* hasonló képernyőképet kell látnunk. Válasszunk egy adatbázist a Selectorban, gépeljünk be egy lekérdezést,

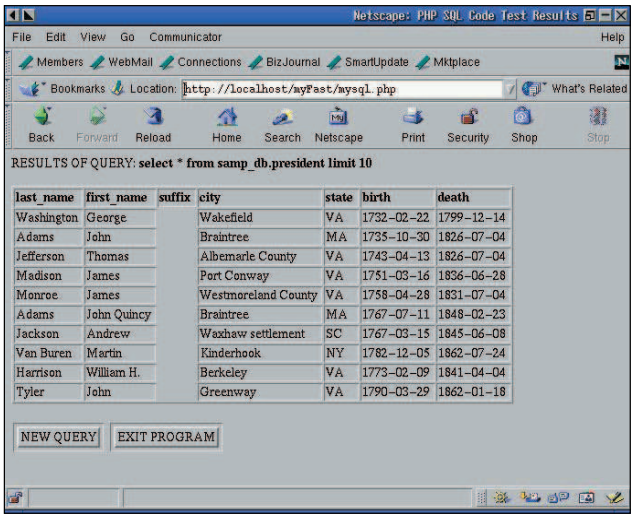
2. lista mysql.php

```
<%/////////// mysql.php ///////////////////////
include ("class.FastTemplate.php");
$tpl = new FastTemplate(".");
$tpl->define(array( main   => "main.tpl",
                    body       => "body.tpl",
                    table      => "table.tpl",
                    headerdata => "headerdata.tpl",
                    row        => "row.tpl",
                    rowdata    => "rowdata.tpl",
                    submit     => "submit.tpl",
                    form2      => "form2.tpl"));
$tpl->assign(array( TITLE     =>
                    ➥"PHP SQL Code Test Results",
                    SUBMIT_NAME => "SUBMIT",
                    SUBMIT_VAL  => "NEW QUERY",
                    FORM_ACTION => "index.php",
                    STRING1     =>
                    ➥"RESULTS OF QUERY: ",
                    QUERY       => "$query",
                    FORM_METHOD => "POST"));
                    //MySQL-t használunk.
$user="bill";       //Ebbe a három sorba
$host="localhost"; //saját beállításainkat
$password="megan"; //írjuk be.
mysql_connect($host,$user,$password);
mysql_select_db($database);
if(!strcmp($query, "")){
//ha nincs lekérdezés, állítsunk be egy
//alapértelmezettet
   $query = "SHOW TABLES";
   }
$query = stripSlashes($query);
$result = mysql_query($query);
//adjuk a fejléccellákat a fejlécsorokhoz.
for ($i = 0; $i < mysql_num_fields($result);
                                    ➥$i++) {
   $thval = mysql_field_name($result,$i);
   $tpl->assign(array(HEADERDATA => "$thval"));
   $tpl->parse(HEADER, ".headerdata");
   }
// adjuk a sorcellákat a sorokhoz, majd
// adjuk a sorokat a testhez.
for ($i = 0; $i < mysql_num_rows($result); $i++) {
   $row_array = mysql_fetch_row($result);
   for ($j = 0; $j < mysql_num_fields($result);
                                    ➥$j++) {
   $trval =  $row_array[$j];
   $tpl->assign(array(ROWDATA => "$trval"));
   $tpl->parse(ROW, ".rowdata");
   }
   $tpl->parse(ROWS, ".row");
   $tpl->clear("ROW");  //<- tegyük el
                        //megjegyzésbe ezt a
                        //sort és figyeljük meg
}                       //az eredményt.
$tpl->parse (TABLE, "table");
$tpl->parse (SUBMIT, "submit");
$tpl->assign(array(SUBMIT_NAME => "goodbye",
      ➥SUBMIT_VAL  => "EXIT PROGRAM"));
$tpl->parse (GOODBYE, "submit");
$tpl->parse (FORM2, "form2");
$tpl->parse (BODY, "body");
$tpl->parse (MAIN, "main");
$tpl->FastPrint();
%>
```

üssük le az EXECUTE gombot, és a lekérdezés eredménye máris megjelenik (lásd a *2. képet*).

Most lássuk a grafikai tervező szerepét! Az a feladatunk, hogy a teljes alkalmazás látványát megváltoztassuk. Képzeletbeli főnökünk éppen most mondta, hogy a szürke hátteret le kell cserélni sárgára. Továbbá azt szeretné, ha minden középre lenne rendezve, nem pedig bal oldalra. Ja igen, és az összes szöveg zöld színű legyen! Egy FastTemplate-alapú honlapon mindezeket a változtatásokat kevesebb, mint egy perc alatt elvégezhetjük a legfelsőbb szintű mintafájl, a main.tpl átszerkesztésével.

Bár jobb webszerkesztők is léteznek, a Netscape Composer is megteszi. A következő lépésekre lesz szükség: a Netscape Navigatorban válasszuk ki a *File/Open Page* menüpontot; gépeljük be az elérési utat vagy keressük ki a példakönyvtárunkban található main.tpl fájlt, majd a megjelenő párbeszédablakban bökjünk az *Open in Composer* gombra. A Netscape Composer egyetlen nagy ablakot fog megnyitni, ahol a szerkesztő részben mindössze a `{BODY}` szó olvasható (*3. kép*). Ez a main.tpl HTML-változata. A főnök által kért változtatások elvégzéséhez először válasszuk az *Edit/Select All* menüpontot. A `{BODY}`, beleértve a zárójelet is, most már sárgán virít. Ezután válasszuk a *Format/Align/Center* pontot, ezáltal elvégeztük a középre rendezést. Újfent jelöljünk ki mindent az *Edit/Select* All segítségével, majd válasszuk a *Format/Text Color* menüpontot, és a mintaválasztékból keressünk egy szép sötétzöldet. Bökjünk az *OK*-ra, és a `{BODY}` immár sötétzöldben pompázik.



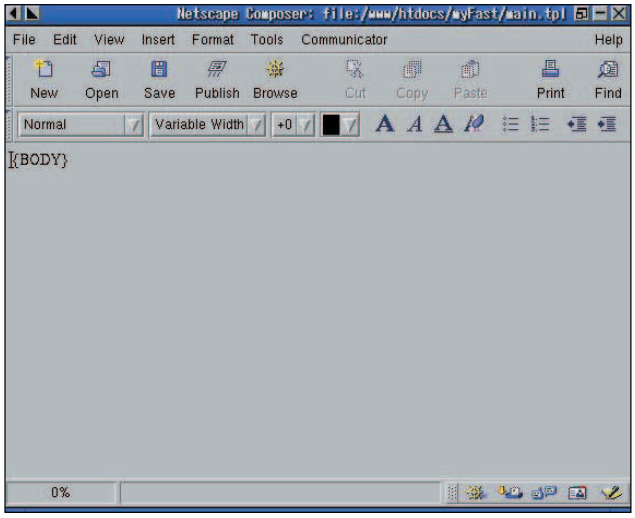
last_name	first_name	suffix	city	state	birth	death
Washington	George		Wakefield	VA	1732-02-22	1799-12-14
Adams	John		Braintree	MA	1735-10-30	1826-07-04
Jefferson	Thomas		Albemarle County	VA	1743-04-13	1826-07-04
Madison	James		Port Conway	VA	1751-03-16	1836-06-28
Monroe	James		Westmoreland County	VA	1758-04-28	1831-07-04
Adams	John Quincy		Braintree	MA	1767-07-11	1848-02-23
Jackson	Andrew		Waxhaw settlement	SC	1767-03-15	1845-06-08
Van Buren	Martin		Kinderhook	NY	1782-12-05	1862-07-24
Harrison	William H.		Berkeley	VA	1773-02-09	1841-04-04
Tyler	John		Greenway	VA	1790-03-29	1862-01-18

2. kép A próbalekérdezés eredménye

Főnökünk két kívánságát máris teljesítettük, csak egy maradt hátra. Válasszuk a *Format/Page Colors and Properties* menüpontot, ahol kattintsunk a *Use Custom Colors* gombra a *Colors and Background* fül tetején. A *Background*-ot választva a megjelenő választéktáblából keressünk egy vonzó sárgát. Kattintsunk az *OK*-ra, az

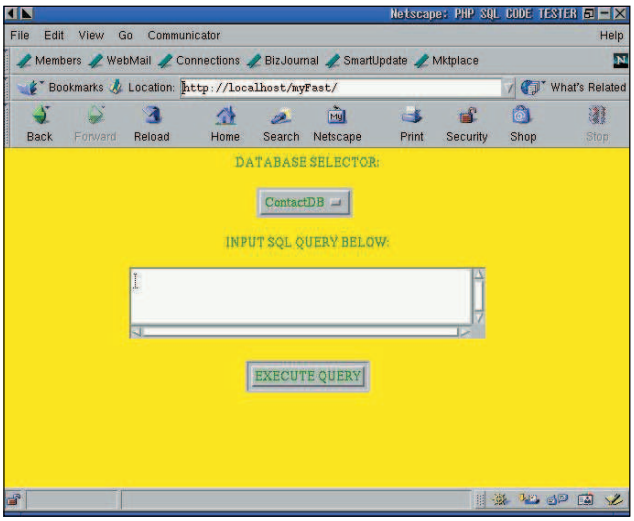
3. lista goodbye.php

```
<%/////////////// goodbye.php /////////////////

include ("class.FastTemplate.php");
$tpl = new FastTemplate(".");

$tpl->define(array(bye_message =>
                   ➥"bye_message.tpl",
                      main => "main.tpl"));

$tpl->assign(array(BYE_MESSAGE =>
        ➥"Thank you and please come again!"));

$tpl->parse (BODY, "bye_message");
$tpl->parse (MAIN, "main");
$tpl->FastPrint();

exit;
%>

Templates enable the physical separation of PHP
code and HTML. Changes to the PHP code in no
way affect the HTML, and vice versa.
```

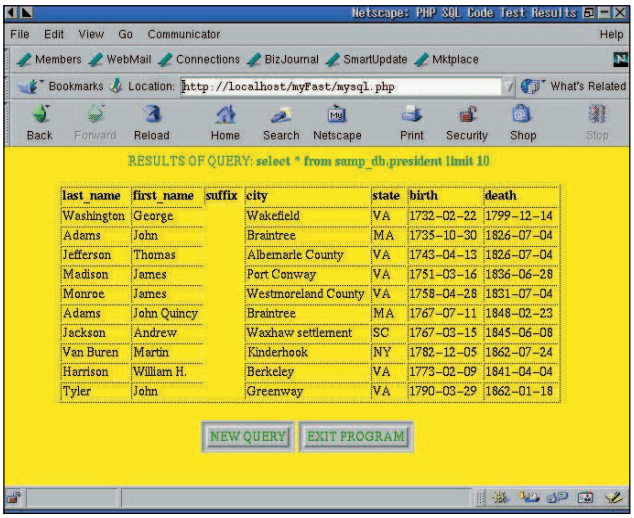


3. kép A fő sablonmodul szerkesztés közben

Apply-ra és újfent az OK-ra. A Composerben most már sárga a háttér. Ezután válasszuk a *File/Save*, és végül a *File/Close* menüpontokat.
Ha most újratöltjük a példa lekérdezőlapját, láthatjuk, hogy immár mindhárom változtatás érvényes: a háttér sárga, a szöveg zöld és minden középre rendezetten jelenik meg, ugyanakkor az alkalmazás programszerűen működik, mint eddig (lásd a *4.* és *5. képen*). Bár a példa igen egyszerű, azt hiszem, hatékonysága mindenki számára nyilvánvalóvá vált.
A FastTemplate-tel kapcsolatban néhány dolgot érdemes még megemlíteni. Először is, rengeteg apró fájllal dolgozunk, ezeket tudni kell kezelni, és a lap minden egyes elérésénél be kell olvasni a lemezről. Ez megterhelt webhelyen jelentős teljesítménycsökkenést okozhat, viszont nem túl gyakori elérésű helyeken lényegtelen szempont.
Az is elképzelhető, hogy minden egyes laphoz saját egyedi mintafájlt készít valaki, ezáltal is csökkentve a mintafájlok számát.



4. kép Az új sablon önműködően hat minden oldalra



last_name	first_name	suffix	city	state	birth	death
Washington	George		Wakefield	VA	1732-02-22	1799-12-14
Adams	John		Braintree	MA	1735-10-30	1826-07-04
Jefferson	Thomas		Albemarle County	VA	1743-04-13	1826-07-04
Madison	James		Port Conway	VA	1751-03-16	1836-06-28
Monroe	James		Westmoreland County	VA	1758-04-28	1831-07-04
Adams	John Quincy		Braintree	MA	1767-07-11	1848-02-23
Jackson	Andrew		Waxhaw settlement	SC	1767-03-15	1845-06-08
Van Buren	Martin		Kinderhook	NY	1782-12-05	1862-07-24
Harrison	William H.		Berkeley	VA	1773-02-09	1841-04-04
Tyler	John		Greenway	VA	1790-03-29	1862-01-18

5. kép A sablon hatása a lekérdezés kinézetére

Benjamin Kahn olyan weblapot tart fenn, amely a FastTemplate-hez való Cached FastTemplate nevű kiegészítéssel foglalkozik ➲ http://zoned.net:81/~xkahn/php/fasttemplate/. A Cached FastTemplate néhány további teljesítménynövelő lehetőséggel egészíti ki az eredeti csomagot. Ezzel a csomaggal a lapok egyes részeit a memóriában lehet tárolni, sőt ezek a gyorstárazó szabályok be is állíthatók. Ez mindenképpen megér egy pillantást, főleg ha már megkedveltük a FastTemplate hatékonyságát.
Ma már nem elegendő önmagában jó honlapot készíteni, nélkülözhetetlen, hogy méretezhető és karbantartható legyen. A DreamWeaver-szerű HTML-készítők használata ugyan igen könnyű, de az általuk készített HTML meglehetősen szegényes és csúnyácska. Ezekben a kesze-kusza HTML fájlokban senki sem szeret PHP-kódok után bogarászni egy egyszerű módosítás végrehajtásához. A HTML-minták használata tehát nagymértékben megkönnyítheti a webfejlesztők életét.

Bill Cunningham
nemrégiben vonult vissza az US Marine Corps-tól, ahol Solaris rendszergazdaként dolgozott. Jelenleg Észak-Karolinában Linux/MySQL-alapú honlapok webfejlesztőjeként a charlotte-i Heafner Tire Group alkalmazásában áll.

Teljesítménynövelés párhuzamosítással

Pörgessük fel és alakítsuk át unalmas soros eljárásainkat úgy, hogy többprocesszoros és megosztott memóriakörnyezetben is futhassanak.

Ebben az írásban azt szemléltetjük, miként lehet átalakítani egy soros eljárást úgy, hogy nagyobb hatékonysággal futhasson szimmetrikus feldolgozást végző (symmetric multiprocessing), illetve megosztott memóriát használó (distributed-memory) környezetekben. Ahhoz, hogy ezt a feladatot teljesíthessük, egyszerű alkalmazást fejlesztünk három lépésben; soros, többszálú és megosztott többszálú változatban.
A párhuzamos programozás elméleti szempontjain túl néhány gyakorlati kérdést is megvizsgálunk, amelyek programozás közben merülhetnek fel. A példákat a C++ és a POSIX-szálak (pthreads) segítségével készítettük el, a szimmetrikus és a megosztott feldolgozáshoz pedig az MPI programkönyvtárat használtuk.

A feladat megfogalmazása

A megoldandó feladat: prímszámok keresése adott határok között. A feladat előnye, hogy egyszerű, ugyanakkor ezen keresztül bemutathatók a párhuzamos programozás fogalmai.

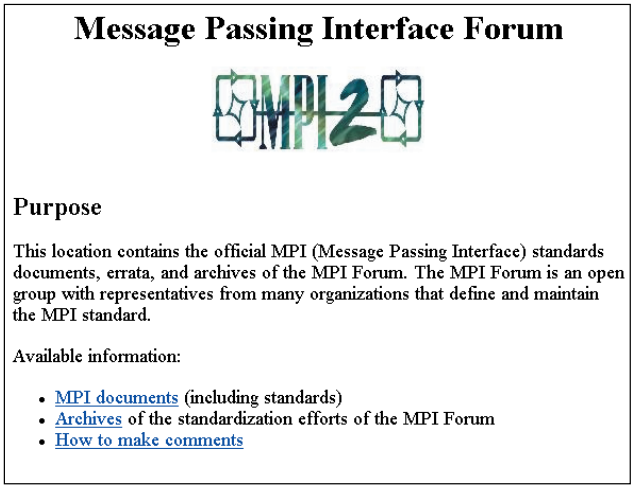
Soros megvalósítás

Az első változatnál úgy döntöttünk, hogy a számok tartományát egy olyan objektummal ábrázoljuk, amely képes az adott tartományban megszámolni a prímszámokat. (lásd az *1. listát*).
A program fordítása és futtatása a következőképpen történhet:

```
bash$ g++ -o primes primes.cpp
bash$ ./primes 0 10000
There were 1229 primes.
```

Többszálú megvalósítás

Ahhoz, hogy a többprocesszoros gépeken növelhessük a végrehajtási sebességet, szálakat kell használnunk. Szeretnénk ragaszkodni az előző változat szerkezetéhez, így meg kell találnunk a módját, hogy minden objektumnak saját végrehajtási szála legyen. A C++ és a pthreads keverése sajnos nem túl könnyű, mivel a `pthread_create( )` C és nem C++ szerkesztésű függvényt vár. Ezt úgy oldottuk meg, hogy készítettünk egy kiegészítő osztályt, illetve egy statikus tagfüggvényt (lásd a *2. listát*).
A `CountPrimes` objektum egyéb részei nem változtak. Három dolgot szükséges itt megjegyezni: 1. A Threaded osztály elvont osztály (abstract class). 2. Az `entry( )` statikus tagfüggvény, ami azt jelenti, hogy ismeri a Threaded osztályt, de nem kötődik annak egy adott példányához. Ezért aztán nem is megy keresztül a névkiértékelésen, így felhasználható C stílusú függvényként. Ez lesz az a függvénymutató, amit átadunk a `pthread_create( )`-nek a „szálasítandó” objektum mutatójával egyetemben. 3. A `run( )` tagfüggvény tisztán virtuális függvény, ezért minden, a Threaded osztályból származtatott osztályban készíteni kell belőle egy változatot. Ez a függvény lesz a származtatott osztály fő végrehajtási pontja, és visszatérési értéke fogja a számítás eredményét megadni. A CountPrimes osztály esetében a függvény egyszerűen kiszámolja és visszaadja a teljes tartományt.
A soros megoldásból szeretnénk minél többet megtartani, ezért csak egyetlen további paramétert (értéket) adunk meg, amely a feladat végrehajtásához felhasználható szálak számát határozza meg. Mivel előre nem tudhatjuk, hány objektumra (szálra) lesz szükségünk, dinamikusan foglalunk le számukra memóriát (lásd a *3. listát*).
E változatban található még néhány ravasz trükk, ezért a kódon egy kicsit részletesebben is végigmegyünk. Először kettőre állítjuk a szálak számának alapértelmezett értékét, majd megvizsgáljuk, hogy a felhasználó nem állított-e be valamilyen más értéket. Készítünk egy pthread_t nevű dinamikus tömböt, amely a szálak azonosítóit fogja tárolni, majd egy másik dinamikus tömböt a CountPrimes objektummutatók számára is. Ez nagyon fontos, mivel valamennyit különböző tartományban kell majd életre hívnunk („példányosítanunk”). Igazából a CountPrimes objektumokból nem is hozhatnánk létre statikus tömböt, hiszen nem határoztunk meg alapértelmezett létrehozó függvényt (konstruktort). Ez a felépítés rákényszerít bennünket, hogy helyesen használjuk az objektumot.



➲ http://www.mpi-forum.org/

Ezután egy ciklus következik, amely az egyes szálakat indítja, hogy ellenőrizzék a megfelelő számtartományokat. Figyeljük meg, hogy eddig még egyáltalán nem foglalkoztunk a terheléselosztással (load balancing). Erre a kérdésre még később visszatérünk. A legfontosabb mozdulat itt az, hogy minden CountPrimes objektum dinamikusan jön létre, mutatóik pedig a fent létrehozott tömbben tárolódnak.
A szálakat tulajdonképpen a `thread_create( )` függvény indítja el, melynek értékként a belépő tagfüggvény mutatóját, illetve az objektumra hivatkozó mutatót adjuk át. A létrehozott szál azonosítója a szálazonosító tömbben tárolódik.
Ezután a `pthread_join( )` és a szálazonosítók segítségével megvárjuk, amíg a szálak végeznek számítási feladatukkal. Mivel `run( )` függvény visszatérési értékének dinamikusan foglaltuk le a helyet, most fel kell szabadítanunk azt. Minden szál visszaadott értékét hozzáadjuk a számlálóhoz.
Végül megsemmisítjük a CountPrimes objektumokat, majd a szálazonosító tömböt és az objektummutató tömböt is. A programot a következőképpen lehet lefordítani és használni:

1. lista Prímek számítása

```
class CountPrimes {
public:
  CountPrimes(long start_, long stop_);
  long total();
private:
  long start;
  long stop;
  long count;
  bool counted;
  bool is_prime (long candidate);
};

CountPrimes::CountPrimes(long start_, long stop_)
  :start(start_),stop(stop_),count(0),
      ➥counted(false){
  if (start>=stop)
    throw range_error("Start >= Stop");
}

bool CountPrimes::is_prime (long candidate){
  // különleges esetben
  if (candidate<0) // negatív
    return false;
  if (!candidate) // == 0?
    return false;
  if (candidate==1) // az 1 nem számít prímnek
    return false;
  if (candidate==2)
    return true;
  // általános esetben
  for (long i=2; i<sqrt(candidate)+1; i++)
    // a jelölt maradék nélkül osztható i-vel?
    if (!(candidate%i))
      return false;
  // ha eddig eljutottunk, a szám prímszám
  return true;
}

long CountPrimes::total(){
  if (counted) // csak egyszer kell számolnunk
    return count;
  for (long i=start; i<=stop; i++)
    if (is_prime(i))
      count++;
  // most, hogy kiszámoltuk, állítsuk a jelzőt
  // igazra
  counted=true;
  return count;
}

// Ezután az objektumot magától értetődő módon
// használhatjuk a parancssorból megkapott
// értékekkel:

int main (int argc, char *argv[]){
  if (argc<3)
    usage(argv[0]);

  try {
    CountPrimes
      ➥counter(atol(argv[1]),atol(argv[2]));
    if (counter.total()>1){
      cout << "There were " << counter.total();
      cout << " primes." << endl;
    }
    else{
      cout << "There was " << counter.total();
      cout << " prime." << endl;
}
```

```
bash$ g++ -o primes_threaded
➥primes_threaded.cpp-lpthread
bash$ ./primes_threaded 0 10000 4
There were 1229 primes.
```

Megosztott megvalósítás

Az üzenetátadó felület (Message passing interface avagy MPI) a megosztott programok általános programozási felülete (API). Az MPI használatának sok előnye van, de a legnagyobb mégis az, hogy forrásszinten mindig megfelelő marad, függetlenül attól, hogy éppen milyen MPI-megvalósítást használunk. A példában mi a Notre Dame MPI-változatára (LAM – local area multicomputer, lásd a *Kapcsolódó címeket*) hivatkozunk és feltételezzük, hogy azt helyesen állították be.

A megosztott programozás egyik általános módja a mester-szolga modellre épül. Ebben a modellben az egyik folyamatot mesternek nevezik. Ez a folyamat osztja szét a munkát a szolgák között. A szolgák üzennek a mesternek, ha elkészültek, és új feladatot kérnek, amennyiben még van elvégzendő munka. Ez az egyszerű elvű modell nagyon jól működik, ha a feladat nem igényel komolyabb összehangolást és a szolgák teljesen függetlenek lehetnek. Ezt a feladattípust gyakran „kényszerpárhuzamosítottnak” is nevezik (embarrassingly parallel).

Az új program elkészítéséhez először el kell döntenünk, hogyan alakítsuk át a korábbi változatot a mester-szolga modellnek megfelelően, és be kell iktatnunk a szükséges MPI-hívásokat, hogy megosszuk a feladatot és begyűjtsük az eredményt. A *4. lista* bemutatja, milyen változtatások szükségesek a `main()` függvényben.

A megosztott program elején meg kell hívnunk az `MPI_Init( )` függvényt, hogy felvehessük a kapcsolatot a többszámítógépes rendszerrel (multicomputer). A következő két függvény a rangunkat és a számításban részt vevő számítógépek számát állapítja meg. Az MPI a rendszer valamennyi számítógépén ugyanazt a programot indítja el. Ezért szükséges futásidőben meghatároznunk a rangunkat, hogy eldönthessük, mesterek vagy szolgák vagyunk-e. A `master( )` vagy a `slave( )` függvényt rangunktól függően hívjuk meg. Miután végeztünk a számítással, meg kell hívnunk az `MPI_Finalize( )`-t, hogy elszakadjunk a rendszertől. A `slave( )` függvény mindössze egyetlen értéket vár, nevezetesen a felhasználandó szálak számát. Így a telepben található többprocesszoros (SMP) gépek teljes számítási kapacitását kihasználhatjuk.

A szolgák feladata, hogy munkára várakozva üldögéljenek, végrehajtsák a munkát és visszaadják az eredményt. Ezt kell folytatniuk mindaddig, amíg meg nem kapják a jelzést, hogy nincs több munka (lásd az *5. listát*).

A `slave( )` függvény kódjának nagy része hasonló az előző

2. lista Kiegészítő osztály és statikus tagfüggvény készítése

```
class Threaded {
public:
  Threaded() {};
  virtual ~Threaded() {};
  static void *entry (void *ptr);
  virtual void *run ()=0;
};

void *Threaded::entry(void *ptr){
  return reinterpret_cast<Threaded*>
       ➥(ptr)->run();
}

class CountPrimes : public Threaded {
public:
  CountPrimes(long start_, long stop_);
  long total();
  void *run();
private:
  long start;
  long stop;
  long count;
  bool counted;
  bool is_prime (long candidate);
};

void *CountPrimes::run(){
  long *return_val=new long(total());
  return reinterpret_cast<void*>(return_val);
}
```

„szálasított” változat `main( )` függvényéhez. Az egyetlen különbség az, ahogyan a szolga a számlálandó prímek határértékeit megkapja, illetve ahogyan az eredményt visszaadja.

A szolgák végtelen ciklusban várakoznak a mestertől érkező munkára, amit a `MPI_Recv()` függvényen keresztül kapnak meg. Ez a függvény a mester által küldött két hosszú egész számot (long integer, long) várja, melyek a feldolgozandó tartomány határértékeit adják meg. Miután a mestertől átvette az adatot, a szolga ellenőrzi az üzenet állapotát, hogy lássa, a munka véget ért-e már (KILL üzenet), amennyiben igen, befejezi a működését. Ha még van munka, a változókat átnevezzük, hogy pontosan az előző változatban lévő kódot használhassuk. Az egyetlen hátralévő lépés, hogy az eredményt visszaküldjük az `MPI_Send( )`-en keresztül. Itt egyetlen long értéket küldünk vissza, amelyben a szolga által talált számok darabszámát tároljuk.

A mester feladata valamivel összetettebb, mivel el kell döntenie, hogyan ossza fel a szolgáknak kiküldendő munkát és hogyan gyűjtse be az eredményt. Először kiküldi a munka egy részét a szolgáknak, majd megvárja, amíg az eredmény visszaér hozzá. Amikor a mester megkapja az eredményt, egy másik munkaegységet küld ugyanannak a folyamatnak, feltéve, hogy van még hátralévő munka. Ha nincs több, minden folyamatot még egyszer végigkérdez, nem maradt-e esetleg valamilyen eredmény, majd minden szolgának kiadja a kilépési utasítást (lásd a *6. listát*).

A `make_work( )` függvény dönti el, hogyan kell felosztani a munkát és mikor van annak vége. Egyszerű soros modellt választottunk, ahol a darabok méretét a `STEP_SIZE` határozza meg. (lásd a *7. listát*).

3. lista Szálak létrehozása

```
int main (int argc, char *argv[]){
  int num_threads(2);
  // helyesen hívtak meg?
  if (argc<3)
    usage(argv[0]);
  if (argc==4)
    num_threads=atol(argv[3]);

  try {
    pthread_t *t_id=new pthread_t[num_threads];
    CountPrimes **counter=
                 new CountPrimes*[num_threads];
    // indítsuk a szálakat
    long start(atol(argv[1]));
    long stop(atol(argv[2]));
    long incr((stop-start)/num_threads);
    for (int i=0; i<num_threads; i++){
      counter[i]=new CountPrimes(start,
        ➥(start+incr)>stop?stop:start+incr);
      start+=incr+1;
      pthread_create(&t_id[i],NULL,
                 counter[i]->entry,counter[i]);
    }
    // most várjuk meg az eredményt
    long count=0;
    for (int i=0; i<num_threads; i++){
      void *return_val;
      pthread_join(t_id[i],&return_val);

count+=*(reinterpret_cast<long*>(return_val));
      delete reinterpret_cast<long*>(return_val);
    }
    for (int i=0; i<num_threads; i++)
      delete counter[i];

    delete[] counter;
    delete[] t_id;

    if (count>1){
      cout << "There were " << count;
      cout << " primes." << endl;
    }
    else{
      cout << "There was " << count;
      cout << " prime." << endl;
    }
  }
  catch (range_error e){
    cout << "Exception: " << e.what() << endl;
    exit(EXIT_FAILURE);
  }
  return EXIT_SUCCESS;
}
```

Az egyes gépek közötti terheléskiegyenlítés kulcsa a `STEP_SIZE` változó. Ha az értéke túl nagy, egyes gépek üresjáratba kerülhetnek, míg mások túlságosan nagy tartományokkal bajlódnak. Ha viszont túl kicsi, akkor túl nagy lesz a kapcsolati terhelés. Az értéket általában a legegyszerűbb tapasztalati úton meghatározni. A részletekkel

5. lista Szolgák

```
void slave(int num_threads){
  long result;
  long bounds[2];
  MPI_Status status;

  while(true){
    MPI_Recv(bounds, 2, MPI_LONG, 0,
      ➥MPI_ANY_TAG, MPI_COMM_WORLD, &status);
    if (status.MPI_TAG == KILL)
      return;

    try {
      long start(bounds[0]);
      long stop(bounds[1]);

/*... ide jön a szálasított változat main()
      függvényének kódja ...*/

      MPI_Send(&count,1,MPI_LONG, 0,
                    0, MPI_COMM_WORLD);
    }
    catch (range_error e){
      cout << "Exception: " << e.what() << endl;
   }
  }
}
```

4. lista A main() változásai

```
int main (int argc, char *argv[]){
  int num_threads(2);
  int my_rank;
  int nprocs;

  // MPI előkészítése
  MPI_Init(&argc,&argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);
  MPI_Comm_size(MPI_COMM_WORLD, &nprocs);

  // helyesen hívtak meg?
  if (argc<3)
    usage(argv[0]);
  if (argc==4)
    num_threads=atol(argv[3]);
  long start(atol(argv[1]));
  long stop(atol(argv[2]));

  if (my_rank == 0)
    master(start,stop,nprocs);
  else
    slave(num_threads);

  MPI_Finalize();

  return EXIT_SUCCESS;
}
```


a Teljesítmény részben bővebben is foglalkozunk.
Az MPI programok fordításához az mpicc vagy az mpiCC használható, attól függően, hogy C vagy C++ kódot fordítunk-e. A megosztott program futtatásához először be kell indítani a többgépes rendszert a `lamboot`-on keresztül, majd a programot elindítani az `mpirun` paranccsal. Az MPI-munkamenet végén a rendszert a `wipe` paranccsal állíthatjuk le:

```
bash$ mpiCC -O -o primes_mpi
primes_mpi.cpp -lpthread
bash$ lamboot

LAM 6.3.2/MPI 2 C++/ROMIO
-University of Notre Dame

bash$ mpirun -O -np 16 primes_mpi
➥-- 0 10000000
There were 664579 primes.
bash$ wipe
```

Ha a `lamboot` futtatásával gondok akadnak, használjuk a `recon` parancsot annak kiderítésére, mi is okozza a hibát. Ha a `recon` sikertelen, valószínűleg nem futtathatunk programokat a távoli gépeken jelszó begépelése nélkül. Ha ssh-t használunk, bizonyosodjunk meg róla, hogy ennek jelzésére beállítottuk a `LAMRSH`-t:

```
bash$ export LAMRSH='which ssh'
```

Az mpicc kapcsolói lényegében ugyanazok, mint amelyeket egyébként a fordítónak szoktunk átadni. Az egyetlen kivétel mind a mpicc, mind az mpirun esetében a `-O`, amely azt mutatja, hogy a rendszer azonos típusú számítógépekből áll, így endian átalakításra nincs szükség.

A mpirun `-np` kapcsolója az elindítandó feladatok számát határozza meg (azaz általában a rendszer csomópontjainak számát). A két mínuszjel (`--`) utáni minden további érték átadódik a futó főprogramnak.

Teljesítmény

A párhuzamos programozás hatékonyságának bemutatásához meg kell mutatnunk, hogy az eltelt idő a párhuzamos változatban rövidebb. A csomópontonkénti százszázalékos teljesítménynövekedés általában nem érhető el, kivéve ha a feladatot a rendszer nagy darabokra bontja fel és komolyabb összehangolásra sincs szükség. Próbáinkat 16 két 700 MHz-es Pentium III-as processzort és 384 MB memóriát tartalmazó gépből álló telepen futtattuk. A programmal 0 és 10 000 000 között számítottuk ki a prímeket. Íme az egyes programok futási eredményei:

- Soros megvalósítás egyetlen csomóponton: 6:29,28 mp.
- Többszálú megvalósítás egyetlen csomóponton: 3:24,24 mp.
- Megosztott (és többszálú) megvalósítás 16 csomóponton: 11,05 másodperc.

Ezek az eredmények azt mutatják, hogy a processzorok száma egyenesen arányos a sebességgel (32x-es sebességgyorsulás a soros megvalósításhoz képest).

Terheléselosztás

A többgépes rendszerek programozásakor az egyik legnagyobb gond, hogy miként használhatjuk ki a lehető legjobban az összes számítógépet, illetve a többprocesszoros gépek összes processzorát. Nyilván el szeretnénk kerülni, hogy néhány gép üresjáratban várjon más számítások eredményére, melyeket egy másik gép vagy processzor végez. Ezt a finom művészetet nevezik terheléselosztásnak (load balancing).

6 lista Szolgák kiléptetése

```
void master(long start, long stop, long nprocs){
  long work[2];
  long result;
  long total(0);
  long current(0);
  MPI_Status status;

  // osszunk ki némi munkát
  for (int rank=1; rank<nprocs; ++rank){
    // határok beállítása ehhez a munkához
    if (make_work(work,&current,stop))
      MPI_Send(work,2,MPI_LONG,
               rank,WORK,MPI_COMM_WORLD);
  }
  // küldjünk még munkát, ha van
  while(make_work(work,&current,stop)){
    MPI_Recv(&result,1,MPI_LONG,MPI_ANY_SOURCE,
      ➥MPI_ANY_TAG,MPI_COMM_WORLD,&status);
    total+=result;
    MPI_Send(work,2,MPI_LONG,status.MPI_SOURCE,
      ➥WORK,MPI_COMM_WORLD);
  }
  // fogadjuk a különleges kérelmeket
  for (int rank=1; rank<nprocs; ++rank){
    MPI_Recv(&result,1,MPI_LONG,MPI_ANY_SOURCE,
      ➥MPI_ANY_TAG,MPI_COMM_WORLD,&status);
    total+=result;
  }
  // utasítsuk kilépésre a szolgákat
  for (int rank=1; rank<nprocs; ++rank){

MPI_Send(0,0,MPI_INT,rank,KILL,MPI_COMM_WORLD);
  }
  cout << "There were " << total;
  cout << " primes." << endl;
}
```

A terheléselosztás részletes ismertetése természetesen meghaladja cikkünk kereteit, de a megoldandó feladat néhány részletét azért megvizsgálhatjuk, hogy megtanuljuk, miképpen növelhetjük a teljesítményt. Példánkban a számítások nagy részét az `is_prime( )` függvény végzi. Természeténél fogva végrehajtási ideje nagyban függ a bemenő értékek számától. Figyeljük meg, hogyan osztottuk fel a munkát a második megoldásban, amikor csak két szálat használtunk: a számok felét az egyik, másik felét a másik szálnak adtuk át. Ez eredendően kiegyensúlyozatlan, hiszen a számokat sorban osztottuk el. A kisebb számokat kapó szál sokkal hamarabb fog végezni, mint a nagyobbakon dolgozó, így a processzor üresjáratba kerül. A gond legalább két módon megoldható: amikor felosztjuk a számtartományt, az egyes szálaknak felváltva küldjük a számokat; vagy egyszerűen még több szálat használunk, ami kisebb darabokra osztja fel a feladatot, és nagyobb mértékben bízza a terheléselosztást a rendszermag ütemezőjére. Természetesen ez csak addig működőképes, amíg az ütemezéssel töltött idő meg nem haladja a feladat felosztásával megtakarított időt.

A terhelés elosztása lényegesen hatékonyabb volt a megosztott megoldásban: minden gépnek csak kisebb munkaszakaszokat küldtünk, és csak akkor kaptak újabbat, ha az előzővel már végeztek. A kiküldött (a mi megoldásunkban a `STEP_SIZE` változó által meg-

7. lista. Soros megoldás

```
bool make_work(long *work, long *current,
       ➥long stop){
  if (*current>=stop)
    return false; // nincs több munka

  work[0]=*current;
  work[1]=*current+STEP_SIZE;
  *current=*current+STEP_SIZE+1;

  if (work[1]>stop)
    // ügyeljünk rá, nehogy átlépjük a határt
    work[1]=stop;
  return true;
}
```

határozott) darabok méretére azért oda kell figyelni, mivel jelentősen megnövelhetjük a hálózati forgalmat – tényleges sebességnövekedés nélkül. Hasonló megoldást alkalmazhattunk volna a szálak kiegyensúlyozására is, az átláthatóság kedvéért azonban nem ezt tettük.

Eric Bourque
(ericb@computer.org) PhD hallgató Kanadában, a montréal-i McGill Egyetemen. A számítógépes grafika és a képalapú mintázatrajzolás problémáival foglalkozik. Rendelkezik egy szakirányú (MSc) és egy zenei (szaxofon BMus) diplomával is.

Kapcsolódó címek

Jó angol nyelvű C++ könyvek gyűjteménye
➲ http://www.telegraph-road.org/books.html

Bjarne Stroustrup, a C++ megalkotójának könyve, a C++ programozási nyelv magyar változatának honlapja
➲ http://www.zolix.hu/konyv_hu.html

Linux Paralell Processing HOWTO
➲ http://www.beta.ttt.bme.hu/HOWTO/Paralell-Processing-HOWTO.html
➲ http://www.linuxdoc.org/HOWTO/Paralell-Processing-HOWTO.html

C++ FAQ
➲ http://www.parashift.com/c++-faq-lite/

comp.programming.threads FAQ
➲ http://www.LambdaCS.com/newsgroup/FAQ.html

MPI Forum
➲ http://www.mpi-forum.org/

LAM
➲ http://www.mpi.nd.edu/lam

LAM FAQ
➲ http://www.mpi.nd.edu/lam/faq/

A cikk teljes forráskódja megtalálható a következő címen:
➲ ftp://ftp.cim.mcgill.ca/pub/people/ericb/primes.tar.gz

A PostgreSQL és a PHP (2. rész)

Varázsoljuk PostgreSQL-lel tárolt adatbázisainkat a Webre, és egyúttal kóstoljunk bele napjaink egyik leggyorsabban fejlődő programozási nyelvébe, a PHP-ba.

Sorozatunk előző részében telepítettük a PHP-t és beállítottuk a PostgreSQL-t, hogy PHP-ból elérhessük az adatbázisokat. Most megismerkedünk a PostgreSQL biztonsági beállításaival, a PHP postgreses adatbázisokat támogató függvényeivel, írunk egy rövid webes alkalmazást, és a cikk végén megnézzük, hogy miként kell eljárni, ha a PostgreSQL egy újabb változatára szeretnénk átállni.

Az előző cikk végén lévő példában (test.php3) kapcsolódni próbáltunk a php_db adatbázishoz a php_user felhasználó nevében, a heureka jelszóval:

```
<?php
$connection = pg_Connect (
  "dbname=php_db port=5432 user=php_user
  password=heureka");
?>
```

A PostgreSQL biztonsági beállításai

Mielőtt tovább ismerkednénk a PHP-val, egy kis kitérőt kell tennünk a PostgreSQL felhasználóinak jelszavai kapcsán. Próbáljunk rossz jelszót megadni a test.php3 fájlban vagy változtassuk meg a php_user jelszavát psql-ben az `ALTER USER php_user WITH PASSWORD kukucs;` paranccsal. Elképzelhető, hogy a PostgreSQL így is engedi a hozzáférést az adatbázishoz. Ha ezt nem akarjuk, módosítsuk a `/etc/postgresql/pg_hba.conf` fájlt. Itt eltérő biztonsági szinteket (felhasználó-ellenőrzési módszereket) állíthatunk be a különböző helyekről érkező kérésekhez. Ha egy helyben megbízunk (`trust` értéket állítunk be) – ez az alapértelmezett biztonsági szint helyi hálózatunkra, – nem történik semmilyen jelszóellenőrzés. Ha `password` a beállított érték, a PostgreSQL ellenőrzi a jelszót, de a jelszavak sima szövegként haladnak a hálózaton. Ha `crypt`-et adunk meg, a jelszavak titkosítva utaznak a hálózaton, de vigyázzunk, a `/var/lib/postgres/data/pg_pwd` fájlban (*10. lista*) a jelszavak kódolatlanul tárolódnak. Ilyenkor a `pg_pwd`-ben levő jelszót kódolja a rendszer, majd ezt a kódolt jelszót hasonlítja össze a belépni kívánó felhasználó által megadott jelszó kódolt változatával. A rendszer további hiányosságairól bővebben a `postgresql-doc/README.passwords` fájlban olvashatunk.

Ehhez kapcsolódik még néhány fontos dolog: ha nem `trust`-ot állítottunk be a helyi hálózatunkra, a psql-t `psql -u` paranccsal kell indítanunk, különben nem kér jelszót, így azután nem férhetünk hozzá az adatbázisokhoz. Ekkor azok a felhasználók, akiknek nem állítottunk be jelszót, többé nem tudnak bejelentkezni. Ilyen a Postgres is, tehát mindig állítsunk be jelszót a felhasználónak. Van egy *do.maintenance* karbantartó parancsfájl, amit a cron démon futtat a Postgres-felhasználó nevében (Debianon a /etc/cron.d/postgres fájl segítségével), ezt kell kibővíteni a `-u postgres -p jelszó` szöveggel.

A másik említésre méltó dolog, hogy azok a felhasználóink, akik maguk is létrehozhatnak felhasználókat, teljes jogú urai a PostgreSQL-nek, azaz minden felhasználó jelszavát (így a Postgresét is) megváltoztathatják, sőt, megnézhetik a *pg_pwd* fájlban (10. lista).

Adatelérés PHP3-mal

Most már minden készen áll ahhoz, hogy elérjük a PostgreSQL adattáblákat, és megjelenítsük a tárolt adatokat a weboldalunkon. Ahhoz, hogy ezt kipróbáljuk, hozzunk létre táblákat és engedélyezzük egy felhasználónak az elérést! Hozzuk létre az allatok táblát – amelyben az állatok és tulajdonosaik neve található – a

```
create table allatok (allatnev char(8), tulaj
       char(8));
```

utasítással, majd tegyünk bele egy sort az

```
insert into allatok values ('Méhecske','Szilvia');
```

paranccsal. Ezután adjunk jogot a php_user felhasználónak a táblához:

```
grant select on allatok to php_user;
```

(A lemezmellékletben szereplő `php_proba.sql` fájlban megtalálhatók ezek az utasítások, elég a psql-ben kiadni a `\i php_proba.sql` utasítást.) A grant utasítás általános alakja `GRANT engedélyek ON táblák TO felhasználó;` ahol az engedélyek ALL, SELECT, INSERT, UPDATE, DELETE, illetve RULE. Ezek a szavak megfelelnek az azonos nevű SQL utasítás végrehajtásához való joggal, kivéve az ALL-t – ami minden jogot megad, és a RULE-t – amely a szabályok létrehozását engedi, és nem szabványos SQL utasítások. A felhasználó lehet PUBLIC, ez az összes felhasználót takarja, vagy egy csoportnév, amely a felhasználók egy csoportjára utal. Erről bővebb tájékoztatást a PostgreSQL leírásban találhatunk.

Egy adatbázis tábláira adott jogosultságokat a `\z psql` parancs kiadásával nézhetünk meg (*11. lista*). A jobb oldali oszlop egy sorában levő értéksorozat jelzi, kinek milyen jogokat adtunk felhasználó/csoport=jogok formában. A jogokat karaktersorozat jelképezi, ahol r:SELECT, w:UPDATE/DELETE, a:INSERT, r:RULE lehet. Ha az egyenlőségjel bal oldalán nincs semmi, akkor a sor mindenkire érvényes jogokat mutat.

Miután elkészült a tábla, tovább próbálgathatjuk a PHP3-at. A test2.php3 (*12. lista*) program kapcsolódik az adatbázishoz. A `pg_Connect()` függvény eredménye egy szám, amely azonosítja a kapcsolatot. Több adatbázishoz is kapcsolódhatunk, ekkor mindegyik kapcsolatnak más-más száma lesz. A további PHP-PostgreSQL függvényekben ezt a kapcsolatazonosítót kell használnunk.

A `pg_Exec (kapcsolat_azonosító, SQL-szöveg);` függvény az SQL-szövegben levő SQL utasítást hajtatja végre. A visszaadott szám azonosítja a lekérdezést. Hiba esetén 0, eredményt nem szolgáltató SQL utasítások (INSERT, DELETE, UPDATE stb.) esetén

10. lista A /var/lib/postgres/data/pg_pwd fájlban normál szövegként tárolódnak a jelszavak

```
billgates     1002  t   t   t   t   \N   \N
proba         1003  t   t   t   t   \N   \N
phpuser       1004  f   f   f   f  heureka \N
jeno          1006  f   f   f   f   \N   \N
php_user      1005  f   f   f   f  heureka \N");
```

11. lista A \z psql utasítással a táblákhoz rendelt jogosultságokat nézhetjük meg

```
php_db=>    \ z
Database      =    php_db
+————————+————————————————————————+
| Relation |  Grant/Revoke Permissions  |
+————————+————————————————————————+
| allatok  |     !"=","php_user=r"}     |
| szemely  |     {"="}                  |
+————————+————————————————————————+
```

12. lista A test2.php3 program. Kiírja egy SQL tábla első sorának első mezőjét

```
<?php
 $connection = pg_Connect (
      "dbname=php_db port=5432 user=php_user
       ➥password=heureka");

 $result = pg_Exec($connection,
                   "SELECT * FROM allatok;");
 $row = pg_fetch_row ($result, 0);
 print "A tábla elso sorának elso eleme: <B>"
       . $row[0] . "</B>\n";
 pg_Close($connection);
?>
```

1, lekérdezéseknél egynél nagyobb ez az érték. A későbbiekben, ha egy lekérdezés eredményét el akarjuk érni, ezt az azonosítót kell használnunk.

A `pg_fetch_row (lekérdezés_azonosító, sorszám);` függvény az eredmény *sorszám*-adik sorát adja vissza tömbként. A tömb elemei a sor mezői. Az első eredménysor a 0-ik sorszámú. A `print` paranccsal vastagon szedve kiírjuk a `$row` tömb nulladik elemét. Végül bezárjuk a kapcsolatot a `pg_Close (kapcsolat_azonosító);` függvénnyel.

A test3.php3 program (*13. lista*, a sorszámok csak a magyarázat kedvéért szerepelnek) végigolvassa a táblát és HTML táblázatként kiírja. A PHP-program a 3. sornál kezdődik. A kapcsolat létesítése és az SQL utasítás kiadása után (4–6. sorok) kiírjuk a táblázat fejlécét, piros hátterű cellákban. A `pg_NumFields (lekérdezés_azonosító)` függvény megadja, hány oszlopa van az eredménytáblázatnak. A 9. sorban indított számlálásos ciklussal végigmegyünk az oszlopokon és a `pg_fieldname (lekérdezés_azonosító, oszlop_sorszám)` függvénnyel – ez az oszlopok nevét adja vissza – a 10–11. sorokban kiírjuk a fejlécet. A `pg_numrows (lekérdezés_azonosító);` függvény

13. lista A test3.php3 program. Egy teljes eredménytábla kiírása

```
1 A tábla sorai:<P>
 2 <TABLE BORDER=1 BGCOLOR=YELLOW>
 3 <?php
 4  $connection = pg_Connect
 5    ("dbname=php_db port=5432 user=php_user
       ➥password=heureka");
 6  $result = pg_Exec($connection,
       "SELECT * FROM allatok;");
 7
 8  print '<TR>';
 9  for($i=0; $i<pg_numfields($result);$i++) {
10    print '<TD BGCOLOR=RED><B>'.
11       pg_fieldname($result,$i)."</B></TD>";
12  };
13  print '</TR>';
14
15  for($i=0; $i<pg_numrows($result); $i++) {
16    $row = pg_fetch_row ($result, $i);
17    print '<TR><TD>'
   ➥.implode('</TD><TD>',$row)."</TD></TR>\n";
18  };
19
20  pg_Close($connection);
21 ?>
22 </TABLE>
```

14. lista A test4.php3 program. Az include() parancs bemutatása

```
<HTML>
<HEAD>
<TITLE>test4</TITLE>
</HEAD>
<BODY bgcolor=black text=white>
<?php
 include("test4.inc");
 print "<H1><CENTER>Helló</CENTER></H1>\n";
 include("test4.inc");
?>
</BODY>
</HTML>
```

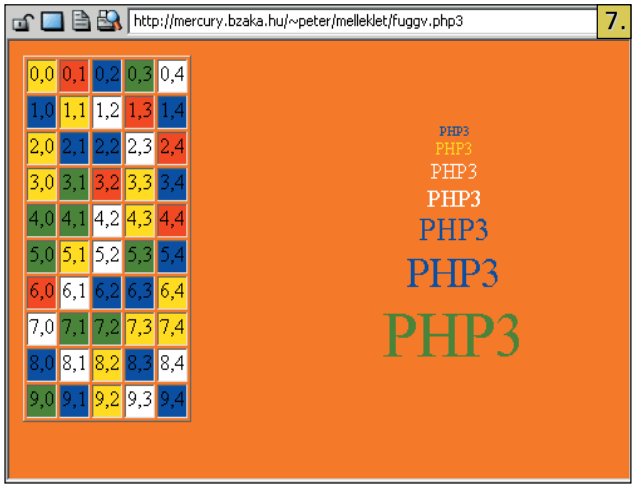
15. lista A test4.inc fájl. Példa include-fájlra

```
<TABLE bgcolor=red width=100%>
<TR><TD><CENTER>Szia</CENTER></TD></TR>
</TABLE>
```

visszaadja az eredménytáblázat sorainak a számát. Ennek segítségével (15. sor) egy számlálásos ciklussal olvassuk be a sorokat a $row tömbbe (16. sor), ezt az `implode()` függvénnyel alakítjuk át karakterláncca és írjuk ki (17. sor).

16. lista A fuggv.php3 program

```
<?php
include("fuggv.inc");
?>
<HTML><HEAD></HEAD>
<BODY BGCOLOR=ORANGE>
<TABLE BORDER=0 WIDTH=100%>
 <TR>
  <TD WIDTH=50%>
    <TABLE BORDER=1>
       <?php MyTable(); ?>
    </TABLE>
  </TD>

  <TD>
    <CENTER>
    <?php PHP3_Caption(1,7); ?>
    </CENTER>
  </TD>
 <TR>
</TABLE>
</BODY>
</HTML>
```



Hasznos PHP-trükkök

Include-fájlok, függvények, űrlapok

Mielőtt megírnánk egy „komolyabb” alkalmazást, meg kell ismerkednünk a nyelv néhány fontos tulajdonságával, ezért átmenetileg szakadjunk el az adatbázisoktól.

Elsőként megemlíteném, hogy PHP-programunkat lehetőségünk van több fájlban tárolni. Így a különálló részeket egyszerűen használ-

17. lista A fuggv.inc fájl. Példa függvényekre

```
<?php

srand((double)microtime()*1000000);

function random_color() {
  $colors=array('white','red','green',
                'yellow','blue');
  return $colors[rand(0,count($colors)-1)];
};

function MyTable() {
  for($i=0; $i<10; $i++) {
    print '<TR>';
    for($j=0; $j<5; $j++) {
       print '<TD BGCOLOR='
             ➥.random_color().">$i,$j</TD>";
    };
    print "</TR>\n";
  };
};

function PHP3_Caption($minsize,$maxsize) {
   for($i=$minsize; $i<=$maxsize; $i++) {
     print "<BR><FONT SIZE=$i COLOR="
           ➥.random_color().
            ">PHP3</FONT>\n";
   };
};
?>
```

hatjuk későbbi programjainkban, és kisebb mérete következtében a forrás is olvashatóbb lesz. A részeket beilleszteni az `include (fájlnév);` utasítással lehet. Erre példa a test4.php3 (*14. lista*) program és a test4.inc (*15. lista*) fájl. A test4.inc fájl piros hátterű táblázatot rajzol egyetlen cellával, melynek közepén a *Szia* szó szerepel. Ezt a fájlt fűzzük be kétszer test4.php programunk szövegébe, a *Helló* szó kiírása előtt és után (*6. kép*).

Az include-fájlok nevének nem kötelező .INC-re végződni, az bármi lehet, akár .PHP is. A php-végződés előnye, hogy az Internetről láthatatlan a tartalma, ugyanis amikor valaki megpróbálja letölteni, a webkiszolgáló értelmezi a fájlt, és csak az abban lévő kimenet tartalma fog megjelenni a böngészőben. Azért használok mégis .INC végződést, hogy könnyen megkülönböztethetők legyenek az include-fájlok a programfájloktól.

Függvények írása sem nehéz PHP-ban. A fuggv.php3 program (*16. lista*) meghívja a `MyTable ()` és a `PHP3_Caption ()` függvényeket (a program eredménye a *7. képen* látható). Mindkét függvény megvalósítása a fuggv.inc fájlban (*17. lista*) található. Az `srand ()` paranccsal beállítjuk a véletlenszám-előállító kezdőértékét. A `function random_color ()` részben egy véletlen színt választó függvény van, ami a return utasítással ad vissza értéket a hívó programnak. A `function MyTable ()` kezdetű blokkban két számlálásos ciklussal írjuk ki a 7. kép bal oldalán látható táblázatot. Itt látunk példát a `random_color ()` függvény használatára is. A `PHP3_Caption ($minsize,$maxsize)` függvény készíti el a 7. kép jobb oldalát. A két átadott érték a PHP3 szöveg méretének alsó és felső korlátja. A PHP-nyelv megengedi a változók cím és érték szerinti használatát, alapértelmezett (default) értékének megadását és változó számú átadott érték kezelését is.

18. lista A form.html fájl egy egyszerű űrlapot valósít meg

```
<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>
<BODY>

<FORM action="mutat.php3" method="post">

<P>Írj be valamit:
<INPUT type="text" name="szoveg1">
<INPUT type="text" name="szoveg2">

<P><INPUT type="submit" value="Elküldés">
<INPUT type="reset" value="Törlés">

</FORM>
</BODY>
</HTML>
```

19. lista A mutat.php3 program.
Űrlapmezők neveinek és értékeinek kiírása

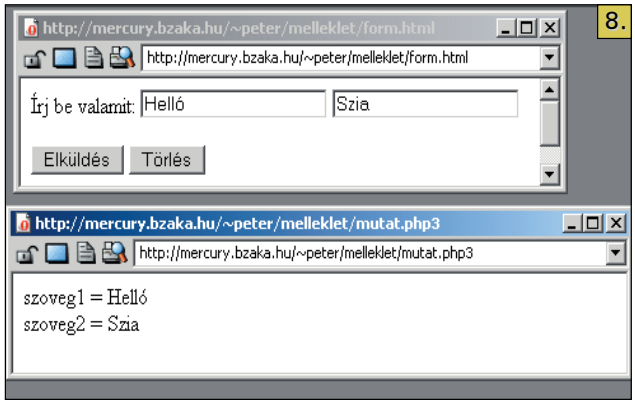
```
<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>
<BODY>
<?php
 reset ($HTTP_POST_VARS);
 while (list($key, $val)=
each($HTTP_POST_VARS)) {
    print "$key = $val<BR>\n";
 };
?>
</BODY>
</HTML>
```

Erről bővebben a PHP-leírás *functions.arguments.html* oldalán olvashatunk.

Szükségünk lesz űrlapok kezelésére is. Legyen az űrlapunk a form.html fájl (*18. lista*). Ebben két beviteli mező van, szoveg1 és szoveg2 néven, valamint elküldő- és törlőgomb. Elküldéskor a mutat.php3 program kapja meg a beviteli mezők neveit és értékeit. PHP3-ban a POST eljárással kapott név–érték párokat a `$HTTP-POST-VARS` tömbön keresztül érhetjük el; ha GET-et használtunk, akkor a `$HTTP-GET-VARS` tömb áll a rendelkezésünkre. Ezek úgynevezett társított vagy nevesített (asszociatív) tömbök, ezek lényege, hogy a tömb első oszlopa változók neveit, a második pedig az adott változóhoz tartozó értékeket tartalmazza. A tömb egy értékét a hozzá tartozó névvel is indexelhetjük. A tömb adatait emellett elérhetjük a 0, 1, … indexek használatával is, ilyenkor a páros indexekre a neveket kapjuk, a páratlanokra az értéket. Ha például a szoveg1 beviteli mezőbe a *Hello* szót írtuk, elküldés után a mutat.php3 programban a `$HTTP_POST_VARS ["szoveg1"]` kifejezéssel kaphatjuk meg a Hello értéket. A tömböknek van belső mutatójuk is, ennek segítségével az elemeket indexelés nélkül, sorban egymás után is elérhetjük.

20. lista A valaszt.php3 program. Választólistás űrlap

```
 1 <?php
 2  $connection = pg_Connect(
 3       "dbname=php_db port=5432 user=php_user
        ➥password=heureka");
 4  $result = pg_Exec($connection,
              "SELECT * FROM allatok;");
 5  pg_Close($connection);
 6
 7  print '<FORM action="egy.php3"
        ➥method="post">'."\n";
 8
 9  print '<SELECT size="6"
        ➥name="allatok">'."\n";
10  for($row=0; $row < pg_NumRows($result);
              $row++){
11    print '<OPTION VALUE='.pg_Result
        ➥($result, $row, 1).'>';
12    print pg_Result ($result, $row, 0) .
        ➥' gazdája ' .
13          pg_Result ($result, $row, 1) .
                  ➥'</OPTION>' . "\n";
14  }
15  print '</SELECT>'."\n";
16
17  print '<P><INPUT type="submit"
        ➥value="Ezt választom">'.
18        '<INPUT type="reset"
        ➥value="Választás törlése"></FORM>';
19
20 ?>
```

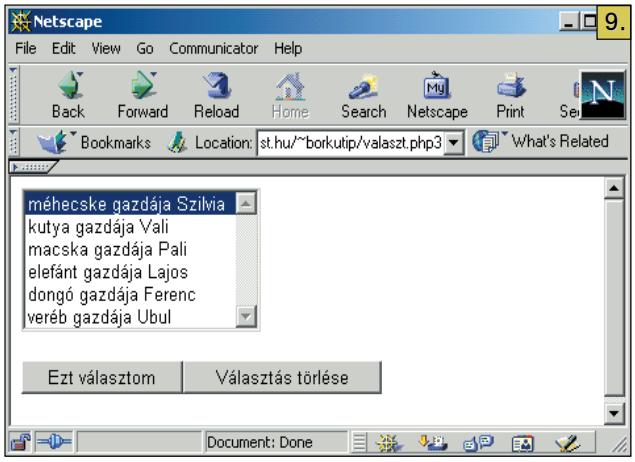


A PHP beállításától függően az űrlap beviteli mezői elérhetők lehetnek egyszerű változóként is, azaz a szoveg1 beviteli mező értéke lekérdezhető a `$szoveg1` globális változón keresztül. Sokan nem szeretik ezt a szolgáltatást, ugyanis könnyű összekeverni a program által egyébként is használt globális változókat az űrlapokon keresztül kapottakkal. Erről bővebben a leírás *configuration.html* oldalán olvashatunk a track_vars és a register_globals részeknél.

Nézzük meg a mutat.php3 programot (*19. lista*). A `reset ($HTTP_POST_VARS);` sorral a tömb belső mutatóját a tömb elejére állítjuk. A `while (list($key, $val) = each ($HTTP_POST_VARS))` kezdetű blokk egy elöl tesztelős ciklus, amelynek magjában kiírjuk a `$key` és a `$val` változók értékeit. A `$key` tartalmazza az űrlapmező nevét, a `$val` pedig az értékét.

21. lista Az egy.php3 program egyetlen értéket dolgoz fel

```
<?php
 $val = $HTTP_POST_VARS["allatok"];

 $connection = pg_Connect("dbname=php_db
  ➥port=5432 user=php_user password=heureka");
 $result = pg_Exec($connection,
    "SELECT * FROM szemely
          ➥WHERE nev='$val';");
 pg_Close($connection);

 print pg_Result ($result, 0, 0).','.
       pg_Result ($result, 0, 1).','.
       ((pg_Result ($result, 0, 2)=='f') ?
            'nő' : 'férfi') ."\n";
?>
```



Az az izgalmas rész, ahol a `$key` és `$val` megkapja értékét, ez pedig a ciklusfeltételben történik meg. A `list (változólista)` kulcsszó hatására egyben kezelhetünk változókat, úgy, mintha egy tömb lennének. Az `each (tömb)` kiválasztja a tömbből a belső mutatója által mutatott név–érték párt és hátrébb állítja a mutatót. Így a `list ($key, $val) = each ($HTTP_POST_VARS)` értékadásra a `$key` és `$val` változók megkapják az általunk kívánt értékeket. Az elöl tesztelős ciklus miatt az űrlap mezőjét kiírjuk, nem kell tudnunk, hogy pontosan mi volt a mezők neve és hány volt az űrlapon. Futtatásának eredményét láthatjuk a *8. képen*.

Egy alkalmazás

Most már visszatérhetünk az adatbázis-kezeléshez. Legyen a feladatunk az, hogy a példaadatbázis `allatok` táblájában levő állatok gazdáiról tudjunk meg adatokat. A programunk felkínálja az összes bejegyzést a táblából. A felhasználó válasszon innen egy sort, és az ott levő gazdához tartozó adatok jelenjenek meg egy újabb weboldalon.
Elsőként nézzük meg a valaszt.php3 program eredményét (*9. kép*).
A böngészőben egy lista jelenik meg az állatokról és gazdáikról. A képernyő alján két gomb található az elküldésre és a választás törlésére.
A program (*20. lista*) sorait megszámoztam. Tekintsük át a parancsokat! A lista első része (1–5. sorok) csatlakozik az adatbázishoz és lekéri az összes sorát. A 7. sor eredménye lesz a weboldal első sora, megnyit egy űrlapot, amely az adatokat az egy.php3 programnak küldi el. A lényeges rész a 9–15. sorokban van. Itt állítja elő a program a választólistát. A 10. sorban végigmegyünk az eredménytábla összes során és mindegyik sorból készítünk egy HTML sort:

```
<OPTION VALUE=gazda neve>állat gazdája gazda
neve</OPTION>
```

Így az egy.php3 programnak valamelyik állattartó nevét fogjuk elküldeni. A 17. és 18. sorok egy submit és egy reset gombot állítanak elő. Az egy.php3 program (*21. lista*) a $val változóba teszi az *allatok* névhez tartozó értéket a `$HTTP_POST_VARS` tömbből. Ezután kiválasztja azokat a sorokat a *szemely* táblából, ahol a *nev* értéke megegyezik a `$val` változó értékével. Azért kell `$val`-t aposztrófok közé tenni, mert a tömb *nev* oszlopa szöveges típusú adatokat tartalmaz. A program utolsó része kiírja az eredménytábla nulladik sorát. A `(pg_Result ($result, 0, 2) == 'f') ? 'nő' : 'férfi')` eredménye 'nő' vagy 'férfi' – attól függően, hogy a harmadik mező értéke 'f' lesz-e. A program kimenete egyetlen sor a böngészőben, például: Szilvia ,1978-03-14, nő.

Áttérés új PostgreSQL-változatra

Nagyobb váltásoknál módosul az adatbázisok tárolási formátuma. Mivel az újabb változatok nem mindig működnek a régi adatbázisokkal, ki kell menteni az adatbázisokat, letörölni az adatbázis-könyvtárat (általában a `/var/postgres/data` könyvtár), és meg kell szabadulni a régi Postgrestől, telepíteni kell az újat, majd beüzemelni és visszaállítani az adatbázisokat. Ezalatt természetesen meg kell akadályoznunk, hogy az adatbázisok változzanak. Legegyszerűbb, ha leállítjuk a PostgreSQL-t. (A folyamatról olvashatunk a /usr/doc/postgresql-doc/README.Debian.migration.gz fájlban.)
Frissítsünk mi is a 6.3.2-ről (ami a Debian Slink-ben található) a 6.5.3-ra (ez a Potato-változat része).

Az adatbázisok mentése és visszatöltése

Állítsuk le a Postgrest a `/etc/init.d/postgres stop` parancscsal. Rendszergazdaként át tudunk váltani a postgres felhasználóra a `su - postgres` utasítással. Ezután adjuk ki a `postgresql-dump -t db.out` parancsot. A db.out fájl a /var/postgres könyvtárban jön létre. Belenézhetünk: átlagos szöveges fájl, amely a psql számára készült. Másoljuk biztonságos helyre, például a saját könyvtárunkba. Éles rendszereken természetesen mentsünk le mindent a következő lépések végrehajtása előtt!
Töröljük a `/var/postgres/data` könyvtárat.
Töröljük a postgresql csomagot: `dpkg --remove postgresql`
Telepítsük az új PostgreSQL-t. Telepítés alatt beállíthatjuk a használt dátumformátumot és karakterkódolást. Ha nem történt hiba, létrejön a `/var/postgres/data` könyvtár, benne a rendszeradatbázisok és a *template1* adatbázis.
Hozzuk létre régi adatbázisainkat. Lépjünk be postgres néven: `su - postgres`, és adjuk ki a `psql -e elérési_út/db.out` parancsot. Ha valami hibát veszünk észre, elképzelhető, hogy javítani kell a db.out fájlon. Ekkor kezdjük újra a folyamatot: állítsuk le a Postgrest, töröljük a data könyvtárat, lépjünk be postgresként, majd adjuk ki az `initdb` parancsot, ez létrehozza a data könyvtárat a szükséges adatbázisokkal.
A listák megtalálhatók a 14. CD-melléklet Magazin/Cikkekhez könyvtárában.

Borkuti Péter (borkuti@freemail.hu),
matematika-informatika szakos tanár, rendszergazda, informatikus, rendszerépítő és programozó.

JavaServer Pages

Kiszolgálóoldali fejlesztői és futtató környezet kialakítása Linux-rendszeren.

A ma készülő alkalmazások jelentős része a többrétegű modell elve alapján készül. Ez a tervezési mód annyira előretört, hogy mostanában még a régi alkalmazások is e modell kapcsán kapják meg az „egyrétegű”, illetve a „vastag ügyfél” nevet. Természetesen ebben a felfogásban nincs semmi új, hiszen a nagyobb programokat mindig is úgy tervezték, hogy azok függőleges és vízszintes tagolású programegységekből álljanak. Erre klasszikus példa a TCP/IP protokoll, ahol a legismertebb megvalósításnál függőlegesen négy réteg van, ugyanakkor például az IP-réteg is több együttműködő részből áll (ICMP, IP stb.). Itt a *réteg* szó arra is utal, hogy az egymással kapcsolatban álló programelemek hány különálló futáskörnyezetre bonthatók. Amennyiben alaposan megvizsgáljuk a Unix-, vagy a Windows-rendszereket, akkor belátható, hogy ez az elgondolás már nagyon régi (lásd a `.so` és `.dll` fájlokat mint kiszolgálókat). Egy háromrétegű alkalmazás jellemző kialakítása a következő: adatbáziskezelő (AB) réteg – üzleti logika (ÜL avagy lényegi rész) réteg – felhasználói felület (FF avagy megjelenítési) réteg. Itt az adatbázisréteg és az üzleti logika, valamint az üzleti logika és a felhasználói felület kapcsolata elképzelhető egy-egy kiszolgáló-ügyfél kapcsolatként is.

Az üzleti logika nagyjából állandó részét megvalósító alapprogramot alkalmazáskiszolgálónak nevezzük, ennek számos esetben a legfontosabb része egy webkiszolgáló (HTTP-kiszolgáló), bár lényeges kiemelni, hogy a CORBA, DCOM, RMI és más módszerek önmagukban is hatékony üzleti logika kialakítását teszik lehetővé. Alkalmazásunk szolgáltatásait mégis érdemes lehet egy webkiszolgáló köré építeni, hiszen a HTTP protokoll további lehetőséget is biztosít számunkra. Miért? A válasz a programozáselmélet típusfogalmával való összehasonlítással érthető meg.

A TCP szállítási réteg, illetve annak Socket vagy TLI (Transport Layer Interface) API-ja csak egy típusnélküli, nyers bájtfolyamot bocsát a rendelkezésünkre, ami még általában alacsony szintű eszköz azok számára, akik például internetes áruházat szeretnének kialakítani. Ezzel szemben a HTTP és a hozzá kapcsolódó társprotokollok és eszközök (például: a MIME) típusos fogalmak, hiszen a rétegek között teljes objektumok (applet, kép, XML stb.) is közlekedhetnek, amik ismerik saját belső ábrázolásaikat, állapotaikat, típusműveleteiket (metódusok). A HTTP kiszolgáló önmagában nem jelent teljes megoldást, viszont jól együttműködhet más programokkal (CORBA, RMI, Servlet, biztonság stb.). Ezt a felépítést nevezzük alkalmazáskiszolgálónak.

A vékony felhasználói réteg gyakran egy HTML-, XML-, illetve Java-alapú (esetleg ActiveX-alapú) vékony ügyfélalkalmazás lehet. Az adatbázisréteg pedig már mindenki számára ismert, hiszen az Oracle, DB/2 stb. programok jól érzékeltetik, hogy miket is kell e rétegben megvalósítani, illetve milyenek azok a hálózati protokollok (pl.: Net *8), amelyek az adatbázis-szolgáltatások hálózaton keresztüli elérését teszik lehetővé.

A webes alkalmazások fejlődéstörténete

A Microsoft Visual InterDev vagy a SUN JSP-vel foglalkozó írásai szerint – a webes alkalmazások fejlődését nyomon követve – három nagy korszakot különíthetünk el:

- *Az első nemzedékbeli webkiszolgálók korszaka.* Ezen megoldásokra a statikus HTML-oldalak és az azokba beágyazott grafikák (*.jpg, *.gif), hangok, később a mozgófilmek voltak a jellemzők. Ezt a világot egészítette ki a böngészőkben bővítményként megjelenő VRML-világ, valamint a böngészőoldali parancsfájlok és a Java-appletek lehetősége. Később az MS kidolgozta az appletek vetélytársaként is feltüntetett ActiveX-módszert.
- *A második nemzedékbeli webes alkalmazások* újdonsága a HTML-oldalak tartalmának dinamikus kialakítása volt. A Hálón szörföző emberek egyre unalmasabbnak és haszontalanabbnak tartották azokat a honlapokat, amelyek könyv módjára, teljes mértékben előre rögzített módon (statikusan) készültek. E módszerek legfőbb képviselői a CGI-programok, az SSI-k (Server Side Include). Érdekes kezdeményezés volt, hogy a webkiszolgálókat – hasonlóan más kiszolgálókhoz – API-val lássák el. Ennek legfőbb hátrányát az képezte, hogy minden webkiszolgáló más-más API-val rendelkezhetett. A CGI-programokat mai szemmel vizsgálva néhány hátrányuk azonnal szembetűnik: nehezen lehet bennük a HTTP-protokoll állapotfüggetlen jellegéből eredő korlátokat kiküszöbölni (például folyamat- vagy viszonykezelés), a CGI-parancsállományok esetenként lassúak és sok erőforrást igényelnek, mert minden CGI-kérésnél egy teljes értékű folyamat indul el. A pontosság kedvéért azonban azt is fontos megemlíteni, hogy például a Perl vagy a PHP-eszközök és az őket futtató környezet igyekszik mindent megtenni azért, hogy a fenti hátrányokat a lehető legjobban mérsékeljék.
- *A harmadik nemzedékbeli kiszolgálóalkalmazások* akkor születtek meg, amikor a fejlesztőeszközöket készítő cégek rájöttek arra, hogy az internetes (és belső hálózati) helyeket is érdemes lenne a hagyományos alkalmazáskészítő szemlélet jegyében fejleszteni. A végcél természetesen itt is az, hogy dinamikus HTML-, újabban XML-oldalakat hozzunk létre a böngésző ügyfél számára. Ennek a felfogásnak nyilvánvaló előnyei vannak. Használhatjuk a már ismert programozási nyelveinket és a vizuális fejlesztőeszközöket. Ennek az irányzatnak az egyik első megvalósítása a Microsoft Active Server Pages (ASP) szabványon alapuló megoldása volt, melynél az alkalmazásokat a Visual InterDev nevű eszközzel lehetett elkészíteni. Az ASP megoldás arra épül, hogy az ActiveX elemeket és az azokat működtető Visual Basic parancsfájlokat a kiszolgálóoldalon használja. Mi ennek a hátránya? Az MS-re oly jellemző megoldás: nagyszerű az ötlet, de a megoldás több részletében sem szabványos. Például a Visual Basic nyelv használata a szabványos C++ vagy Java helyett, a módszer – szerintem – csak windowsos környezetben működik rendesen, amit az OLE, az ActiveX és a Windows lehetőségeinek teljes kihasználásával lehet magyarázni. A hírek szerint az MS újabb stratégiája a „.NET”-elgondolás, amely mindjárt egy új nyelv bevezetésével indul (neve: C#). Vajon miért?

Ezután felmerül a kérdés, hogy létezik-e olyan webes fejlesztő és futtató környezet, amely tudja mindazt, amit az ASP, de felület- és gyártófüggetlen? A válasz igen. A megoldás elnevezése hasonlít az ASP nevére: JavaServer Pages (JSP). Itt szeretnék két dolgot megjegyezni:

➲ http://java.sun.com

1. A jelenlegi legelterjedtebb kiszolgálóoldali módszer még most is a Perl parancsállomány, ennek hátránya a CGI jellegű működésben rejlik.
2. Jelentősen terjed egy másik parancsnyelvi megoldás is, a PHP (PHP Hypertext Processor). A PHP egyszerűsége és nagyszerű alapötlete pont az, ami a JSP és ASP módszereket is jellemzi: készítsünk olyan HTML-oldalt, amelybe azok a kiszolgálóoldali programok vannak beágyazva (itt PHP-parancsfájlok), amelyek egy HTTP-kérés során lefutnak és az így dinamikusan kialakult HTML- (XML-) tartalom kerül át a böngészőbe. A PHP hátránya, hogy igazából ez is egy CGI módszer, néha szükségtelenül bonyolult (például ahány adatbázistípus, annyiféle API van az elérésükhöz).

A JSP és ASP rövid összehasonlítása

Térjünk vissza a JSP-re. Miért is jó ez? Vegyük sorra a párhuzamokat az ASP-vel:

- Az ASP kiszolgálóoldali ActiveX-objektumokat a Java osztályszerkezete fedi le. Ezeket az osztályokat éppen ezért servleteknek is nevezzük. Érdemes az üzleti logikát JavaBeanekben vagy – az IBM és a Sun kezdeményezése kapcsán – Enterprise JavaBeanekben (EJB) megfogalmazni. A JavaBean ugyanolyan elemalapú módszert jelent, mint ami a Delphi vagy Visual Basic (régebben VBX, manapság ActiveX) fejlesztői környezeteket is hatékonnyá teszi. A JSP abban ad nagy segítséget az alkalmazásfejlesztőnek, hogy ezeket a JavaBeaneket egyszerűen el tudja érni, azaz az alkalmazásréteg „tetején” a JSP végzi el a megjelenítő és adatbeviteli (HTML form feldolgozó) szolgáltatásokat, a JSP biztosítja a kapcsolattartást az ügyfélréteg és az alkalmazáslogika között.
- A Visual Basic parancsfájloknak szintén a Java program felel meg, amit ebben a környezetben néha scriptletnek is neveznek. Ki más is mozgósíthatná a .class fájlokban lévő servleteket, mint maga a Java nyelv? Ezen a téren az ASP és a JSP közötti hasonlatosság tökéletes. A JSP-oldalak az ASP-hez teljesen hasonló módon használják a <% ... %>, <%= ... %>, <%@ ... %> tagokat a kiszolgálóoldalon. Ezzel a JSP is tökéletesen megvalósítja az Active Documentnek nevezett megoldást. A különbség csak az, hogy a parancsfájl nyelve maga a Java. Itt felhívjuk a figyelmet egy nagyon fontos tényre: a .class fájlok dinamikus betöltése következtében a teljes Java-eszközkészletet használhatjuk a JSP-oldalakon. Ez azt jelenti, hogy az alkalmazást teljes egészében megírhatjuk Javában, és a HTML/XML-alapú felhasználói felületet kell csak JSP-ben megoldani. A JSP egyébként a servletek továbbgondolása kapcsán született meg. Régebben az SHTML (kiszolgáló-oldali HTLM) fájlok voltak azok, amelyekbe az applethez hasonlóan a <SERVLET code=….class> értékek… </SERVLET> tagok segítségével ágyazhattunk be egy servletet.
- Az ügyféloldali ActiveX-objektumoknak a Java appletek felelnek meg. Amiként az ügyfél- és a kiszolgálóoldali ActiveX objektumok is megvalósíthatnak ügyfél-, illetve kiszolgálókapcsolati lehetőséget, úgy lehetséges az applet és a servlet párbeszéde is.
- Az ASP COM/DCOM-nak a szabványos CORBA felel meg (a CORBA egy környezet- és nyelvfüggetlen protokoll megvalósítása).
- Az ASP OLE DB, ADO vagy ODBC adatbázis-elérési modellje helyett az objektumközpontú JDBC-t használhatjuk.

Ezen a ponton hagyjuk most abba az ASP és a JSP összehasonlítását. A JSP minden operációs rendszeren rendelkezésünkre áll, sőt legjobb megvalósításai a GNU felhasználási szerződése alá tartoznak, azaz beszerzésük ingyenes. (Vigyázat, ez nem a teljes tulajdonlási költséget jelenti!).

A JSP servlet működése

A JSP-oldalak első használata során azokból egy-egy servlet, azaz egy .class fájl keletkezik. A JSP célja tehát az, hogy ne kelljen a viszonylag részletgazdag servletek fejlesztésével foglalkoznunk, figyelmünket a dokumentumok kialakítására irányíthassuk (a beágyazott PHP parancsfájlhoz hasonlóan). Régebben a servletek meghívása hasonló volt a külső CGI parancsfájlok hívásához, azaz a cím így nézett ki: „http://gép/egy_servlet”. Később kialakult az SHTML-módszer, ennek használata a beágyazott PHP-hoz hasonló. Itt már a dokumentum szerkezete áll a figyelem középpontjában, mert a servletek ebbe vannak beágyazva. Jelenleg a JSP tekinthető ezen irányvonal csúcsának. A gondolat azért nagyszerű, mert a JSP-oldalakat egy webgrafikus is el tudja készíteni (ő úgy érzékeli a JSP-tagokat, mintha azok különleges HTML-tagok lennének), ugyanakkor a háttérben továbbra is a jól bevált servletek állhatnak. Itt jegyzem meg, hogy ez a felfogás tökéletes összhangban áll a mostanában viharos sebességgel terjedő XML-módszerrel.
A következőkben olyan JSP servletfejlesztő és -futtató programkörnyezet-kialakítást ismertetek, amely bármelyik operációs rendszeren megvalósítható és ebben a pillanatban a legkorszerűbb. A kialakítás ismertetését Linux operációs rendszerre írom le, de az alapelvek más operációs rendszereknél is teljesen hasonló. A harmadik részben pedig példán keresztül mutatom be e környezet használatát.

A fejlesztői és a futáskörnyezet kialakítása

Nézzük először is az általam használt hozzávalókat:

- Apache 1.3.14 kiszolgáló,
- Apache Jakarta-Tomcat 3.2.1,
- Sun Java 2 SDK Standard Edition for Linux,
- Borland Interbase v6.0 adatbázis-kiszolgáló,
- Borland InterClient JDBC meghajtó az Interbase eléréséhez.

A fenti programok közös jellemzője megbízható működésük, nagy cégek gyártják őket és a beszerzésük ingyenes. Kezdjünk neki kiszolgálónk kialakításának!

Az Apache telepítése

Az Apache kiszolgáló forráskódja a ➲www.apache.org helyről letölthető. A letöltendő fájl neve: apache_1.3.14.tar.gz.
A forráskódból telepített rendszereket a /opt könyvtár alá másoljuk:

```
$ cp apache_1.3.14.tar.gz /opt
```

Csomagoljuk ki:

```
$ cd /opt
$ tar -xvzf apache_1.3.14.tar.gz
```

Ennek hatására a forráskód a /opt/apache_1.3.14 könyvtárba kerül, erre – csupán a kényelem kedvéért – készítsünk közvetett hivatkozást:

```
$ ln -s apache_1.3.14 apache
```

Linux-rendszeren több módszer létezik arra, hogy a forráskódból hogyan készítsünk futtatható rendszert. A legelterjedtebb a configure nevű parancsállomány, ez feltérképezi Linux-rendszerünket és ahhoz illeszkedő Makefile állományt állít elő. Ezt használja az Apache is. Adjuk ki ennek megfelelően a `configure` parancsot:

```
./configure
            --sysconfigdir=/etc/httpd  \
            --datadir=/home/httpd  \
            --logfiledir=/var/log/httpd \
            -- disable-rule \
            --enable-shared=max \
            --enable-module=most
```

Ez a parancs olyan Makefile-t készít, amely előírja azt is, hogy hol lesz az Apache beállítási fájljának a helye (/etc/httpd), a webtartalom gyökere (/home/httpd) és a naplófájl. Továbbá engedélyezzük a futás közben betöltődő modulok használatát is.
A következő lépés a forráskód lefordítása a `make` parancs kiadásával. A fordítás után a bináris kód telepítése a `make install` paranccsal lehetséges. A futtatható Apache-rendszer a telepítés után a /usr/local/apache könyvtárban helyezkedik el.

A SUN JAVA 2 telepítése

A jdk-1.2.2-se.tar.gz fájlt (Java 2 Standard Edition) is másoljuk a /opt könyvtárba, majd a `tar -xvzf jdk-1.2.2-se.tar.gz` segítségével az ismertetett módon csomagoljuk ki. Ekkor létrejön egy /opt/jdk1.2.2 könyvtár, ahol a Java 2 rendszer található. Készítsünk erre is hivatkozást:

```
ln -s jdk1.2.2 jdk
```

Egy kicsit előretekintve módosítsuk a /etc/profájl fájlt, mert itt állítja be az általunk használt bash-héj környezeti változóit (Windowsban erre az autoexec.bat-ot használnánk).

```
# A /etc/profile végére ezt írjuk:
# A java home könyvtár
JAVA_HOME="/opt/jdk"
# A programok keresési útvonalának kiegészítése:
PATH=$JAVA_HOME/bin:$PATH
# A TOMCAT JSP és servlet szolgáltató helye
TOMCAT_HOME="/opt/tomcat"
# Az apache helye
APACHE_HOME="/usr/local/apache"
TCJ=/opt/tomcat/lib
JDKJ=/opt/jdk/jre/lib
CLASSPATH=.:$JDKJ/rt.jar
CLASSPATH=$CLASSPATH:$TCJ/servlet.jar:
      ➥$TCJ/jasper.jar:$TCJ/ant.jar:
      ➥$TCJ/servlet.jar:$TCJ/jasper.jar:
      ➥$TCJ/ant.jar:$TCJ/servlet.jar:
      ➥$TCJ/jasper.jar:$TCJ/ant.jar:$TCJ/jaxp.jar:
      ➥$TCJ/parser.jar:$TCJ/webserver.jar:
      ➥$JAVA_HOME/lib/tools.jar
export CLASSPATH, JAVA_HOME, TOMCAT_HOME,
APACHE_HOME
# A /etc/profile változtatás vége.
```

Ezekkel a változtatásokkal a Java fordító és futtató is meg fogja találni a .class fájlokat. Ezeket a .class fájlokat régebben a fájlrendszerben helyezték el, majd .zip fájlban tárolták (itt is benne volt a könyvtárszerkezet). Mostanában .jar fájlokat használnak, ennek zip a formátuma, de a jar Javában íródott. A Java fordító és futtató a .class fájlokat igény szerint, dinamikusan használja, ehhez azonban meg kell találni őket. A keresés a következőképpen zajlik: a program indító könyvtára, ha itt nincs, akkor a JAVA_HOME által kijelölt helyen lesz. Amennyiben ott sincs, akkor a CLASSPATH által megjelölt helyeken folytatódik a keresés.

Az Apache Jakarta-Tomcat telepítése

Az Apache-projekt webkiszolgálóját már régen kiegészítette servlethívási modullal, amit a mod_jserv.so (Apache_JSERV csomag) modul valósít meg. A JSP használatát először az erre épülő GNU JSP tette lehetővé. Ezt a párost váltotta fel az egységes Jakarta-Tomcat csomag, ami a servlet és JSP-lehetőségeket is magában foglalja. A csomagot szintén az Apache projekt ➲ www.apache.org helyéről lehet letölteni. A letöltendő fájl neve: jakarta-tomcat-3.2.1.tar.gz (bináris), illetve jakarta-tomcat-src-3.2.1.tar.gz (forráskód). Másoljuk be mindkét fájlt a megszokott /opt helyre, majd csomagoljuk ki a bináris csomagot a `tar -xvzf jakarta-tomcat-3.2.1.tar.gz` paranccsal, ennek hatására létrejön a /opt/jakarta-tomcat-3.2.1 könyvtár. Készítsünk rá hivatkozást:

```
ln -s /opt/jakarta-tomcat-3.2.1 tomcat
```

Emlékezzünk vissza, hogy a /etc/profile fájl TOMCAT_HOME változója erre a hivatkozásra mutat. A Tomcat telepítése ezzel a kicsomagolással lényegében befejeződött. Még annyit kell tenni, hogy a Tomcat-rendszer mod_jk.so nevű fájlját bemásoljuk a /usr/local/apache/libexec helyre. Itt tárolja az Apache a dinamikusan töltődő modulokat. A mod_jk.so csatolófelülete valósít meg az Apache számára. Ez azt jelenti, hogyha .JSP vagy servlet kérés megy az Apache felé, akkor az ezt kiutalja a mod_jk.so modul használatával a Tomcat felé. A Tomcat előállítja a dinamikus HTML-tartalmat, ezt az Apache szolgáltatja az ügyfél felé. Ahhoz, hogy az Apache tényleg így működjön, ezt meg kell mondani neki. Létezik a /opt/tomcat/conf könyvtárban egy „mod_jk.conf-auto” fájl, amit nem szabad szerkeszteni, mert úgy jó, ahogy van. A teendőnk csak annyi, hogy az Apache beállítófájljába a /etc/httpd/httpd.conf-ba utolsó sorként írjuk be a következőt:

```
include /opt/tomcat/conf/mod_jk.conf-auto
```

Ezzel az Apache jól fogja használni a kapcsolatot a Tomcathez.
A Tomcatet viszont még be kell állítani. Ezt a server.xml és a workers.properties fájlok módosításával tehetjük meg. Mindkettő a /opt/tomcat/conf könyvtárban helyezkedik el.

1. A workers.properties módosítása a következő sorok beírását, kijavítását jelenti:

```
# adjuk meg, hogy hol van a Tomcat
workers.tomcat_home=/opt/tomcat
# adjuk meg, hogy hol van a Java
workers.java_home=/opt/jdk
```

```
# adjuk meg, hogy a Linuxban ez a path határolójel
ps=/
# A Java virtuális gép helye
worker.inprocess.jvm_lib=/opt/jdk/jre/lib/i386/
classic/libjvm.so
```

2. A server.xml szerkesztése
Ha jó nekünk, hogy a JSP fájlok és a servletek a /opt/tomcat/webapps/root helyen vannak, akkor ezt a fájlt nem szükséges átírni.
Ezzel a Tomcat is működőképes lett. Hogyan indítsuk ezek után a webkiszolgálónkat?
Az első lépés a Tomcat indítása:

```
/opt/tomcat/bin/startup.sh.
```

Ez a parancsfájl is a kicsomagolt Tomcat része.
A második lépés az Apache indítása:

```
/usr/local/apache/bin/apachectl start
```

A webkiszolgáló leállítása a `/usr/local/apache/bin/apachectl stop`, majd a `/opt/tomcat/bin/shutdown.sh` parancspárral lehetséges.

A Borland Interbase adatbázis-kiszolgáló telepítése

A Borland szabaddá tette Interbase néven futó adatbázis-kiszolgálóját, ami tudását tekintve megközelítőleg az MS SQL kiszolgálóval egykategóriás, és webes célokra kiválóan alkalmazható.
A csomag telepítése rendkívül egyszerű. Az ...Interbase...6.0.tar.gz csomagot másoljuk a /opt helyre és a `tar -xvzf ...` paranccsal csomagoljuk ki. Ezután csak annyi a teendő, hogy a /etc/services fájlba beírjuk az új TCP/IP-szolgáltatást, azaz a fájlt a következő sorral kell kiegészíteni:

```
gds_db 3050/tcp
# hálózatos interbase
```

Ebből azt is látjuk, hogy az Interbase-t használó ügyfelek másik számítógépen is futhatnak, és amikor az adatbázis-kiszolgálót akarják használni, akkor ezt a 3050-es TCP-kapun tehetik meg. Ez hasonló az Oracle Net * protokolljához.
Az Interbase kiszolgálót a következő paranccsal lehet elindítani:

```
# az & hatására a folyamat háttérben fut.
/opt/interbase/bin/ibserver &
```

Az Interbase telepítésekor felkerült egy isql nevű program is, ami az adatbázis felé egy SQL parancsfelületet nyújt (hasonlóan, mint az Oracle *Plus). Indítsuk el „sysdba” felhasználóként, „masterkey” jelszóval:

```
isql -u sysdba -p masterkey
```

Ekkor egy „SQL>” készenléti jelet kapunk meg, ahonnan SQL-parancsokat adhatunk ki. Hozzunk létre egy új adatbázist az /opt/interbase/adatok könyvtárba:

```
SQL> create database
"/opt/interbase/adatok/webimi.gdb";
```

Az Interbase-adatbázisok egy-egy gdb kiterjesztésű fájlban találhatók (itt van minden: táblák, nézetek, tárolt eljárások, triggerek, indexek stb.).

A következő dbgen.sql parancsfájl egy táblát hoz létre ebben az adatbázisban:

```
# dbgen.sql
create table TELEFONOK
(
 azon    NUMERIC(10) NOT NULL,
 nev     VARCHAR(50),
 telefon VARCHAR(20),
 primary key( azon )
);
```

Ezt a parancsfájlt az isql `input` parancsával futtathatjuk:

```
input /opt/interbase/adatok/dbgen.sql;
```

Próbaként szúrjunk be egy sort ebbe a táblába:

```
Insert into TELEFONOK values(1, 'Nyiri Imre',
'123456789');
commit;
```

Lépjünk ki az isql-ből az `exit`; paranccsal.

Az Interbase Java JDBC meghajtó telepítése

A meghajtó az Interclient...tar.gz fájlban található, amit az eddigiek szerint másoljunk a /opt könyvtárba és csomagoljuk ki.
Ekkor létrejön a /opt/interclient_install_temp_dir könyvtár, lépjünk be, majd futtassuk le az ott lévő „install.sh” parancsfájlt. A parancsfájl azon kérdésére, hogy hova telepítse a JDBC meghajtót, adjuk meg a /opt/interclient könyvtárat. A telepítés után már csak annyi a teendő, hogy az interclient.jar nevű fájlt hozzáfűzzük a CLASSPATH-hoz a /etc/profile-ban. Ezzel kész az Interbase JDBC meghajtó telepítése. Próbáljuk ki, hogy működik-e! Írjunk ehhez Java programot.

```
//
// A program az előző pontban létrehozott
// adatbázist használja
// asz1.java, ahol asz=adatszolgáltató
//
import java.sql.*;
public class asz1
{
  public String ir()
  {
    String databaseURL =
              ➥"jdbc:interbase://localhost/opt/
              ➥interbase/adatok/webimi.gdb";
    String user = "sysdba";
    String password = "masterkey";
    String driverName =
              ➥"interbase.interclient.Driver";

    Driver d = null;
    Connection c = null;
    Statement s = null;
    ResultSet rs = null;

    // Driver load
    try
    {
     Class.forName
              ➥("interbase.interclient.Driver");
    }
    catch ( Exception e)
```

```
    {
      return "???";
    }

    // Kapcsolatlétrehozás
    try
    {
        c = DriverManager.getConnection
(databaseURL, user, password);
    }
    catch ( SQLException e)
    {
        return "???";
    }

    // auto commit hamisra állítása
    try
    {
        c.setAutoCommit (false);
    }
    catch (java.sql.SQLException e)
    {
        return "???";
    }

   // Egy lekérdezése
   try
      {
         s = c.createStatement();
        rs = s.executeQuery
         ("select NEV, TELEFON from TELEFONOK");
       ResultSetMetaData rsmd = rs.getMetaData();
       int cols = rsmd.getColumnCount();
       rs.next();
       return rs.getString("NEV");
      }
      catch ( SQLException e )
      {
            return "???";
      }
 }
} // end class
```

Most fordítsuk le a programot a `javac asz1.java` paranccsal, ennek eredményeként keletkezik az asz1.class fájl. Próbaképpen írjunk egy konzolalapú Java főprogramot, amely ezt az osztályt használja:

```
// imre.java
public class imre
{
  public static void main(String[] args)
  {
    System.out.print( new asz1().ir() );
  }
}
```

A program létrehoz egy új „asz1” típusú objektumot és meghívja annak az ir() tagfüggvényét, ami a telefonok tábla első sorának NEV mezőjét, tehát a „Nyiri Imre” karakterláncot adja vissza. A System.out.print() pedig kiírja ezt a konzolra. Működik tehát az adatbázis-kapcsolat. A programot a `java imre` paranccsal indíthatjuk el. Ennek hatására a „Nyiri Imre” szöveg kiíródik a konzolra.

Ezzel a fejlesztő, futtató környezet minden eleme a helyére került, és befejeztük a programok telepítését.

Az Apache – JSP környezet kipróbálása

A környezet kipróbálásához felhasználjuk az eddig létrehozott eredményeinket:

- A webimi.gdb adatbázist (benne a Telefonok táblát)
- A már elkészített asz1.java és asz1.class fájlokat

A jobb érthetőség kedvéért, valamint a servletek és a JSP-oldalak összehasonlíthatóságához elkészítünk egy olyan dinamikus HTML-oldalt létrehozó modult, amely olyan HTML-oldalt küld az ügyfeleknek, ami a napszaknak megfelelően köszön, majd kiírja a *Telefonok* tábla első sorának telefontulajdonosát.

A feladat JSP-alapú megoldása

Nos tehát, az általunk kialakított környezet működőképességének vizsgálatához készítsünk egy nagyon egyszerű JSP-oldalt, melynek a neve legyen elso.jsp. Ezt a böngészőből a „http://localhost/elso.jsp” sorral vagy egy erre mutató hivatkozással hívhatjuk meg.
Nézzük meg a JSP fájl tartalmát (a JSP fájlok text fájlok):

```
<%@ page import="java.util.Calendar" %>
<% if (Calendar.getInstance.get(Calendar.AM_PM)
       == Calendar.AM ) { %>
Kellemes délelőttöt kívánok! <br>
<% } else { %>
Kellemes délutánt kívánok! <br>
<% } %>
A telefontulajdonos neve: <%= new asz1().ir() %>
<br>
```

Az elso.jsp-re való hivatkozás után az Apache felméri, hogy ez JSP-oldal-e. A mod_jk csatoló segítségével megkéri a Tomcatet, hogy dolgozza fel ezt a fájlt. A Tomcat az elso.jsp fájlból a háttérben egy Java classt (servletet) készít, majd ezt lefuttatja a Java VM-mel. Itt van egy fordítási szakasz, ami csak az első alkalommal történik meg, utána már mindig a lefordított class fog futni. Ennek eredménye a következő dinamikus HTML lesz (csak amit a böngészőből látunk):

Kellemes délutánt kívánok!
A telefontulajdonos neve: Nyiri Imre

Ugye, milyen nagyszerű? Volt egy teljes Java alkalmazásunk az asz1.class fájlban (ezt egy kis túlzással egy JavaBeannek is nevezhetnénk), amit meghívtunk a JSP oldalról. Ugyanakkor az is látszik, hogy tetszőleges Java program írható be a JSP-oldalakra.

A feladat servletalapú megoldása

Természetesen most csak nagyon egyszerű megoldást fogunk látni, így aki a servleteket akarja tanulmányozni, annak ajánljuk a Sun servlet megvalósításait. Nézzük a programot:

```
// Importok
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
import java.net.*;
import java.util.Calendar;

//
// Ennek hívása a böngészőből:
// http://localhost/elso.class
//
```

```
public class elso extends HttpServlet
{
  // a servlet előkészítése
  public void init( ServletConfig conf ) throws
ServletException
  {
    super.init( conf );
  }

  // Szolgáltatása: html oldal előállítás
  //
  public void service( HttpServletRequest req,
                       // inputkérés
                          HttpServletResponse res )
                       // html outputhoz
  {
     String sz = (new asz1()).ir();
                       // a TELEFONOK táblából
     String udv; // üdvözlés

     if (Calendar.getInstance.get(Calendar.AM_PM)
                       == Calendar.AM )
     {
      udv = "Kellemes délelőttöt kívánok! <br>";
     }
     else
     {
      udv = "Kellemes délutánt kívánok! <br>";
     }

    res.SetContentType("text/html");
    ServletOutputStream out =
              res.getOutpitStream();
    out.println("<head><title>Az
              elso.class</title></head><body>");
    out.println( udv );
    out.println("A telefontulajdonos neve: ");
    out.println( sz ); out.println("</body>");
  } // end service

} // end elso class
```

Tekintettel arra, hogy a Tomcat is servletté fordítja a JSP-oldalt (a Tomcat work könyvtárában azt is mindig megnézhetjük, hogyan is néz ki ennek a JSP-ből előállított servletnek .java forrása), így a servletek kezelését is ismeri. Az azonban talán már mindenki előtt nyilvánvaló, hogy érdemes kihasználni azt, ahogy a JSP a servletet önműködően előállítja.

Záró gondolatok

Ez az írás csupán rövid ismertető volt arról, hogy miért is és hogyan használható a Java nyelv a kiszolgálóoldali programozáshoz. Van még néhány olyan kérdés, amit tisztázni kell, mielőtt valaki belevág módszer mélyebb használatába. Nézzük őket!

Milyen fejlesztőeszköz használható?

Az nyilvánvaló, hogy egyszerű Java fejlesztői környezet (Borland Jbuilder, Oracle Jdeveloper, IBM VisualAge for Java) nem elégséges, hiszen a .java, és .class fájlokon kívül még a következő nyersanyagokra is szükség van:

- HTML-szerkesztő,
- JSP-szerkesztő,
- CGI parancsfájlok írása-kezelése,
- Appletkészítő eszköz,
- HTML parancsfájlírást támogató eszköz (JavaScript, VB script),
- Grafikák, hangok stb. kezelése.

Ezenkívül ezeket az alapanyagokat egységes webalkalmazásként kell vezérelni. Ezt a feladatot az MS Visual InterDev remekül ellátja, de létezik az IBM gondozásában (az e-Business jegyében) egy másik eszköz is, ami ugyanilyen magas szinten tud webes alkalmazást kialakítani és vezérelni, sőt a JSP-beli támogatottsága egyértelműen jobb az InterDevnél: az IBM WebSphere Stúdió.
Az IBM VisualAge-ben úgy lehet kipróbálni a webes alkalmazást, hogy egy időben egyik ablakban látjuk a JSP-kódot, a mellette lévőben a JSP-ből előállított servlet kódját, a harmadik ablakban pedig magát a megvalósított HTML, WAP vagy VoiceXML stb. kódját.
Az így kialakított és kipróbált alkalmazás ezután bármilyen környezetben futtatható (Oracle Appserver, Apache + Tomcat, Websphere stb.).
Tanulási célra alkalmas fejlesztői környezet a Sun JSWDK 1.0 eszköz, mely ingyenesen letölthető.

A HTML-oldalak űrlapjainak feldolgozása

A HTML-űrlap-feldolgozás kihasználja a Bean/JSP előnyeit. Az mindenkinek kedvére való, ahogy a PHP egy „nev” nevű beviteli elemet $nev néven ér el. A JSP-ben sem bonyolultabb a helyzet. Egy html űrlap „nev” és „lakcim” nevű mezőjét a bean egy-egy tagfüggvényével tudjuk kezelni, ezeket célszerű getNev(), getLakcim(), setNev(), setLakcim() névre keresztelni, ezután a JSP-motor képes ezeket a mezőket önműködően kezelni. Ezt a folyamatot *introspection*-nek hívják és hihetetlen módon leegyszerűsíti az űrlapfeldolgozást.

A webhely grafikájának megvalósítása

A JSP-oldal lényegében egy HTML-oldal, szerkesztése, formázása gyakorlatilag bármilyen HTML-szerkesztőben történhet. A servletek baja az volt (hasonlóan a CGI parancsfájlokhoz), hogy a HTML-oldal külalakját – kód formájában – magukban hordozzák. Ha az alkalmazás kinézetén valami nem tetszik, akkor a servlet kódjába kell belenyúlni, azt újra kell fordítani és be kell vezetni. A valóságban azonban vannak olyan emberek, akik jól tudnak programozni és vannak olyanok, akik szép formát képesek alkotni. Ennek megfelelően biztosítani kell a külön munkavégzés lehetőségét. Természetesen úgy is fogalmazhatunk, hogy a JSP megkíméli a programozót a látvány programozásától, mert azt ezek után vizuálisan is elkészítheti. A programozók elkészítik a beaneket, a látványtervezők (grafikusok) pedig a JSP-oldalakat, amit ugyanúgy tudnak formázni, mintha HTML-oldalak lennének. A JSP/Bean szemléletű alkalmazás módosítása egyszerű: ha valami rossz a logikában, akkor annak kijavításához nem kell a látványtervező és fordítva.

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.

Kapcsolódó címek

- http://java.sun.com
- http://javasite.bme.hu
- http://www.servlets.com
- http://www.javasoft.com/products/jsp
- http://www.jspin.com

Wine

Hogyan futtassunk Windows-alkalmazásokat Linux alól?

A Linux-felhasználók egyik igen nagy gondja, főleg ha korábban sok windowsos programot használtak, az, hogyan találjanak olyan linuxos programokat, amelyekkel Windows nélküli környezetben is képesek ellátni feladataikat.
A Linux-változatok általában tartalmaznak olyan eszközöket, amelyekkel a különböző operációs rendszereken futtatott programok valamilyen szinten használhatóvá válnak. A linuxos alapcsomagok általában három emulátorcsomagot tartalmaznak: ezek a közismert dosemu és xdosemu a DOS-os programokhoz, a BasiliskII a MacOsen futó programokhoz, illetve a számunkra fontos windowsos alkalmazások használatához: a Wine.
Mindannyian ismerjük, vagy legalábbis hallottunk a VMware-ről, ami ugyan tökéletesen alkalmas a különböző operációs rendszerek és alkalmazásaik futtatására, viszont a programokhoz használni kívánt operácis rendszert mindenképp telepíteni kell a virtuális vagy valós lemezrészre.
A Wine nem az egyetlen Windows-emulátor Linuxhoz, hiszen létezik még például a Wabi is, de ez mára már szinte teljesen háttérbe szorult.
Jelenleg a Wine számos windowsos program futtatására alkalmas, erre az egyik legjobb példát a Corel cég szolgáltatja a Corel Photo-Paint 9 for Linux, illetve a Corel Office 2000 for Linux programcsomagok kiadásával. E két programcsomagban az eredeti windowsos alkalmazást tették futtathatóvá a Wine egyedi beállításai segítségével. Az előbbihez volt szerencsém, és véleményem szerint az első indítástól eltekintve – amikor a betűkészletek Wine alóli használatához azok adatait adatbázisba gyűjti – a program igen megbízható, a windowsos változathoz képest (általában csupán) kis sebességcsökkenéssel tökéletesen működött.
Igen kényelmes helyzetben vannak azok, akiknek valaki vagy valamely cég beállítja a Wine-t, hogy más programok tökéletesen fussanak, de sajnos nem ez az általános. Néha nekünk kell módosítgatnunk kézzel a Wine beállításait a beállításfájlban, vagy a CD-mellékleten található *WineSetupTK* nevű grafikus varázsló segítségével.
Először is lássuk, honnan lehet a Wine-t beszerezni. Szerencsére viszonylag sok helyről letölthetjük, hiszen számos tükröoldal létezik, de az alábbi webhelyekről szinte biztosan sikerülni fog:
➲http://www.winehq.com
➲http://www.codeweavers.com/wine
➲http://www.codeweavers.com/winesetuptk.shtml.
A beállításokat két fájl tartalmazza, ezek a /etc/wine.conf (minden felhasználó ezt alkalmazza, ha nincsenek saját egyedi beállításai), illetve a ~/.wine/config (a felhasználó egyedi beállításait tartalmazza). Fontos, hogy az előbbit csak a rendszergazda szerkesztheti, de mindenki olvashatja, az utóbbit viszont csak a felhasználó szerkesztheti, illetve olvashatja!
E két fájl felépítése közel azonos (a wine-user.pdf alapján).

1. táblázat

A rész neve	Szükséges	Feladata
[Drive X]	igen	A Wine által használt lemezegységeket adja meg
[wine]	igen	A Wine könyvtárait állítja be
[DllDefaults]	programfüggő	Az alapból beolvasandó DLL-ek
[DllPairs]	programfüggő	Javítandó DLL-párok
[DllOverrides]	programfüggő	Lecserélendő DLL-ek
[options]	nem	Esetenként azonos a következővel
[x11drv]	programfüggő	Az X11 használata
[fonts]	igen	Betűkészlet-beállítások
[serialports]	nem	A soros kapuk beállításai
[parallelports]	nem	A párhuzamos kapuk beállításai
[spooler]	nem	A nyomtatási sor beállításait tartalmazza. Pontosabban azt, hogy az egyes kapura küldött adatokkal a program mit kezdjen, például milyen nyomtatási parancsot hajtson végre.
[ports]	nem	Közvetlen kapuhozzáférés
[spy]	nem	Mit tegyen bizonyos hibaüzeneteknél
[Registry]	nem	A Windows rendszerleíró beállításai
[tweak.layout]	programfüggő	A Wine külső megjelenése
[programs]	nem	A Wine indításakor önműködően lefuttatandó program
[Console]	nem	Konzolbeállítások

A továbbiakban az *1. táblázatban* látott részekben megadható beállításokat részletezzük.

1. [Drive X]

Ez a beállítás található a Wine beállításfájl elején. Ez adja meg a Wine által használt adattároló-egység nevét, elérési útját és fájlrendszerét. Lássunk erre egy példát:

```
[Drive C]
```

Megadom az eszköz nevét, itt C: lesz.

```
"Type" = "hd"
```

Megadom az eszköz típusát, ez a következő lehet: /floppy (hajlékonylemezhez) /hd (merevlemezhez) /cdrom (CD-ROM-hoz, csak olvasható eszközhöz) /network (hálózati vagy különböző jogosultságú beállításokat tartalmazó könyvtárakhoz).

```
"Path" = "/mnt/win"
```

Az eszköz elérési útvonala, mely tartalmazza a csatolási könyvtárat. Lehetséges változó értékének beillesztése is:

```
"Path" = "${HOME}"
"Label" = "win"
```

Ez tetszőleges eszköznév megadását teszi lehetővé. Használata

például CD-ROM esetén célszerű, amikor az indított program számára fontos a CD elnevezése.

```
"Device" = "/dev/hda10"
```

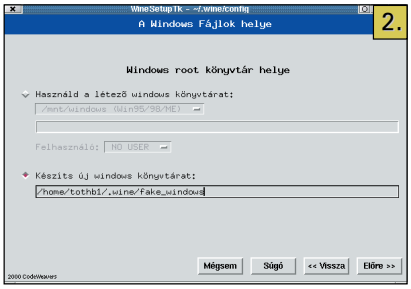
Az eszköz /dev könyvtárbeli, neve a csatolási pont.

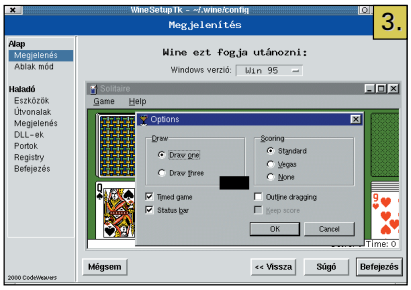
```
"Filesystem" = "win95"
```

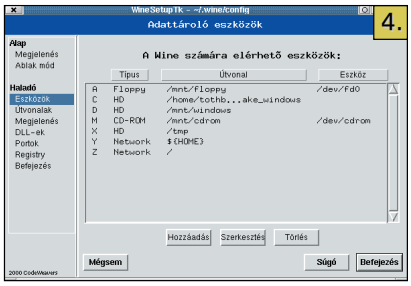
A "Filesystem" változóhoz háromféle fájlrendszer állítható be:

- Msdos: a csatolt FAT 16-os fájlrendszerek esetén használható
- Win95: Unix, FAT 32, CD-ROM-okhoz, egyéb eszközökhöz használható
- Unix: ezt manapság a „Win95” helyettesíti.

Fontos megemlítenem, hogy nem feltétlenül szükséges az összes beállítást használni, mivel alapértelmezésben a „Win95”-ös beállítást tartalmazza, több beállítás pedig bizonyos esetekben csupán tájékoztató jellegű (például a /tmp könyvtár megadása Windows *Temp* könyvtárként).







2. [wine]

A Wine-modul tulajdonképpen a Windows alapkönyvtárának elérési útját, illetve a windowsos környezet egyéb szükséges beállításait tartalmazza.

Ebben a részben a [Drive X] megadott eszközök neveivel is megadhatunk könyvtárakat:

```
"Windows" = "C:\\Windows"
```

A *Windows* könyvtár helye.

```
"System" = "C:\\Windows\\system"
```

A Windows *System* könyvtár helye.

```
"Path" = "C:\\Windows;
        ➥C:\\Windows\\system;
        ➥X:\\;Y:\\"
```

A DOS `path` parancsához hasonló feladatai vannak, futás közben a rendszer e könyvtárakban keresi a programokat.

```
"Temp" = "X:\\"
```

Fontos a *Temp* könyvtár megadása, hiszen a Windows ide helyezi a létrehozandó ideiglenes fájlokat.

```
"GraphicsDriver" = "x11drv"
```

Az alapértelmezett X11 eszközmeghajtót fogja használni.

```
;ShowDirSymlinks=1
```

A Wine alapértelmezésben nem engedélyezi a Windows-programok számára a közvetett könyvtárhivatkozások használatát. A sor kiemelése a beállításfájl szerint, a közvetett hivatkozásokat tartalmazó könyvtárat használó program összeomlása a fájlrendszer-fa sérülését okozhatja, ilyenkor a sérült könyvtár önmagára fog mutatni. (Tapasztaltam sajnos ezt a hibát, ilyenkor például az MC-ben a legfelső sor a sérült könyvtárat fogja megjeleníteni, és ha belelépünk, ugyanazt a tartalmat fogjuk látni, mint előtte. Szerencsére, ahogy beléptem a sérült könyvtárszerkezetbe, ugyanúgy ki is tudtam lépni belőle.)

3. [DllDefaults]

Ez a rész a DLL-csoportok alapértelmezett beolvasási sorrendjét tartalmazza. Ha az első beállítás nem érhető el, akkor a Wine a másodikkal próbálkozik.

```
"DefaultLoadOrder" = "builtin, so, native,
elfdll"
```

A „native” a Microsoft Windows DLL-jeit, a „builtin” a Wine-ba beépített DLL-eket, az so a Unix .so fájljainak valós idejű átalakítását, az elfdll pedig a Wine .so fájljában található Windows-barát DLL összeállítását jelenti.

4. [DllPairs]

A DLL-párokat megadó rész használata ugyan nem kötelező, de időnként szükség lehet rá. Például akkor, amikor SHELL32-t szeretnénk használni egyszerű SHELL-re építve. Az alábbi beállítás használata minden esetben megfelelő:

```
kernel=kernel32
gdi=gdi32
user=user32
commdlg=comdlg32
commctrl=comctl32
ver=version
shell=shell32
lzexpand=lz32
winsock=wsock32
```

5. [DllOverrides]

A beolvasott DLL-csoportok esetenként olyan DLL-eket tartalmazhatnak, amelyek ugyanazokat a DLL-eket olvassák be különböző helyekről. Emiatt meg kell adni azt, hogy melyiket használja a Wine-nal indított program. Például:

```
"kernel32" = "builtin"
```

6. [x11drv]

Az utóbbi hónapokban az X11 eszközeinek használatában igen gyors fejlődés tapasztalható. Az alábbi felsorolás a beállításoknak csak egy részét tartalmazza.

Mennyi a rendszerpalettából használható színek száma?

```
"AllocSystemColors" = "100"
```

Saját színtérképet használjak?

```
"PrivateColorMap" = "N"
```

Mekkora a használt színmélység a több színmélység-beállítású képernyőkön?

```
ScreenDepth = 16
```

A használt X11 felület neve:

```
Display = :0.0
```

Az ablakkezelő kezelje a Wine által megjelenített ablakot?

```
"Managed" = "Y"
```

Az Asztalon futtassa a Wine-ablakot?

Értéke lehet "N", vagy egy felbontás-értékpár ("800x600").

```
"Desktop" = "N"
```

Ha elérhető, használja az XFree86 DGA kiterjesztését?

```
"UseDGA" = "Y"
```

Használja az XShm kiterjesztést, ha az elérhető?

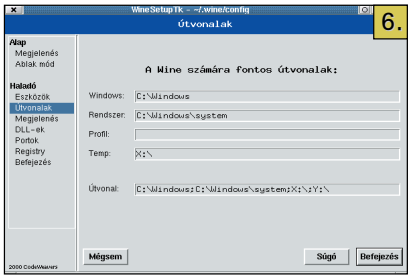
"UseXShm" = "Y"

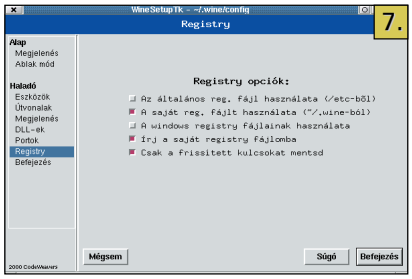
Engedélyezze a DirectX egérrögzítést? (Az egér mozgása csak az ablakra korlátozódik.)

"DXGrab" = "N"

Kétszeres átmeneti tárral jelenítse meg az ablakot? (Ezt az OpenGL-játékok használják.)

```
"DesktopDoubleBuffered" = "N"
;; XVideoPort = 43
```





Ezt a beállítást akkor használjuk, ha a videoeszköz-beállítás több kapucímet is tartalmaz.
Ha nem adunk meg semmit, akkor a Wine az elsőként megtaláltat fogja használni.

7. [fonts]

Ezen rész bemutatása összetett feladat, mivel a betűtípusok telepítése nemcsak a Wine-on múlik, hanem a rendelkezésre álló betűkészlet-kiszolgálón is. A config fájlban általában két alapbeállítás található:

```
"Resolution" = "96"
```

A megjelenített szöveg felbontásáért felel. Értéke 60–120 közötti lehet, 96 az általánosan használt.

```
"Default" = "-adobe-times-"
```

Az alapértelmezett betűtípus megadására szolgál.

8. [serialports]

A használt soros kapuk beállítására szolgáló rész, ebben a DOS kapunevet hozzá tudjuk rendelni a linuxos eszköznévhez:

```
"Com1" = "/dev/ttyS0"
```

Érdekesség, hogy például a modemet használó alkalmazások számára megadhatjuk a modem sebességét.

```
"Com3" = "/dev/modem,38400"
```

9. [parallelports]

A használt párhuzamos kapuk beállítására szolgáló rész, itt szintén a DOS alatti neveket tudjuk összekapcsolni a linuxos eszköznevekkel:

```
"Lpt1" = "/dev/lp0"
```

10. [spooler]

A spoolerbeállítás a párhuzamos kapukra érkező adatkezelést irányítja. Így például az LPT1-re érkező adatokat a Wine az `lpr` paranccsal fogja kinyomtattatni.

```
"LPT1:" = "|lpr"
"LPT2:" = "|gs -sDEVICE=bj200
          ➥-sOutputFile=/tmp/fred -q -"
"LPT3:" = "/dev/lp3"
```

11. [ports]

Közvetlen kapuhozzáférés esetén meg kell adni a kapuk címeit – külön az írásra és az olvasásra használtakat:

```
;read=0x779,0x379,0x280-0x2a0
;write=0x779,0x379,0x280-0x2a0
```

12. [spy]

Ez a rész a hivatalos leírás szerint a hibaüzenetek kiíratásához szükséges.

13. [registry]

A Windows 95-től a *Registry* (Rendszerleíró adatbázis) a Windowsok fontos részévé vált, hiszen felváltva a különböző *.ini fájlokat, a rendszer ebben tárolja a fontosabb beállításokat. Továbbá a hozzá kapcsolódó fájlokban tárolódnak a különböző rendszer- és felhasználói beállítások. A Wine munkakönyvtárában általában létrehoz egy vagy több .reg fájlt a módosuló rendszerleíró beállítások tárolására. Az összes ebben a részben szereplő változó logikai típusú, ezért értékük „igen” esetén `Y/y/T/t/1` lehet, „nem” esetén pedig `N/n/F/f/0`. Alapértelmezésben az összes beolvasott érték a saját .reg fájlba kerül eltárolásra. A rendszerszintű rendszerleíró fájl a /etc könyvtárban található. A következő sorok a *Registry* tartalmát próbálják meg érthetőbbé tenni:

Beolvassa-e rendszerszintű *Registry* fájlt?

```
"LoadGlobalRegistryFiles" = "N"
```

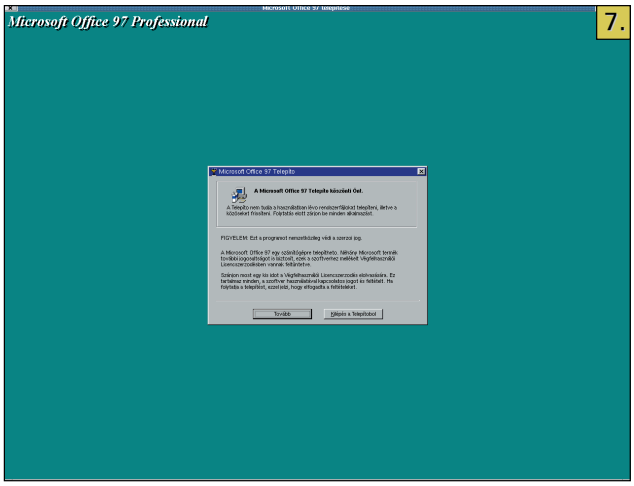
Beolvassa-e saját *Registry* fájlomat, ami a ~/.wine/ könyvtárban kerül elhelyezésre?

```
"LoadHomeRegistryFiles" = "Y"
```

Ha a gépünkön van telepített eredeti Windows, akkor lehetőség nyílik a rendszerleíró adatainak beolvastatására is.

```
"LoadWindowsRegistryFiles" = "N"
```

A változásokat a felhasználó saját rendszerleíró fájljába írja vissza?



(Ez a szolgáltatás akkor hasznos, ha a központi rendszerleíró fájlt nem akarjuk változtatni. Ilyenkor ugyanis a változások elvesznének, de így a felhasználó saját rendszerleírójában tárolódik.)

```
"WritetoHomeRegistryFiles" = "Y"
```

A rendszerleírót időszakosan lementheti a rendszer (a megadott érték másodpercekben értendő).

```
; PeriodicSave=600
```

Csak a módosított kulcsok kerülnek mentésre. Ez azt jelenti, hogy nem készül „másolat” a Windows vagy rendszerszintű rendszerleíróról, hanem a felhasználó saját Wine-könyvtárában készül egy kisméretű fájl, amelyben csak a módosított rendszerleírókulcsok kerülnek tárolásra.

```
"SaveOnlyUpdatedKeys" = "Y"
```

14. [Tweak.Layout]

A kiválasztott ablak külső megjelenésének beállítására szolgál. Három beállítás használható: a Windows 3.1 (ez az alapértelmezett), a Windows 95 és a Windows 98.

```
"WineLook" = "Win95"
```

15. [Console]

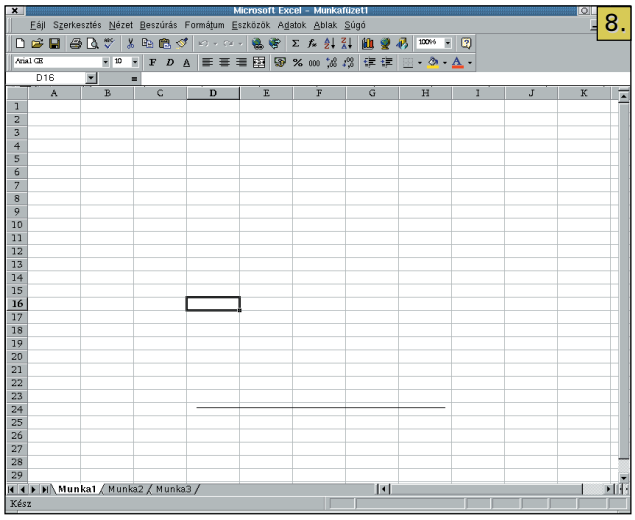
A konzolt alkalmazó programok számára fontosak az alábbi beállítások:

Drivers=tty	Használandó meghajtóeszköz.
XtermProg=nxterm	Használandó terminálprogram.
InitialRows=25	A terminál sorainak száma.
InitialColumns=80	A terminál oszlopainak száma.
TerminalType=nxterm	Termináltípus megadása.

Fentiek alapján elvégezhetjük a saját ízlésünknek és a futtatandó programnak megfelelő Wine-beállításokat. Ha jobban megnézzük, akkor láthatjuk, hogy vannak olyan beállítások, amelyeken kezdőként csak igen sok próbálkozás után tudunk átvergődni és eredményt elérni. Szerencsére a *WineSetupTk* nevű varázsló segítségével mindezt viszonylag gyorsan elvégezhetjük.

A következőkben próbáljunk meg az említett programocska által létrehozott Wine-beállítás segítségével Microsoft Office 97-et telepíteni. Fontos, hogy ne legyen a gépünkön sem FAT-típusú fájlrendszer, sem pedig telepített Windows operációs rendszer.

Az első lépés a WineSetupTk beszerzése:
➲http://www.codeweavers.com/winesetuptk.shtml. A program magyarított változata megtalálható a CD-mellékleten. Telepítéséhez, mint ahogy a neve is sugallja, a tcl és a tk programcsomagok telepítése szükséges. A `/usr/bin/ws` parancs kiadásával indítható.



8.

Az első párbeszédablakban be lehet állítani, hova mentse a program a Wine-beállításokat tartalmazó config fájlt (*1. ábra*).

A következő ablakban (*2. ábra*), ha minden rendben zajlott, a program önműködően megtalálja a telepített Windows alapkönyvtárát. Ha ilyen nincs, mint ahogy nálunk sincs, úgy felajánl egyet. Fogadjuk el és lépjünk tovább!

A következő párbeszédablakban (*3. ábra*) grafikus környezetben találkozhatunk a config fájl tartalmának magyarázata során már ismertetett beállításokkal.

Mindenki a telepítendő programjaitól függően állíthatja be a megjelenítést. A *Megjelenés* részben a Windows 95-ös felület használatát (a Windows 98-as mód nemigen tér el a Windows 95-től) ajánlom, illetve az *Ablak mód* beállításnál az *Irányított mód*-ot (így ugyanúgy tudjuk kezelni az Excel ablakot, mint a szokványos X-es alkalmazásoknál).

Az *Eszközök*-nél (*4. ábra*) elvileg a program önműködően megtalálja a Windows-könyvtárat és annak beállításait, ha mégsem, akkor módosítsuk azt.

Az *Útvonalak* párbeszédablakban (*5. ábra*) a Windows operációs rendszernek megfelelő beállításokat találjuk. Ezeket nyugodtan elfogadhatjuk és továbbléphetünk.

A *Megjelenés* párbeszédablakban az X11-gyel kapcsolatos beállítások szerepelnek, ezeket szintén nem kell módosítanunk.

A DLL-ek párbeszédablak tartalmaz egy alapbeállítást, ez az esetek igen nagy százalékában működőképesnek bizonyult. A betöltött DLL-ek megváltoztatását azonban csak gyakorlott Windows-felhasználóknak ajánlom.

A sorrendben következő *Portok* párbeszédablakban az eleve kitöltött beállításokat módosíthatjuk.

A *Registry* párbeszédablakban (*6. ábra*) a saját rendszerleíró fájl használatát (beolvasás, írás), illetve a frissített kulcsok beállítását választottam ki, mivel egyedüli felhasználó vagyok a gépemen, és nincs telepített Windows a gépen.

A beállítások végrehajtása és az Office 97 korong beszerzése után a parancssorban adjuk ki a következő parancsot:

```
wine ./SETUP.EXE
```

(természetesen számít a kis-, illetve nagybetű).

Ha minden jól megy, Office-unk telepítése (*7. ábra*) hibajelzés nélkül zajlik egészen az utolsó pontig, ahol – mivel nincs igazi user.dat, illetve system.dat fájlunk – a bejelentkezésnél hibát fog jelezni. Ha van ilyenünk, akkor a hibaüzenet elmarad, de ha ez már működő Windows része volt, akkor annak minden jellemzőjét tartalmazza, és ez nekünk sajnos nem felel meg, ugyanis nem rendelkezünk az összes Windows-eszközzel.

Telepítés során válasszuk az egyéni telepítést, és figyeljünk arra, hogy az Access- és a Windows-rendszerhez kapcsolódó elemek az alaprendszer hiánya miatt a későbbiekben gondot okozhatnak (például *Indítópult*), ezért ezeket ne telepítsük.

A „sikeres” telepítés után már csak el kell indítanunk – például az Excelt – az alábbi paranccsal:

```
cd /home/tothb1/.wine/fake_windows/Program
        ➥Files/Microsoft Office/Office
wine -language hu ./EXCEL.EXE
```

Az elindított Excel (*8. ábra*) négy hiba kivételével megfelelően működött. Többszöri próbálkozás után az alábbi hibák maradtak meg:

- Magyar Office-ról lévén szó, hiányolja a Tahoma betűkészletet, ez azonban a betűkészletek helyes beállításával orvosolható.
- A MSCREATE.DIR könyvtárat az Office szerette volna kezelni, de mivel nem engedélyeztem a közvetett könyvtárhivatkozások használatát, ez a hiba minden indításkor felbukkan.
- A *Fájl/Új dokumentum létrehozása* menüpont használatakor az Excel fagyott, viszont ha elindítottam az Excelt, és az önműködően létrejövő üres dokumentumot szerkesztettem és mentettem el, nem volt semmilyen nehézségem.
- A betűkészletek nem igazán működőképesek.

Néhány jó tanács

- A wine-os programokat első indításkor mindig valamilyen grafikus terminálról indítsuk, így figyelemmel tudjuk kísérni a hibaüzeneteket.
- Mindig legyen kéznél valamilyen feladatkezelő: ha valami esetleg megsérülne, ne okozzon gondot a törlése.

Összességében a Wine jó választási lehetőséget kínál azok számára, akik szeretnének úgy áttérni Linux-rendszerre, hogy mellette régi-régi Windows-programjaikat is használni kívánják. Természetesen ez a felület nem biztosít százszázalékos biztonságot, ezért kérek mindenkit, aki telepíti a WineSetupTk programot, figyelmesen olvassa el a leírást.

Tóth Béla (tothb1@freemail.hu)
Nős, két gyermek büszke atyja. Dolgozott földmérőként, majd térinformatikus szakmérnöki képesítést szerzett. Egyaránt otthonosan mozog a CAD és a térinformatikai programokban, valamint a DOS- és Windows-alkalmazásokban. Legkedveltebb elfoglaltsága már két és fél éve a Linux.

PoV-Ray ismeretek (1. rész)

Cikksorozatunkban a PoV-Ray sugárkövető programot mutatjuk be.

Ebben a részben áttekintjük a PoV-Ray főbb szolgáltatásait, megismerkedünk a legkisebb gépkövetelményekkel, elkészítjük az első egyszerű jelenetünket, majd összefoglaljuk a kamera és a fényforrások meghatározásának lehetőségeit.

Talán azzal illene kezdenem, hogy mi is a PoV-Ray. Szabadon használható program, mely igen sok szolgáltatással rendelkezik. Fotószerű képeket készíthetünk a sugárkövetés eljárásának felhasználásával. A PoV-Ray a *David K. Buck* és *Aaron A. Collins* által készített DKBTrace 2.12-es változatán alapul, melyet a szerzők szabadidejükben alkottak meg.

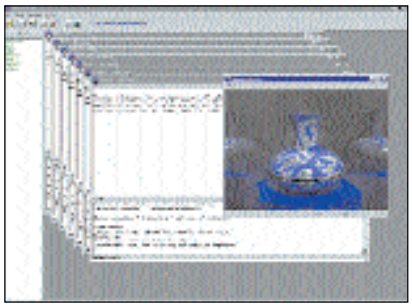
A programmal létrehozott jeleneteket szövegszerkesztővel fogjuk elkészíteni, a jelenetleíró nyelv írásmódja hasonló a C nyelvben megszokottéhoz. Természetesen nem kell mindent kézzel beírnunk, mert számos előre meghatározott alakzattal dolgozhatunk, és amit egyszer már megalkottunk, azt a későbbiekben felhasználási helyén egyszerűen beilleszthetjük. Alakzatok mellett színállandókat és előre meghatározott mintázatokat is tartalmaz a PoV-Ray-csomag, továbbá a program képes kétdimenziós képek alapján domborzat megjelenítésére is, ehhez látványos köd-, szivárvány- és atmoszférahatásokat alkalmazhatunk. A modellek szerkezetét CSG (Constructive Solid Geometry)-módszerrel is meghatározhatjuk, ami nagymértékben megkönnyíti a bonyolultabb modellek elkészítését. Ebben az esetben az egyszerű alakzatokból (gömb, kocka, tórusz stb.) logikai műveletekkel állítjuk elő a számunkra megfelelő formát. Ha például egy domború lencsét szeretnénk meghatározni, akkor ezt úgy is megtehetjük, hogy gömböt metszünk el egy kockával, majd az eredményül kapott testet lemásoljuk, és a másolatot szembefordítjuk az eredetivel. Megfelelő elhelyezés után már készen is van a domború lencse. A program lehetőséget ad számos egyedi, különleges hatás létrehozására is, például: tűz, felhőzet, füst, valamint poros levegőben láthatóvá váló fénysugár megjelenítése. Mintázatként a sokfajta beépített mintázat (például: márvány, leopárd mintázat, bump mapping, színátmenet, pepita mintázat) mellett használhatunk képeket is.

A kimeneti fájlformátumok között a következők találhatók meg: PNG, TGA és BMP, valamint Machintosh felületen .PIC formátumban is tárolhatjuk a képeket. IBM-PC felületen a kiszámolt képeket 15- és 24-bites színmélységben jeleníthetjük meg. Az elkészült képeket legfeljebb 48-bites színmélységben tárolhatjuk.

Mindezek mellett a program számos felületen elérhető, köztük Linuxon, Amigán, MS-DOS-on, Sun operációs rendszeren, Windowson és Apple Machintoshon.

➲ www.provray.org

➲ www.xlcus.co.uk/povray/island/pan.html



➲ www.xlcus.co.uk/povray/

Erőforrásigénye – a leírás szerint – a mai normákhoz képest igen szerény, a legnagyobb igényeket Windows alatt futtatva tapasztalhatjuk, de itt is megelégedhetünk 16 MB memóriával és egy 486 DX-es processzorral. Linuxon megelégszik 386-os processzorral, 4 MB memóriával és 10 megányi szabad tárhellyel. A gyorsabb futtatáshoz ajánlott Pentium processzor és 32 MB memória, de természetesen minél nagyobb teljesítményű a gépünk, annál hamarabb várhatunk eredményt.

Linux alatt a PoV-Ray minden hivatalos változatnak részét képezi, ha azonban valakinek mégsem lenne meg, akkor a ➲ http://www.povray.org címen elérheti, a Download csoportban letöltheti. Ennyi bevezető után lássunk egy a programmal készített képet, utána még közelebbről megismerkedünk a lehetőségekkel.

Első képünk elkészítéséhez tekintsünk át néhány alapfogalmat, amelyek megértése a további munkához és a példákhoz is elengedhetetlen. Legfontosabb, hogy a PoV-Ray koordinátarendszerét értsük meg. Ez „balkezes” koordinátarendszer, az Y tengely mutat felfelé. Leginkább úgy szemléltethetnénk, ha bal kezünk mutatóujjával mutatnánk felfelé és a középső-, hüvelykujjunk segítségével derékszögű koordinátarendszert képeznénk. Ilyen helyzetben a Z tengelyt jelképező gyűrűsujjunk a nézőponttól távolodó irányba mutat.

A jelenetek elkészítéséhez a Nedit szövegszerkesztőt használtam kitűnő makrózási lehetőségei következtében, felhasználva a lemezmellékleten található előre elkészített, dot.nedit fájlban található makrókat is. Ezt az állományt felhasználói könyvtárunkba kell bemásolnunk .nedit néven.

A következő lista alapján megérthetjük a PoV-Ray jelenetleíró nyelvének alapvető írásmódját és a jelenet felépítését.

1. lista A PoV-Ray alapvető írásmódja és a jelenet felépítése

```
#include "colors.inc"

global_settings
{
        ambient_light 0.7
}

// --------------------------- Kamera
camera
{
  location  <0.0, 0.5, -4.0>
  look_at   <0.0, 0.7,  0.0>
}

// --------------------------- Fenyforras

light_source {
        <4,4,-4>
        1.2*White
}

// ---------------------------- Alaplap
plane {
        y, -1
        pigment {
                checker pigment {Black},
                     pigment {White}
                scale <2,2,2>
                }
}

// ---------------------------- Hatso fal
plane {
        -z, -20

        pigment {
                image_map { gif "gekko.gif" }
                scale 15.9
                translate y*-1.5
                }

}


// --------------------------- Egy gomb
sphere
{
  0.0, 1
  texture {
          pigment {
             radial
             color_map {
              [0.0 rgbt <0, 0.3, 0, 0.2>]
              [0.5 rgbt <0.2, 0.2, 0.3, 0.2>]
              [1.0 rgbt <0.3,0,0,0.2>]
             }
             frequency 8
             turbulence 0.9
          }
          finish {
                  specular 0.6
                  ambient 0.6
                 }
          }
  rotate x*90
}
```


Amint azt a lista alapján is láthatjuk, a jelenet tartalmaz egy kamerát, ami meghatározza a megjelenítendő kép értékeit (oldalarányait), a nézőpontunkat és a kamerával használt objektív értékeit. Minden jelenet magába foglal legalább egy fényforrást, bizonyos esetekben ez megegyezik a környezeti szórt fénnyel – ez nem kézzelfogható fényforrás, de megvilágítja a jelenetet. Amennyiben egy jelenetben semmilyen fényforrást nem határozunk meg, akkor a PoV-Ray alapértelmezett környezeti megvilágítással számítja ki a képet.

Most pedig lássuk a kamera meghatározását és fontosabb értékeinek értelmezését! Az írásmódot a PoV-Ray leírásában található formában adom meg. Vektort a programban <x, y, z> alakban kell megadni, a lebegőpontos és egész értékek megadása pedig azonos a más programnyelvekben megszokottal. A programba beépítve találhatjuk meg az egységvektorokat mindhárom irányban, tehát ha valamilyen helyet vagy irányt nem akarunk teljes koordinátákkal megadni, akkor ezeket az egységvektorokat használhatjuk a megfelelő szorzótényezővel. Például: a <5.2,0,0> vektor egyenértékű a következővel: x*5.2.

Elsőként vegyük sorra a használható objektívtípusokat:

Perspective

Ez a klasszikus objektív, amit a fényképezőgépekben is alkalmazunk. A vízszintes látószöget egyrészt a „direction” vektor és a „right” vektor hosszúságának aránya határozza meg, másrészt az „angle” érték segítségével is meghatározható. Ez utóbbi használata egyszerűbb. A látószög értéke 0 és 180 fok között lehet.

Orthographic

Ebben az esetben párhuzamos vetítést használunk, a kép oldalarányait pedig az „right” és az „up” vektorok hosszúságának aránya szabja meg. Például a klasszikus 4:3 arányú megjelenítéshez „up y” és „right 1.33*x” vektorok megadásával juthatunk.

Fisheye

Halszemoptika, gömbszerű vetítéssel. Ez a típus is ismerős a fényképészet területéről, a látószöget az „angle” kulcsszóval adjuk meg. 180 fokos látószöggel a szokásos halszemoptikának megfelelő hatást érhetünk el, míg 360 foknál minden körülöttünk lévő tárgyat láthatunk. Ezen optikával kör vagy ellipszis alakú képet kapunk eredményül.

Ultra_wide_angle

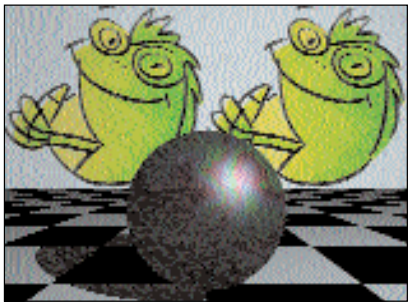
Hasonlít a halszemoptikához azzal a különbséggel, hogy mindenképpen téglalap (vagy négyzet) alakú képhez jutunk. A látószöget szintén az „angle” kulcsszó segítségével határozhatjuk meg.

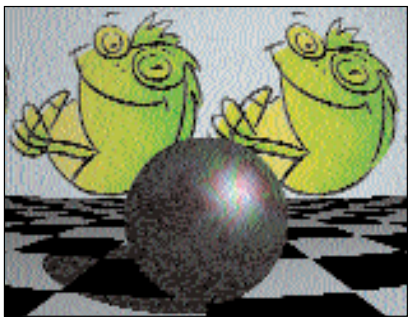
Omnimax

Ez a vetítési mód egy 180 fokos halszemoptikánál használtnak felel meg, de a vízszintes látószög kisebb és nem változtatható az „angle” kulcsszóval sem. Ilyen képeket (filmeket) használnak a kupola alakú vetítővászonnal rendelkező úgynevezett Omnimax filmszínházakban.

Panoramic

Panorámaképek előállításához használható, mert kiküszöböli azt a perspektív vetítésnél tapasztalható hibát, amit a 180 fokot megközelítő látószögeknél tapasztalunk. Itt szintén alkalmazható az „angle” érték.

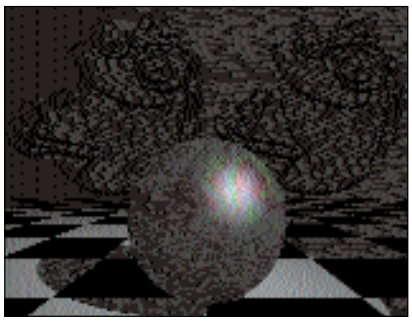


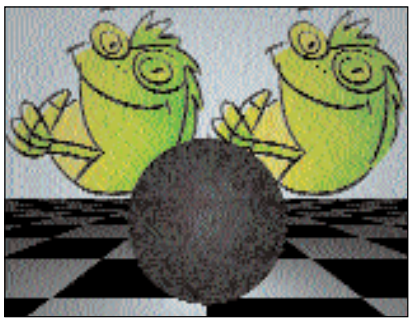


Cylinder

Ebben az esetben a jelenetet egy henger felületére képezzük le, a megadott valós érték 1, 2, 3 vagy 4 lehet. 1-es értéknél a hengerfelület függőleges állású, 2-es értéknél vízszintes, 3-as értékkel szintén függőleges irányú felületet adunk meg, de itt a nézőpont helyzete a henger tengelyén mozog, míg a 4-es érték mozgó nézőpontot és vízszintes tengelyű hengert határoz meg. A kamera látószögére vonatkozóan csak a perspektív vetítésnél van megkötés (0–180 fok közé kell esnie). Különleges hatásokat érhetünk el, ha 360 foknál nagyobb látószöget adunk meg, ennek hatására a képen a jelenet többször fog megjelenni. Hogy pontosan hányszor láthatunk ismétlődést, az a kamera egyéb értékeitől is függ, így érdekes lehetőség nyílik az ismerkedésre, próbálgatásra.

A kamera helyzetét a „location” kulcsszóval adhatjuk meg, míg a célpont helyét, ahová majd kameránk néz, a „look_at” kulcsszóval. Mindkettő egy-egy vektort vár értékként. Az „up” és „right” vektorok szerepe többnyire az, hogy meghatározzák a kamera irányát, de ezt sokkal könnyebben megtehetjük a „direction” és a „look_at” értékek használatával. Ennek ellenére ez a két vektor mégis fontos, mert ezekkel adjuk meg a kép oldalarányait. Az egyes vetítési típusoknál más-más értelmezéssel kell számolnunk. Perspektív, panoráma és nagy látószögű (ultra_wide_angle) objektív használatakor a vektorok meghatározzák a kép arányait és a „direction” vektorral együtt a látószöget is. A „direction” helyett ajánlott az „angle” használata. Párhuzamos vetítés esetén szintén e két vektor arányának megfelelő képet kapunk, de itt a „direction” értékét a program nem veszi figyelembe, mint ahogyan a halszemoptikánál sem. Hengerfelületre való vetítést alkalmazva, annak 1-es és 3-as típusánál a henger tengelyének iránya megegyezik az „up” vektor által kijelölt iránnyal, és a 3-as típusnál ennek a vektornak a hosszúsága meghatározza azt is, hogy a tér mely koordinátái között kell a képet kialakítani. Például „up 4*y” azt jelenti, hogy csak az y=-2 és y=2





közötti pontok lesznek láthatók. Ezeknél a típusoknál minden vetítősugár merőleges lesz az y tengelyre. A 2-es és 4-es hengeres vetítési típusnál a hengerfelület tengelye párhuzamos a „right” vektorral és minden vetítősugár merőleges az x tengelyre.

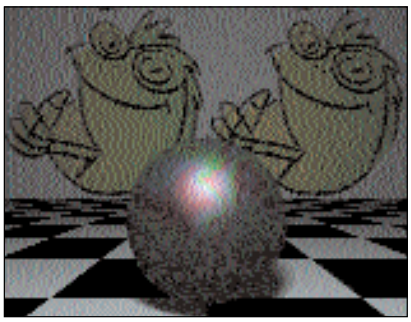
Nem szabad elfelejtenünk, hogy a „right”, az „up” és a „direction” vektoroknak egymásra merőlegesnek kell lenniük, hogy elkerüljük a torzulásokat. Ha ez a helyzet nem áll fenn, a számítás elindításakor egy figyelmeztető üzenetet kapunk.

Az „angle” értéket, mint az a korábbiakból is kitűnik, a látószög meghatározásához használjuk, így most csak a pontosítás kedvéért adom meg az összefüggést a „right”, a „direction” vektorok hosszúsága és a látószög között:

```
látószög= 2 * arctan(0.5 *
        ➥right_hosszúsága /
        ➥direction_hosszúsága)
```

A PoV-Ray képes mélységélesség szimulálására is, ami azt jelenti, hogy a kamera fókuszában lévő terület élesebb lesz, mint a fókuszon kívül eső részek. Ennek pontosabb meghatározására alkalmas az „aperture” (rekesz) kulcsszó: nagyobb értékek esetén az élesebb terület kisebb lesz, míg kisebb rekeszt használva az élesebb terület mérete megnő. Az éles terület középpontját a „focal_point” (fókuszpont) vektor segítségével adjuk meg. Ezt célszerű a jelenet fő témájára állítani. Alapértelmezésben a PoV-Ray nem számol mélységélességgel, tehát ezt nekünk kell bekapcsolni az „aperture” és a „blur_samples” értékek megfelelő megválasztásával. A „blur_samples” értéke határozza meg azt, hogy az életlen területek elmosásához a program hány környező fénysugár átlagát vegye figyelembe.

Miután megismerkedtünk a kamera meghatározásával és használatával, tekintsük át a négy legfontosabb fényforrás általános formáját és a különböző értékek használatát. A fényforrás helyét egy vektorral adjuk meg, a színét pedig egy „rgb” kulcsszó után szereplő három 0-1-ig terjedő valós értékkel (piros, zöld és kék összetevők mértéke).



Például: color <rgb 0.3, 0.3, 0.3>

A „shadowless” kulcsszó alkalmazásával az adott fényforrás nem képez árnyékokat. Az „adaptive” kulcsszó után szereplő egész érték a mintatúlvételezést befolyásolja, ennek eredményeképpen az éles árnyékok elmosódnak. Az 1-es értéknél 4, a 2-es értéknél 9, a 3-as értéknél 25, míg 4-es értékű mintatúlvételezéssel a program 81 környező pont átlagaként fogja meghatározni egy-egy pont színezetét.

Mivel a fényforrásoknak alapértelmezésben nincsen megjelenési formájuk, szükség lehet arra, hogy a fényforrást megfeleltessük valamilyen tárgynak, vagyis „formába

2. lista A kamera általános alakja a fontosabb értékekkel

```
camera {
    [ perspective | orthographic |
      fisheye | ultra_wide_angle |
      omnimax | panoramic |
      cylinder FLOAT ]
    location <VEKTOR>
    look_at <VEKTOR>
    right <VEKTOR>
    up <VEKTOR>
    direction <VEKTOR>
    right <VEKTOR>
    angle VALÓS
    blur_samples VALÓS
    aperture VALÓS
    focal_point <VEKTOR>
  }
```

3. lista A pontszerű fényforrás általános formája

```
  light_source {
    <F NYFORRAS_HELYE>
    color <SZ"N>
    [ shadowless ]
    [ adaptive EGESZ ]
    [ looks_like { OBJEKTUM } ]
    [ atmospheric_attenuation BOOL ]
  }
```

4. lista A fényszóró fényét egy pontra irányítjuk

```
light_source {
       <0,0,0>
       color < rgb 1, 0.5, 0.5 >
       spotlight
       radius 10
       falloff 20
       fade_distance 8
       fade_power 1
       point_at <1,2,1>
}
```

5. lista Jól látható a téglalap alakú területet megvilágító fényforrás használata

```
light_source {
        <0,4,-4>
        1.2*White
       area_light x,y,4,4
       fade_distance 5
       fade_power 1
}
```

öntsük". Ilyen eset lehet például egy gyertya lángjának a megjelenítése. A „looks_like" kulcsszó pontosan ezt a célt szolgálja, a megadott objektum a fényforrás alakjaként fog megjelenni.
Az objektumot előre kell meghatározni és a „no_shadow" kulcsszó használatával ki kell zárni az árnyékot vető tárgyak közül azért, hogy az objektum árnyéka ne takarja el a jelenet többi részét. Erre láthatunk példát a lemezmellékleten található tutor1_5.pov fájlban.
A PoV-Ray alapértelmezésben nem számol azzal a ténnyel, hogy a fény szétszóródik a ködben és az atmoszférán áthaladva veszít erősségéből. Az „atmospheric_attenuation" bekapcsolt állapotában (atmospheric_attenuation on) azonban ezt a szolgáltatását is használhatjuk, látványos hatásokat érve el vele.
A következő fontos fényforrás a spotlámpa. Ez a valóságban a fényszórónak felel meg. A fentieken felül további öt értéke lehet. Meghatározásához a szín beállítása után meg kell adnunk a „spotlight" kulcsszót, majd a fényszóró által állandó erősséggel megvilágított szögtartományt a „radius" kulcsszó után. Lehetőségként használható az is, hogy az egyenletes megvilágításon túl még hány fokos tartományban világítson a lámpa. Ezt a „falloff" kulcsszóval és a hozzá tartozó valós értékkel határozhatjuk meg. A „fade_distance" az egyenletes megvilágítás fényforrástól való távolságára van hatással, vagyis minél nagyobb ez az érték, annál messzebbre világíthatunk az erősség csökkenése nélkül. Az erősség csökkenésének módját adja meg a „fade_power" érték. Ha ez 1, akkor a fény erőssége lineárisan csökken, 2-es értéknél pedig négyzetesen. A fényszóró fényét pontosan egy pontra irányíthatjuk a „point_at" kulcsszó után megadott vektor segítségével.
Ugyanilyen értékkel használhatunk henger alakú fénycsóvát is a megvilágításra, ekkor a „spotlight" kulcsszó helyett a „cylinder"-t kell alkalmaznunk. Míg a fényszóró kúp alakú térrészt világít meg, a hengeres fényforrás fénye henger alakú térrészben érzékelhető.
Az első rész utolsó témájaként egy olyan fényforrást ismertetek, amely egyaránt használható nagyobb területek, valamint a környezeti megvilágítás megvalósítására.
A környezeti szórt fény kiindulási pontja nem mindig határozható meg pontosan, előfordulhat, hogy pontosan ilyen fényforrásra van szükségünk. Itt tehát az „area_light"-ról lesz szó, ennek alapvető értékei megegyeznek a spotlámpánál leírtakkal, kivéve a „radius" és a „falloff" kulcsszavakat, melyek itt nem értelmezhetők. Mivel ez a típus téglalap alakú területet világít meg, kiegészítésképpen megadható a téglalap oldalainak aránya és a fényforrás két tengelye.
Ha jobban megfigyeljük az egyes fényforrások hatására kialakuló árnyékokat, akkor észrevesszük, hogy ezzel a típussal érhetünk el igazán valósághű árnyékokat, itt ugyanis már nem jelentkezik a pontszerű fénynél tapasztalható éles szélű árnyék.
Ezzel elérkeztünk sorozatunk első részének végéhez, remélem, hogy ezzel a rövid ízelítővel sikerült felkelteni az érdeklődést e nagyszerű program iránt. A további fejezetekben tárgyaljuk az alapvető testeket, az anyagok és mintázatok alkalmazását, az összetettebb alakzatok modellezését, az animáció lehetőségeit, végül néhány hasznos segédeszközt is bemutatok, amelyekkel megkönnyíthetjük munkánkat.
A lemezmellékleten található példák segítik a fentiek megértését, kiszámoltatásukhoz a következő parancssori értékekkel indítottam a PoV-Ray-t:
#povray -i*példafájl_neve*.pov +w*Szélesség* +h*Magasság* +L/usr/lib/povray3/include
Az utolsó érték (+L*könyvtárnev*) azt az elérési utat adja a program tudtára, ahol a beillesztendő fájlok találhatók. Erre az útvonalra a colors.inc miatt van szükség.
Várom az érdeklődők kérdéseit a következő levélcímre ➲ dzooli@freemail.hu.
A listák megtalálhatók a 14. CD-melléklet Magazin/Cikkekhez könyvtárában.

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.

Figyelem! Lépnünk kell!

Los Angelesben élek, ahol még él a Hollywood 10 és a Blacklist emléke. Még Sundance-napjaimból ismerek számos olyan írót, akik feketelistára kerültek, és tudom, a HUAC-vel (➲ http://www.english.upenn.edu/~afilreis/50s/blacklist.html) kapcsolatos élményeik megkeserítették vagy sokszor megtörték őket. Rendkívül tehetséges emberek voltak, akiknek életét a „Nagy Hazugság” tette tönkre.

A Microsoft emberének, *Jim Allchin*-nak a szavaival ugyan nagyon könnyű tréfát űzni – van is miért viccelődni ezekkel –, ugyanakkor egyfajta jelzésértékkel is bírnak. Ugyancsak könnyű elutasítani azt a látszólag alaptalan és nevetséges vádat, hogy az ingyenes programok nem felelnek meg az amerikai szellemnek, de azt hiszem, nem elég kikacagnunk ezt a véleményt. Az a lehetőség is mindig fennállhat, hogy néhány alulművelt és meggondolatlan ember szó szerint értelmezi Allchin kijelentéseit, különösen akkor, ha ezek az emberek köztisztviselői állást töltenek be valahol.

A GNU/Linux cégeknek, illetve azoknak a magánszemélyeknek, akiknek szenvedélyük a Linux, hallatniuk kell a hangjukat a Capitol Hillen, valamint a helyi és az állami törvényhozásban. A Linux International (➲ http://www.li.org/) nemrég változtatta meg a saját szabályzatát oly módon, hogy így már lehetőséget teremt szakmai csoportok létrehozására és a lobbizásra is. A SIIA (➲ http://www.siia.net/), a programozói ipar legrégebbi és legnagyobb kereskedelmi társasága évi rendes gyűlésén egy Open Source Division találkozót tartott (erről bővebben lásd lentebb) azzal az ötlettel, hogy egy tervezetet hozzanak létre a törvényhozókkal való kapcsolatfelvétel céljából.

Az olyan könnyű célpontokra tüzelni, mint Allchin, hatalmas sikerélménnyel járhat, mégis elővigyázatosan kell viselkednünk, hiszen az olyan szervezetek kereteinek kihasználása, mint például a Linux International vagy a SIIA, közelebbi betekintést enged a dolgokba.

A Linux Professional Institute

A Linux Professional Institute (➲ http://www.lpi.org/) egy olyan szabványosított, nemzetközi programot tervez, amely alkalmas az egyéni Linux-tudás mérésére és igazolására. A programok szigorú teljesítménymérési folyamaton mennek keresztül, így biztosítható, hogy megfeleljenek a Linux-szakértők, illetve az őket alkalmazó, vagy velük kapcsolatba kerülő szervezetek elvárásainak. Pillanatnyilag az első szint érhető el, mely az alapvető rendszerfelügyeleti tudást ellenőrzi; jelenleg pedig a második és harmadik szinten dolgoznak. Az LPI megőrzi a gyártóktól való függetlenséget, a tesztelés pedig világszerte azonos módon, angol nyelven folyik a Virtual University Enterprise (➲ http://www.vue.com/) segítségével.

„A Linux Professional Institute fenntartja a vizsgákra való felkészülési mód szabad megválasztását” – állítja *Dan York*, az LPI elnöke. Honlapjukon megtalálható azoknak az ajánlott könyveknek, képzési központoknak és tanfolyamoknak a listája, amelyek segítséget nyújthatnak a felkészülésben. Az LPI a fontosabb gyakorlati ipari szabványok és pszichometriai alapok segítségével dolgozta ki vizsgáit. Más szavakkal élve, egy több mint 1400 embert felölelő munkahelyi felmérésben kérdezte meg a résztvevők véleményét az első szint tesztjéről. A felmérés megpróbálja számba venni azt a tudásanyagot, amelyre az egyes szakmai szinteken a szakértőknek szükségük van, illetve azokat feladatokat, amelyeket el kell tudniuk végezni. A cél York szerint az, hogy a valós élet által is igazolható célkitűzéseket hozzanak létre. A tesztek jogi védhetősége úgyszintén fontos kérdést jelent.

Az LPI jelenleg a második szint tesztjét fejleszti, és máris 200 szakértőt kérdezett meg. A feladat hosszadalmas és rengeteg időt vesz igénybe, de egy megbízható tesztelési alap elkészítése megéri az időt és az energiát – vélekedik York.

Az LPI fel tudná használni a Linux-szakértők segítségét a vizsgázási folyamatban. Olyan emberekre van szükségük, akik el tudnák készíteni, le tudnák ellenőrizni, és le tudnák fordítani a második szint kérdéseit; illetve el tudnák látni az első szint kérdéseinek folyamatos karbantartását, hogy azok a mindenkori fejlesztések eredményeit tükrözzék. Mindemellett segítségre lenne szükségük az anyagiak és a marketing területén is. Mivel költségvetésük nem tartalmaz marketingkiadásokat, a minősítési folyamat résztvevői csak szóbeszéd útján szereznek tudomást az LPI-ről. A cégek oldaláról is meggondolandó kezdeményezés lenne, ha alkalmazottaik néhány órát az LPI-kezdeményezésnek szentelnének. További felvilágosítást a ➲ http://www.lpi.org/ oldalra ellátogatva, majd a Getting Involved hivatkozásra kattintva találhatunk.

A SIIA Open Source Division találkozót tart San Diegóban

A Software Information Industry Association (SIIA) San Diegó-i éves tanácskozásán megtartotta az Open Source Division szakmai csoport alakuló ülését. A napirenden olyan kérdések szerepeltek, mint a nyílt számítástechnikai modell jövője vagy a szabad programok piaci elfogadottságának növelésére alkalmas eszközök megtárgyalása. Szintén szóba kerültek azok a közérdekű fejlesztések, amelyek a nyílt forrású programokat fejlesztő cégeket érintik, valamint elkészült a jövőbeli lépéseket felvázoló ütemterv is. A RedHat volt az első olyan szabad forrású programokkal foglalkozó cég, amely csatlakozott a SIIA-hoz.

A SIIA a programozó cégek nemzetközi kereskedelmi szervezete. A közelmúlt trösztellenes eljárása során a SIIA olyan beadványt nyújtott be, amely a kormány oldalán sorakoztatott fel érveket. Tette mindezt annak ellenére, hogy a SIIA igazgatóságában a Microsoft egyik vezetője is helyet foglalt. A beadvány benyújtását követően a Microsoft néhány napon belül lemondott tagságáról. *Ken Walsh*, a SIIA elnöke szerint fontosabb volt a helyes út követése az igazgatósági tagok túlnyomó többségének egyetértésével, mint a meghajlás egyetlen tag akarata előtt.

Gnome Alapítvány

A Gnome alapítvány (➲ http://www.gnome.org/) kiadta a Gnome 1.4-es változatát, egy kiforrott, megbízható asztali környezetet. A kiadás nemcsak számos apró díszítőelemet tartalmaz, hanem több kiegészítő csomagot is, köztük az Eazel fájlkezelő rendszerét, a Nautilust. A kiadás nagy előrelépést jelent a Gnome számára, amely mind megbízhatóbb, és mind nagyobb teljesítményt nyújt. A Gnome 1.4-es változatáról a *Help* fülről kiindulva bőséges leírást található. Nagyszámú tükröoldalról tölthető le, melyek listája a Ximian oldalán (➲ http://www.ximian.com/desktop/download/.php3) található meg, valamint számos Linux-csomag részeként is elérhető.

Leslie Proctor
(lesproctor@yahoo.com)
a .org közösség tevékeny résztvevője, világszerte több .org-szervezet tanácsadójaként dolgozott.

JavaBeans

A babok használata a programozók minden rétege számára leegyszerűsíti a JSP-oldalakkal való munkát.

A Kovácsműhely legutóbbi két írásában a Jakarta-Tomcat nyílt forráskódú servlettel és az Apache Software Foundation által támogatott JavaServer Pages (JSP) motorral foglalkoztunk. Mint láthattuk, JSP-hez servletet írni nem is olyan nehéz és nem is annyira időigényes feladat. Az egyetlen nehézséget talán a kiszolgálóoldali Java használatának elsajátítása okozhatja. A servletek ugyan a Java-programokból elérhető kifejezőerő és hatékonyság teljes kínálatát nyújtják, de arra kényszerítenek, hogy viszonylag alacsony szinten gondolkozzunk. Valahányszor csak egy HTML formátumú szöveget szeretnénk a felhasználó böngészőjére küldeni – ami, ugye, meglehetősen gyakran megesik –, a PrintWriter objektumot kell használnunk, ami a HTTP válasz-objektumhoz van rendelve:

```
PrintWriter out = response.getWriter();
out.println("<HTML><Body>This is illegal HTML
            </Body></HTML>");
```

Ebben az esetben a JSP-k siettek a segítségünkre – ezek feltételezik, hogy amit nem végrehajtható kódként jelöltünk, azt egy az egyben kell elküldeni a felhasználó böngészőjére. Ezáltal azonban rögvest új gond keletkezett: ha például a JSP egy relációs adatbázis-kezelőhöz szeretne csatlakozni, akkor Java-kódsorok tucatjait, esetleg százait szükséges beilleszteni.
A megoldás az, hogy a JSP-n kívüli kódcsomagokat készítünk, melynek tagfüggvényeit olyan formai követelményekkel hívhatjuk meg, amelyek inkább hasonlítanak a HTML-re, mint a Javára. Ezek a JavaBeansnek nevezett kódcsomagok megkönnyíthetik a JSP-kel való munkát – akár egy magasabb szinten dolgozó, tapasztalt programozó számára, akár egy tapasztalatlan programozónak, aki inkább az alkalmazhatóság előnyeit szeretné kihasználni.
Ebben a hónapban a JavaBeans birodalmában teszünk egy rövid körutat. Írunk néhány saját babot, beépítjük őket a JSP-be, és megvizsgálunk egy-két ehhez kapcsolódó hibát és csapdát.

Mi is az a bab?

A babok létrehozójának szemszögéből a JavaBeanek bizonyos feltételeknek megfelelő Java-osztályok. (Hamarosan megvizsgáljuk, mik is pontosan ezek a feltételek.)
A JSP-t írók számára viszont a babok olyan különleges tárolóosztályt jelentenek, amelyekben bizonyos adatokat tárolhatunk, illetve kaphatunk vissza. Az adatok egyes darabjait *tulajdonságoknak* nevezzük, amelyeket külön-külön beállíthatunk, illetve lekérdezhetünk. Ugyan nem minden tulajdonságot lehet lekérdezni vagy beállítani, a babok felhasználói felülete JSP-oldalról egységes és könnyen érthető.
Mivel a babok csak kötött műveletkészletet fogadnak el, létezik néhány különleges JSP-tag, amelyek segítségével használhatjuk őket. Ezeknek a tagoknak az alkalmazásával mérsékelhetjük a közvetlenül JSP-be helyezett Java-kód mennyiségét. Ez nem csupán azt jelenti, hogy csökken a zavar és átláthatóbbá válik a kód, hanem egyúttal a nem programozók számára is lehetővé teszi, hogy a babok erejét kihasználják anélkül, hogy meg kellene tanulniuk Javában programozni.

Nem szokatlan jelenség, hogy egy JSP-ben egyszerre több babot is felhasználnak, szükség szerint tárolva és visszaszedve a különféle tulajdonságokat. Például egy hálózati áruház bevásárlókosara használhat egy babot a leltár tárolásához, egy másikat a felhasználó kosarához, és még továbbiakat a felhasználó nyelvének követéséhez vagy a fizetés és kiszállítás beállításaihoz. Ezek a babok mind külön Java-osztályok, kezelésüket azonban különleges JSP-tagok végzik, amelyek a JSP-t író elől elrejtik sokrétűségüket.
Természetesen a babokat több honlapon (vagy egy honlap több részében) is fel lehet használni. Ha egyszer már létrehoztunk egy hasznos JavaBeant, amely érdekes képességekkel bír, mások is könnyedén behelyezhetik azt saját Java-osztály keresési utunkba (classpath), és máris JSP-ből élvezhetik ezek előnyeit.

Babok készítése

A bab létrehozásához először írnunk kell egy java.io.Serializable csatolófelületet megvalósító Java-osztályt, azaz olyan babot készítünk, amely képes lemezre írni és visszaállítani önmagát. Ha osztályunk mezői egyszerű Java-típusok, például egészek és karakterláncok, akkor a Serializable megvalósítás nem igazán fontos.
Az *1. lista* egy egyszerű babmegvalósítást mutat be. Ez a bab egyetlen példányváltozót (`userID`) és két tagfüggvényt tartalmaz. A `getUserID` tagfüggvény a `userID` pillanatnyi értékét adja vissza, míg a `setUserID` tagfüggvény e mező értékét állítja be. Mivel ezek a tagfüggvények a SSP-kben használatos formátumoknak megfelelnek, JSP-inkből felhasználhatjuk őket.
A saját Apache projekt Jakarta-Tomcat servlet, illetve JSP-rendszer 3.2-es változatát futtató rendszeremen Java-osztályaimat a $TOMCAT_HOME/classes könyvtárba kellett elhelyeznem. (A $TOMCAT_HOME egy környezeti változó, amely a Tomcat-telepítés helyére mutat; saját rendszeremen értéke /usr/java/jakarta-tomcat-3.2.1/). Ha ez a „classes” könyvtár létezik, akkor hozzáadódik a Tomcat `CLASSPATH` környezeti változójához, így az új osztályok számára kényelmesen használható helyet hoz létre.
Az osztály maga igen egyszerű, a különböző típusú tagfüggvények bemutatásához a következőket kell elkészítenünk:

1. A bab létrehozóját (constructor), amely nem kap egyetlen értéket sem, és egy vagy több mezőt állít be. A mi példánkban a SimpleBean létrehozó a `userID` változót állítja az alapértelmezés szerinti nullára.
2. Tulajdonságlekérdező tagfüggvényt, amely értéket ad vissza a hívónak. Akárcsak a bab létrehozója, a tulajdonságlekérdező tagfüggvény sem vár értéket.
3. A tulajdonságbeállító tagfüggvényt, amely egyetlen értéket vár (az új értéket), a hívónak azonban nem ad vissza semmit.

Ne feledjük, a babok ugyanolyan osztályok, mint a legtöbb Java-osztály, azaz mielőtt használnánk, újra kell fordítani őket. Továbbá a Tomcat servlettároló az újrafordított osztályokat nem tölti be önműködően. A legbiztosabb, ha minden babosztály újrafordítása után újraindítjuk a Tomcatet is.

1. lista SimpleBean.java

```
package il.co.lerner;

import java.util.*;
import java.lang.*;
import java.sql.*;

public class SimpleBean
             implements java.io.Serializable {

     private int userID;

     // ----------------------------------------
     // Constructor

     public SimpleBean () {
          // Beállítjuk a mezőnket
          userID = 0;
     }

     // ----------------------------------------
     // userID tulajdonság lekérdezése

     public int getUserID()
     {
          return userID;
     }

     // ----------------------------------------
     // userID tulajdonság megváltoztatása

     public void setUserID (int newID)
     {
          userID = newID;
          return;
     }
}
```

A bab használata

Most, hogy egyszerű babosztályunkat már elkészítettük, lássuk, miképpen használhatnánk a JSP-ből! A JSP három különleges, egyedi tagot ismer fel, amelyek mindig „jsp:" kezdetűek. Mielőtt folytatnánk, egy figyelmeztetés: ezek az egyedi tagok, melyek lehetővé teszik, hogy a JSP-ből JavaBeanekkel dolgozzunk, nem HTML-ben, hanem XML-ben íródtak! Míg a HTML lazán meghatározott szabvány, ahol nem mindig kötelező lezárni a tagokat, az XML sokkal szigorúbb. Minden nyitó `<tag>`-ot le kell zárni egy megfelelő `</tag>`-gal. Lezáró `</tag>` nélkül az XML értelmező-hibával kilép. A tag azonban saját magát is lezárhatja, ha a végére perjelet helyezünk a következők szerint: `<tag/>`.
Ez azt jelenti, hogy a JavaBeanst használó a JSP-ben – amely HTML-kimenetet készít – két kissé eltérő írásmódra kell figyelni. Egy idő után a két formai követelmény közti váltás teljesen természetessé válik. Ráadásul a JSP értelmező hibaüzeneteiből viszonylag könnyű meghatározni, hol felejtettük el kitenni a bezáró perjelet valamelyik JavaBean-tagra. Ennek ellenére az írásmódok ilyen keverése a kezdők számára elég őrjítő lehet, és elsajátításuk során jópár nehézségre van kilátásuk.
A JavaBean-osztályokhoz különleges, `<jsp:useBean/>` formátumú tagokat használunk. Ez a tag arról tájékoztatja a JSP-t, hogy keressen meg és töltsön be egy adott babot, majd a JSP-nkben készítsen

2. lista use-simple.jsp

```
<HTML> <Head><Title>SimpleBean
                    használata</Title></Head>
<jsp:useBean id="simple"
             class="il.co.lerner.SimpleBean"/>
<Body> <H1>SimpleBean használata</H1>
<P>Eredeti érték: <jsp:getProperty name="simple"
                   property="userID"/></P>
<jsp:setProperty name="simple" property="userID"
                 value="300"/>
<P>Új érték:<jsp:getProperty name="simple"
property="userID"/></P>
</Body> </HTML>
```

belőle egy példányt. A `<jsp:useBean/>` tag arra is lehetőséget nyújt, hogy a példánynak nevet adjunk, amit a későbbiek során felhasználhatunk. Lássuk például, miképpen tölthetnénk be imént készített SimpleBean-osztályunkat a JSP-ből:

```
<jsp:useBean id="simple"
             class="il.co.lerner.SimpleBean"/>
```

Amint látható, a tag perjellel (/) végződik – követve az XML írásmódját. Előfordulhat olyan eset is, amikor jobb szétválasztani a `<jsp:useBean/>` tagot nyitó `<jsp:useBean>` és záró `</jsp:useBean>` részre; ugyanis ami ilyenkor található a két tag között, csakis akkor hajtódik végre, amikor a bab először töltődik a memóriába. Természetesen az egyszerűsített forma is igen gyakori.
A `<jsp:useBean/>` tag két kötelező értékkel bír. A `class` érték azt a csomagot és osztályt nevezi meg, ahol a babunk található. Az `id` érték a JSP-n belül egyedi nevet rendel a babhoz. Akárcsak a változónevek esetében, nem árt, ha a babokkal való munkához egyértelmű azonosítókat választunk. Minél inkább magától értetődő a név, annál könnyebb lesz később nyomon követni a JSP-t.
A babpéldányunkból adatokat elővarázsolni és beállítani a `<jsp:setProperty/>` és `<jsp:getProperty/>` tagokkal tudunk. Mindkét tag egy `name` értéket vár, amelynek azonosnak kell lennie a korábban beállított `id`-vel (biztos vagyok benne, hogy megvan az oka, amiért a `<jsp:useBean/>`-ben `id` tulajdonságot használunk, míg a `<jsp:setProperty/>` tagban `name` tulajdonságot, de azért mindenképpen zavarónak tartom). A következő taggal tehát visszakaphatjuk az `userID` tulajdonság értékét:

```
<jsp:getProperty name="simple"
                 property="userID"/>
```

Az előbbi sor egyszerű babunkból letölti a `userID` tulajdonságot, és a JSP-be helyezi. Érdemes megjegyezni, hogy nemcsak egyszerűen letölti az értéket, hanem egyszersmind láthatóvá is teszi a felhasználó számára. Azt is érdemes megfigyelni, hogy tulajdonságnevünk kis- és nagybetűi kicsit megváltoztak. A babunkban található `getUserID` tagfüggvény eléréséhez, azaz a `userID` tulajdonsághoz a `<jsp:getProperty/>` tagot kell használnunk. Ez senkit ne tévesszen meg, a tulajdonságnevek nem kis- és nagybetűfüggetlenek; az átalakítás egyszerűen az olvashatóság kedvéért történik.
A tulajdonság értékének megváltoztatásához a `<jsp:setProperty/>` tagot használjuk. Ez a tag semmit sem ad vissza, elfogad viszont egy `value` tulajdonságot, amelynek értéke aztán a babosztály megfelelő tagfüggvényéhez kerül:

4. lista calculator.jsp

```
<HTML>
<Head><Title>Calculator</Title></Head>

<jsp:useBean id="calculator"
 class="il.co.lerner.Calculate"/>

<Body>
<H1>Egyszerű számológép</H1>

<jsp:setProperty name="calculator"
property="*"/>

<P>arg1 legyen <jsp:getProperty
   name="calculator" property="arg1"/>.</P>
<P>arg2 legyen <jsp:getProperty
   name="calculator" property="arg2"/>.</P>

<P>Összeg: <jsp:getProperty name="calculator"
            property="sum"/></P>

<P>Hányados:
     <% try { %>
          <jsp:getProperty name="calculator"
           property="quotient"/>
     <% } catch (Exception e) { %>
          <B>Hiba! Osztás nullával </B>
     <% } %>
</P>

<P>Különbség: <jsp:getProperty
              name="calculator"
              property="difference"/></P>

</Body>
</HTML>
```

```
<jsp:setProperty name="simple" property="userID"
                 value="300"/>
```

A *2. lista* egy teljes a JSP-t tartalmaz, amely bemutatja, miként használhatjuk JSP-ből a SimpleBean-osztályt. Először megjeleníti a `userID` tulajdonság alapértelmezett értékét, majd új értéket rendel a tulajdonsághoz, végül ezt az újat szintén megjeleníti.

Értékek és tulajdonságok

Nyilvánvaló, hogy általános gyakorlat a tulajdonságokat a bab példányváltozóiban tárolni, ahogyan azt mi is tettük a SimpleBean-osztály esetében. Ilyenkor a `<jsp:setProperty/>` meghívása tulajdonképpen a mező értékét állítja be, míg a `<jsp:getProperty/>` a pillanatnyi értékét adja vissza. Természetesen a tulajdonságoknak nem kötelező mezőkhöz kötődniük, ugyanígy tárolhatók és lekérdezhetők egy relációs adatbázis-kezelőből. Amikor egy tulajdonságot lekérdezünk, az is megoldható, hogy a visszatérési értéket valós időben számítsuk ki, ahelyett hogy egyszerűen csak kiolvasnánk egy példányváltozóból. Képzeljük el, például milyen könnyedén készíthetnénk egyszerű matematikai műveleteket végző babot! Beállíthatnánk két írható, illetve olvasható tulajdonságot (nevezzük őket arg1-nek és arg2-nek), majd létrehozhatnánk számos csak olvasható tulajdonságot, amelyek a korábban megadott paraméterekből számított értékeket adnák vissza. A 3. lista (mely az ftp://ftp.ssc.com/pub/lj/listings/issue86 címen is elérhető) egy ilyen, a fentieket bemutató egyszerű babot tartalmaz Calculate.java néven.

Mivel `setSum` tulajdonság nem létezik, a JSP-motor a `sum` tulajdonságra nem fogja engedélyezni a `<jsp:setProperty/>` meghívását. Ugyanakkor engedélyezi az arg1 és arg2 beállítását, illetve azt is, hogy bármelyik tulajdonságot lekérdezzük.

Most hogy van egy működő babunk, akár fel is használhatjuk a JSP alól. A calculator.jsp forrását a *4. lista* tartalmazza, ahol néhány alapműveletet végzünk az imént készített JavaBeannel.

A calculator.jsp első és legérdekesebb része az a mód, ahogy a tulajdonságokat beállítja:

```
<jsp:setProperty name="calculator" property="*"/>
```

Megszokott helyzetben fognánk az egyik GET vagy POST tagfüggvénnyel kapott értéket, és a következő jelölésmóddal valamelyik tulajdonsághoz rendelnénk:

```
<jsp:setProperty name="calculator"
                 parameter="foo" property="arg1"/>
```

Más szóval: a fenti tag fogadja a `foo` értéket és ezzel meghívja a calculator-bab `setArg1` eljárását. Amikor viszont csillagot használunk, ahogyan a 4. listában is, akkor azt jelezzük a JSP-motornak, hogy az összes megkapott értékre szükségünk van, és mindegyiket ugyanolyan nevű tulajdonsághoz szeretnénk rendelni. Így az `arg1` érték az arg1 tulajdonságba kerül és így tovább.

Az én rendszeremen, ahol a calculator.jsp-t az examples/jsp címre helyeztem, a következő URL segítségével rendelhetem az `arg1`-hez az ötös értéket és az `arg2`-höz a húszas értéket: „http://localhost/examples/jsp/calculator.jsp?arg1=5&arg2=20”. Természetesen ez nem éppen „bolondbiztos” rendszer. Könnyen futásidejű hibát idézhetek elő, például a következő URL átadásával: „http://localhost/examples/jsp/calculator.jsp?arg1=5&arg2=20.0”. A JSP-motor megpróbálja hozzárendelni a 20,0 (ez egy lebegőpontos szám) értéket az `arg2` paraméterhez (amely egész szám), és kudarcot vall.

Ha el szeretnénk kerülni a futásidejű hibákat, például közrezárhatjuk a `<jsp:setProperty/>` és `<jsp:getProperty/>` tagjainkat scriptlet-tagokkal, kihasználva a hagyományos Java try-and-catch eljárást. A következő kód például szavatolja, hogy soha nem kell foglalkoznunk a nullával osztás kivétellel, amikor a `getQuotient`-et használjuk:

```
<P>Négyzetgyök:
     <% try { %>
          <jsp:getProperty name="calculator"
          property="quotient"/>
     <% } catch (Exception e) { %>
          <B>Hiba! osztás nullával</B>
     <% } %>
</P>
```

Ugyanakkor ez a kódolási mód Javát visz a JSP-be, és pontosan ezt szerettük volna elkerülni, ezért is kezdtünk a babokkal foglalkozni. Az, hogy nem programozó társaink képesek-e kezelni az effajta kódot, nagymértékben függ a környezetünktől, illetve attól hogy milyen típusú babokat készítettünk.

A babok hatóköre (scope)

Amint az a múlt hónapban kiderült, a servlettároló egy időben minden servletről csak egyetlen másolatot tölt a memóriába. Ez azonban

6. lista viewblog.jsp

```
<HTML>
<Head><Title>Web-napló</Title></Head>

<jsp:useBean id="blog"
class="il.co.lerner.Weblog"/>

<Body>
<H1>Aktuális Web-napló tartalom</H1>

<table>
<jsp:getProperty name="blog" property="blog"/>
</table>

</Body>
</HTML>
```

nem okoz gondot, mivel a Java többszálú, és egy adott servletnek lehetősége van egyszerre több HTTP-kérelemmel is foglalkozni. Mivel a JSP-k tulajdonképpen álruhás servletek, ugyanígy rájuk is vonatkozik a többszálas futtathatóság.
Felmerül a kérdés, mi történik JavaBeanjeinkkel: hányszor töltődnek be, mekkora a hatókörük (scope). Ha a JSP egy olyan tulajdonságot állít be, amely az osztály mezőit módosítaná, hatással van-e ez ugyanennek a babnak a többi példányára is?
A válasz: attól függ. A babok elérési területeinek négy fajtáját különböztethetjük meg, és a választott elérés típusa alapjaiban meghatározza a felhasznált bab viselkedését. Az alkalmazásszintű hatókör (Application scope) egyetlen másolatot jelent a babból minden, a servlettárolóban futó JSP-hez. A folyamatszintű (Session scope) a felhasználóhoz rendelt – attól a pillanattól kezdve, hogy belépett a honlapra, addig a percig, amíg ki nem lép onnan. Ha a felhasználó két böngészőablakot nyit a rendszerre, azok azonos folyamatnak minősülnek. A kérelemelérési szint (Request scope) a HTTP-kérelem végéig tart. Ez olyankor hasznos, ha az egyik JSP-ben szeretnénk beállítani a bab tulajdonságát, majd a `<jsp:forward/>` taggal belső átirányítást hajtunk végre egy másik JSP-re, s ebben a második JSP-ben szeretnénk a babot felhasználni. Végül a lapelérési szint (Page scope) egyetlen JSP-lapra vonatkozik. Amikor a lap feldolgozásra kerül, az elérhetőség is megszűnik.
Alapértelmezés szerint a babok a folyamatszintű hatókörrel futnak. Ennek megváltoztatásához a scope-értéket kell a `<jsp:useBean/>`-be helyezni:

```
<jsp:useBean id="simple" scope="application"
             class="il.co.lerner.SimpleBean"/>
```

Ha egyszer végrehajtottuk a fentieket, az egyik JSP beállította értékek az összes többiben is elérhetők lesznek. Természetesen – mivel az alkalmazás- és kiszolgálószintet egy időben több JSP is elérheti – szálbiztosnak kell lennie.
Gondoljunk rá, milyen elérhetőségű lesz a babunk és ha szükséges, tegyük szálbiztossá. Ha a bab nem szálbiztos, mindenképpen jelezzük ezt a leírásban, nehogy mások véletlenül hibásan használják fel.

Webnapló

Az elmúlt hónapban folytattuk egyszerű vizsgálódásunkat a JSP-vezérelt webnaplók terén, melyek közvetlenül értek el relációs adatbázisokat a legutóbbi webnaplóbejegyzés lekéréséhez. Bár az ilyen JSP természetesen jó, de meglehetősen nehézkes, bonyolult a nyomon követése, és a kódtól a logikát nem választja szét olyan választékosan, mint ahogyan azt elvárhatnánk.
Ha az adatbázis-logikát JavaBeanbe helyezzük, célkitűzéseink közül elérhetünk néhányat: a kód újrafelhasználható, akár a nem programozók számára is elérhető lesz, és lehetővé teszi, hogy anélkül változtassuk meg az adatforrást vagy a belső logikát, hogy ehhez újra kelljen írnunk a JSP-t. Az 5. lista (elérhető a ➲ ftp://ftp.ssc.com/pub/lj/listings/issue86 címen) tartalmazza babunk forráskódját, amelyet szálbiztossá is tettem (a `synchronized` kulcsszó felhasználásával a `getBlog` tagfüggvényben), így alkalmazásszinten elegendő egyetlen példányt készítenünk. Ez a JavaBean hozzákapcsolódik a webnapló-adatbázishoz és lekérdezi a legfrissebb adatokat. A *6. lista* egy kis JSP-t tartalmaz, amely JavaBean segítségével jeleníti meg a webnaplót. Tulajdonképpen semmi különleges nincs az 5. listában, viszont számos olyan dolgot egyesít magában, amelyeket az elmúlt pár hónapban megvitattunk. Immár rendelkezünk egy olyan módszerrel, amellyel mindenki képes webnaplónk adataihoz hozzáférni anélkül, hogy egyetlen sor kódot is kellene írnia! Néhány egyszerű JSP-taggal a jelenlegi blog-tartalmat könnyen és egyszerűen behelyezhetjük a HTML-lapba.

Összegzés

A JavaBean-babok nagyszerűek: a JSP-be kerülő kódmennyiség csökkentése végett helyezzük a lapba, mellyel egyúttal a használatát is egyszerűsítjük. Ugyanakkor egy összetett JSP még mindig tartalmaz valamennyi kódot: ha például a ciklusokban kell ismételni valamit, vagy összetett adatokkal kell dolgozni.
A következő hónapban megtudjuk, miként írhatunk saját – az e havihoz hasonló – jsp: tagokat a JSP testreszabott eljáráskezelő képességének kihasználásával. Így elkészíthetjük a saját tagjainkat, melyekhez olyan kódot rendelhetünk, amilyet csak akarunk. A JSP lehetővé teszi, hogy felépítsük saját formátumnyelvünket és emellett az eddig elérhető tagokat is felhasználhassuk.

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/).

Kapcsolódó címek

Az Apache Software Foundation Jakarta-Tomcat Java servletekkel és JSP-kel foglalkozó projektje a ➲ http://jakarta.apache.org/ címen érhető el. Erről a címről letölthetjük a forráskódot, a binárisokat, a leírást, valamint a telepítéshez szükséges utasításokat, illetve a teljesítmény növelését segítő ötleteket.
➲ http://jakarta.apache.org/tomcat/jakarta-tomcat/src/doc/mod_jk-howto.html

Hans Bergsten, az O'Reilly and Associates által közelmúltban kiadott JavaServer Pages igen kitűnő forrásmunka azok számára, akik JSP-oldalakat szeretnének készíteni.
➲ http://www.oreilly.com

Heroes of Might and Magic III Linuxra

Régebben mindenki a körökre osztott stratégiai játékokért bolondult, de manapság úgy tűnik, hogy ebben a lövöldözős és valós idejű stratégiai játékokkal zsúfolt világban vesztettek a népszerűségükből. A 3DO cég Heroes of Might and Magic III: Restoration of Erathia című játékának Loki által Linuxon megjelentetett változata nagy hévvel fogott hozzá e hiba kiküszöböléséhez egy olyan játékban, amelynek függőséget okozó jellemzői – a szó szoros értelmében – annyira rabul ejthetnek, míg a csuklód görcsös remegésben nem tör ki, vagy míg a szemed üvegessé nem válik. Azt gondolom, ha káros szenvedélyekre hajló személyiséged van, ha egy háztartásban élsz valakivel, aki szeretne szót váltani veled a következő pár nap folyamán, ha nem hiszed, hogy a főnököd elfogadná a „most nem tudok beszélni, játszom…”-ot érvényes igazolásként a kivett betegszabadságra – kétszer is gondold meg, mielőtt leülsz játszani a Heroes III-mal. A helyzet az, hogy a játék körökre osztottsága – ami a felszínen olyan jóindulatúnak látszik gyorsléptű rokonaival összehasonlítva – kimossa az ép elmédet, és langymeleg mocsárba húz. Azon kezdesz gondolkodni, hogy már csak pár körre van szükséged ahhoz, hogy felkészítsd magad egy másik kisebbfajta győzelemre a harcmezőn, vagy fontos előnyhöz juss, ami új távlatokat nyit előtted. Új stratégiát, és ez természetesen megint csak arra serkent, hogy elhidd, már csupán néhány kör szükségeltetik ahhoz, hogy felkészülj valami másra; beszippant ez az ördögi kör, ahogy erőnek erejével hajszolod magad a győzelem felé (vagy amíg teljes végkimerülésedben elalszol). A Heroes III a stratégia, a kockázat, a taktika és a szerencse játéka; a Dungeons and Dragons, a Chess, a SimCity és a Warcraft legjobb elemeinek elegye, mesteri módon vegyítve egy olyan RPG-be, amelyben rettenetes hadseregek vezetője lehetsz, és amelyben természetfölötti lények küzdenek varázslók, harcosok és boszorkányok (vagy még rosszabbak) ellen.

Adatok

Gyártó: Loki Entertainment Software
E-mail: sales@lokigames.com
Cím: ➲ http://www.lokigames.com/

A történet

A Heroes of Might and Magic III a legutóbbi folytatás a Might and Magic játékok világának hőskölteményében, és ott veszi fel a fonalat, ahol a Heroes II – a kilencedik cím a mondában – befejeződött. Most Erathia világrészén (ahol a mészárlás és hódítás bonyolult története tárul fel előtted) szerepelsz Catherine Ironfist szerepében, aki arra törekszik, hogy kinyomozza apja halálát, és visszaállítsa Erathia hajdani dicsőségét. A történet természetesen elbeszélő filmjeleneteken és képbevágásokon keresztül tárul fel, hogy könnyen követhető formában elmagyarázza az új helyzetet. Az alapvető elgondolás szerint te rendelkezel a *Hősök* csapatával, a játék fő karaktereivel, valamint a településekkel, melyeket meghódítanak; a nyersanyagforrásokkal, melyeket elfoglalnak; a földekkel, melyeket felderítenek; a hadseregekkel és a kincsekkel, melyeket találnak. A hőseid által teljesítendő küldetések általános D&D stílusú célok (például Öld meg az arany sárkánykirálynőt! Leld meg a Grált, mielőtt az ellenségeid találnák meg!), és ahogy szereplőid fejlődnek és egyre érettebbé válnak, úgy gyarapítják tapasztalataikat, új erőiket, ügyességüket, tudásukat és harcosaik számát, akiket csatába visznek. A játék mérete homéroszi, bár választhatsz rövidebb egyjátékos módot, ha azonban nem vigyázol, gyakran még a középméretű kalandok is eltartanak egy órácskáig. A karaktereid fejlődnek és érettebbé válnak a hosszú célközpontú játékok sorozatában, hogy elérjék a végső győzelmet, ami számos nappalt és éjszakát is igénybe vehet.

A játék menete

Ahogy már az előbb említettem, a Heroes III körökre osztott játék, minden forduló egy erathiai napig tart. Ez olyan nap, amelyben irányíthatod a hőseidet (küzdelem, mozgás, tanulás, tanítás, felfedezés, nyersanyagforrások megszerzése stb.) és az uralmad alatt álló városok tevékenységét (új épületek építése, új harcosok vagy hősök toborzása, áruk, varázstárgyak adásvétele stb.). Ha elvégezted napi teendőidet, befejezed a kört, és addig vársz, amíg az ellenfél is befejezi (a számítógép által irányított ellenfeleknek ehhez mindössze néhány másodpercre van szükségük), majd kezdődnek a következő napi események. Mindegyik városban a harcosok csak hetente egyszer toborozhatók, így ha hétfőn elmész kalandozni és elveszted katonáid nagy részét, csak remélheted, hogy a maradékkal fel tudod tartani a támadókat a hét végéig.
Ha azt mondod, hogy a játékmenet bonyolultnak látszik, igazad van, tényleg az. A tanulókör meglehetősen túlzó és mélyreható pillantást enged a játék minden részletébe, a mellékelt 133 oldalas felhasználói kézikönyv pedig komoly tanulmányozást igényel.
A CD-n található 12 oldalas PDF formátumú oktatókönyv és a játékba épített oktatóküldetés végigjátszása csupán 45 percet igényel. Miután elsajátítottuk a játék alapvető szolgáltatásainak módfelett tömör parancsait, valamint elég megbízhatóan tudjuk már kezelni a játék szerkezetét, belevethetjük magunkat az igazi játékba és megtanulhatjuk a sajátos jellemzőket. Tény, a játék összetettsége ellenére kezelőfelülete elég érthető ahhoz, hogy sok játékos barátságosnak találja és kipróbálja az oktatójátékot, majd pár perc alatt a válogatott vezérlőelemek között kotorászva eleget tanuljon ahhoz, hogy a jó úton járónak érezze magát és lejátsszon egy rövid, aránylag könnyű küldetést. Mi több, a fejlesztők,

tekintetbe véve a játék bonyolultságát, bőséges szöveges segítséget állítottak össze. Tulajdonképpen a játékban bármely helyen egy jobb egérgombkattintás beugró ablakot eredményez, ami szabatosan, bő részletességgel írja le az egérmutató alatt lévő területet, ikont. A játékmenet az irányításod alatt álló hősöket és a városokat helyezi a középpontba. A hősök felelősek azért, hogy felderítsék a területeket a városok körül, és néhány varázstárgyat megtaláljanak a csaknem 130-féléből; olyan nyersanyaglelőhelyeket kutassanak fel, mint például a bányák és egyéb, a városok számára bevételforrást jelentő telepek. Ezenkívül meg kell védeniük a városokat a támadásoktól, továbbá hogy növelni tudják a birtokukat, új városokat kell meghódítaniuk és meg kell küzdeniük más hősökkel. Az általad felkutatott nyersanyagforrások kiaknázásából aranypénzhez juthatsz, segítségével épületeket építhetsz, hősöket és harco-

1. kép A Birodalom felderítése

sokat toborozhatsz. A harcosok értékes szolgálatot nyújtanak hőseid számára: azon serénykednek, hogy a katonák és a lények összes módszerével megtisztított terepen hőseid minél jobban fel tudják javítani a seregüket.

Nyolc különféle típusú, több mint egy tucat különböző épületből álló város van. Az épületek tulajdonságai annak a városnak a típusára jellemzők, amelyben megtalálhatók: például a *Kastély* típusú városban olyan épületek találhatók, mint az őrház (ahol dárdás katonákat lehet toborozni), a kolostor, a kiképzőtér (a lovasok kedvenc búvóhelye), és a természetfeletti építmény, mint a szent *Dicsőség Bejárata*, ahol a roppant erős Arkangyalok toborozhatók – meglehetősen borsos áron. Az ellentétes oldalt a *Pokol* típusú város képviseli, ahol olyan baljóslatú építmények építhetők, mint a tűztó, a természetfeletti Efreetek lakhelye, illetőleg kennel a Pokol Kutyáinak. Itt nyílik lehetőség a halálos Cerberus toborzására, sőt itt található az *Elhagyatott Kastély*, ahonnan az Ördögök hívhatók hadsereged szolgálatába. Más városok adnak otthont a varázslóknak és dzsinneknek, a Pegazusnak és a törpéknek, sőt a sárkányoknak és az emberevő óriásoknak. Összesen 118 különböző fajtájú lény áll rendelkezésedre, mindegyikük saját különleges támadási jellemzőkkel és vonásokkal bír. E jellegzetességek segítségével játszhatnak különböző taktikai szerepeket a csatatéren, a távolról végzett támadásoktól a közelharcig. Minden épületnek; szerkezetétől függően, meghatározott költsége van, és néhány szereplő, illetve épület nyersanyagokkal és különböző eszközökkel szolgálhat. Például a kovács fegyvereket állít elő; a könyvtárak a belátogató hősök tudását gyarapíthatják, a mágustornyokban a betérő hősök varázslatokat és varázsigéket sajátíthatnak el; az új hősök mindig a helyi kocsmákban lebzselnek, arra várva, hogy felbéreld őket a királyság seregeinek vezetésére.

128 különböző hőssel harcolhatsz, ezek mindegyike egyedi jellemvonásokkal bír, erősséggekkel és gyengékkel egyaránt. Ezek a hősök vezetik a haderődet, seregeikkel felkutatják a területeket, megszerzik a birodalom számára szükséges nyersanyagforrásokat, és megvédik a független területeket a betolakodóktól. A hősök közvetlenül nem vesznek részt a csatákban, de különböző varázslataikkal, tudásukkal és tapasztalataikkal hathatós segítséget nyújtanak az irányításuk alatt álló katonáknak. A csata körökre osztott küzdelem során zajlik,

2. kép Egy Erőd kezdeti állapotban

3. kép Csata

hatszögletű rácsokra tagolt harcmezőn, ahogy a *3. képen* is látható. A harcok során mindkét fél felváltva léphet az egyik fél győzelméig. Háromféle diadalt arathatunk: az egyik fél teljes győzelme során megsemmisíti az ellen egész seregét, vagy az egyik fél megadja magát, vagy pedig visszavonul. Visszavonulás esetén a hős ugyan megszégyenülten elfut, de később újra toborozható lesz, és tovább szolgálhatja a seregedet. Megadás esetén azonban túl nagy a szégyen, és a hős soha többé nem tér vissza. A csata megnyerésekor a hős úgynevezett tapasztalatpontokat kap, melyek segítségével az érettség új szintjére léphet – több tudást és tapasztalatot szerezhet. Ezek segítségére lehetnek a csatában vagy a felfedezésben: taktikai szakértelemmel a csata megkezdése előtt győzelemesélyesen helyezheti el a katonákat a harcmezőn; navigációs tudás birtokában egy nap alatt messzebbre tud hajózni; diplomáciai készséggel elcsábíthat másokat, hogy csatlakozzanak a seregéhez és így tovább. Erőddel körülvett város ostromlásakor különleges ostromképernyő tárul elénk, ahol

a támadó hadsereg katapultokkal is bombázhatja a város falait. A Heroes III-ban a grafika csodálatosan részletes, színes és összetett. A különböző kezelőfelületek érthetők és kézre állók, nagy részüket szerencsére az elbűvölten ülő játékos ki is tudja használni. A hanghatások és a háttérzene terén a Heroes III valóban kiváló; a zene úgy változik, hogy mindig illeszkedjék a játék hangulatához. Fedezd fel a királyságot – klasszikus szimfóniát és erőteljes, szárnyaló zenét fogsz hallani. Menj csatába – és a muzsika sejtelmessé, várakozással telivé válik. Vágtass be egy kastélyba, s Arthur korabeli csemballó hangja csendül fel. Menj le a földalatti várbörtönbe és hátborzongató, mély oboával és templomi haranggal fűszerezett dallam hallatszik. A zene sokat emel a játék hangulatán, soha nem tűnik unalmasnak vagy ismétlődőnek.

4. kép A fantasyjátékok őse ismét a kezünkben Linux alatt

5. kép A játék kiválasztása

A többjátékos mód a körökre osztott stratégiai játékok egyik legjobb része, de a Heroes III-nak ez az üzemmódja sajnos a múltat idézi. Mialatt TCP/IP-alapú játékra lehetőség van a játékban, nincs megfelelő módszer arra, hogy csatlakozz valamely típusú központosított, állandóan elérhető közösséghez. Meg kell keresned valamilyen módszerrel a Heores III játékkiszolgáló IP-címét és közvetlenül kell kapcsolódni hozzá. Választásként játszhattok „forró szék” játékot: aki a székben ül, lép, majd átadja a helyét és a következő játékost engedi játszani. Nem nyílik lehetőség levelezőjátékra, mint oly sok más körökre osztott játékban. E szokás szerint az elmentett „forró ülés” fájlokat kicserélik egymás között, és így játszanak nagyon hosszú partikat. Ez a bíráló vélemény az egyetlen rossz pont a Heroes III-ban, az a terület, amit a 3DO komolyan elhanyagolt.

Linuxos megjegyzések

Egy Matrox G400-as alapú, 500 MHz-es PIII-as processzorral szerelt rendszeren kipróbálva hibátlanul teljesített. Sejtésem szerint jól működne egy sokkal gyengébb gépen is, mivel nem igényel 3D-támogatást, és a csaták, illetve a mozgások nem számításigényesek. A Loki legkisebb gépkövetelményként 32 MB-os, 133 MHz-es vasat jelölt meg. Szükséged lesz még egy OSS-megfelelő hangkártyára, valamint egy X-kiszolgálóra, ami meg tud jeleníteni 800x600 felbontást 16 bit színmélységben. A játék telepítése nagymértékben testre szabható, mivel a telepítőprogram megengedi, hogy elkülönítsd a fő összetevőket, és a forgatókönyvek, hangok, zenék, grafikák és videók egy része a CD-n maradjon, ha akarod. Ezáltal 5 MB-tól 350 MB-ig terjedhet az igényelt merevlemez-terület. Attól függően, mennyi adatot hagytál a CD-n, szükséged lesz egy legalább 4x-es sebességű CD-ROM-meghajtóra is; a videókat lassú meghajtóról futtatva csúnya szaggatottság jelentkezhet. Természetesen a Loki legalább 2.2-es vagy újabb rendszert javasol.

Előnyök

- A legjobb körökre osztott stratégiai játék közel s távol
- Változatos hadjáratok és játéktípusok
- Kiváló grafika és zene

Hátrányok

- Többjátékos üzemmód korlátozott
- Bonyolult játékmenet
- A hosszú játékidő egyeseknek nem tetszik

Kisebb gondok vannak az alapértelmezett telepítéssel. Például a Loki által kiadott változat nem tudja alapértelmezetten a teljes képernyős módot. Bár a Loki frissítésfoltja körülbelül csak egy megabájt, számos apró hibát kijavít, beleértve a teljes képernyős hibát is; ajánlatos az alapértelmezett telepítés után lefuttatni, még a játék megkezdése előtt. Érdekességképpen: ez a folt tartalmazza az Aalibet (ASCII Art Library) – amely feltételezhetően a hűséges, de kétségbeesett Heroes III játékosoknak nyújt segítséget abban, hogy X nélkül futtassák a játékot konzolról, nagyszerű ASCII megjelenítéssel. Ráadásul a Loki a Heroes III térképszerkesztőjének béta-változatát is kínálja, mellyel saját Heroes III kalandokat szerkeszthetsz. A 17 MB-os ingyenes demováltozat letölthető a Loki weblapjáról ➲ http://www.lokigames.com/.

Összegzés

Szerintem ez a játék tökéletes: körökre osztott stratégia annak minden finomságával. A kezelőfelület letisztult, sőt, ha ezelőtt még soha nem játszottál RPG-típusú játékkal, a kiváló szakmai munka, a figyelem és a részletesség magas színvonala miatt – ami az egész játékon érezhető – első látásra minden magától értetődőnek tűnik. Bár a játék meglehetősen összetett, rendkívül gyorsan elsajátítható. A Heroes III olyan darab, amivel leülsz játszogatni és csak nyolc órával később fogod abbahagyni, mert aludnod is muszáj valamennyit. Az összes korosztályú pingvinnek ajánlom.

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).

Kylix

Hatékony alkalmazásfejlesztés Linuxon

A Borland cég Kylix rendszere-a nagy sikerű Delphi Linuxra átírt változata. Anyanyelve az *object pascal*, amely a Pascal objektumközpontú továbbfejlesztett változata. A nyelv formai követelményeinek köszönhetően nagyon jól használható összetett objektumszerkezetek kezelésére (például objektumok többszintű egymásba ágyazására), és ugyanezen okból könnyű ezen jól áttekinthető programkódot írni. A rendszer a QT grafikus könyvtár alapjaira épült. Mind a fejlesztőeszköz, mind az általa fejlesztett programok szervesen illeszkednek a KDE ablakkezelő-rendszer felületébe, de természetesen nem csak KDE alatt futtathatók. A rendszernek három változata kapható. Az ügyféloldali alkalmazásfejlesztőknek szánt Desktop Developer, a kiszolgálófejlesztők számára kibocsátott Server Developer, és a nyílt forráskódra fejlesztők számára kiadott ingyenesen elérhető Open Edition. Ez utóbbi megjelenése e cikk írásának pillanatától számítva heteken belül várható.

Telepítés

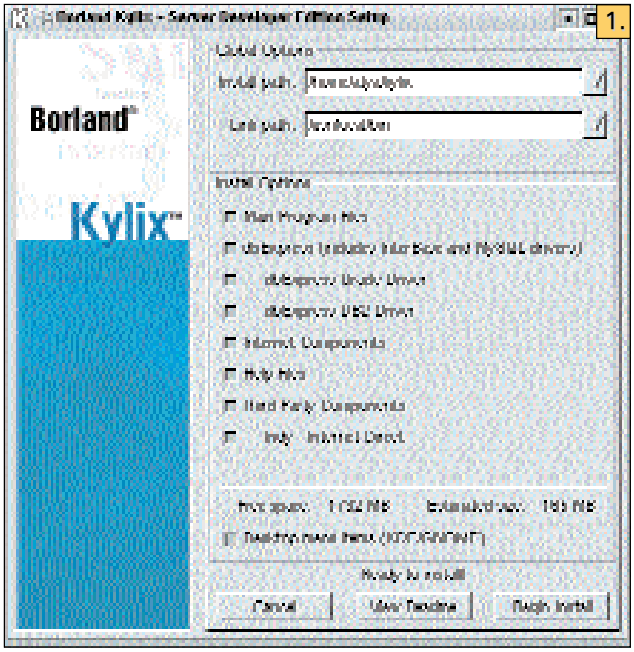
A Kylix viszonylag szerény erőforrásigényű. Futtatásához elegendő egy Pentium II 400 MHz-es processzorral rendelkező számítógép, 128 MB memóriával. Elvileg ennél gyengébb rendszeren is fut, de tapasztalatom szerint a hatékony fejlesztéshez a megadott kiépítés szükséges. A fejlesztők által ajánlott operációs rendszerek:

- SuSE 7.0 vagy későbbi kiadás,
- RedHat 6.2 vagy frissebb változat,
- Mandrake 7.2 vagy nagyobb változatszámú terjesztés.

Ez a gyakorlatban azt jelenti, hogy minden terjesztés használható, amely 2.2-es vagy újabb libc-t tartalmaz és rendelkezik grafikus felülettel. Én Debian Woodyra telepítettem, de kisebb hibákat tapasztaltam. Gyakorlatlan Linux-felhasználóknak az ajánlott terjesztések egyikét javaslom. A Kylix Server Developer dobozos termék tartalmazza a SuSE Linux 7.0-s változatát, így szükség esetén az operációs rendszer is telepíthető. A telepítőrendszer nagyon kényelmes grafikus felületet ad (*1. kép*), ahol ki lehet választani, hogy a rendszer mely részeit szeretnénk telepíteni. Egyetlen dolgot nem értek. Miért nem használja a Linux által adott csomagkezelőt, holott annak éppen ez lenne a feladata? A telepítésnél meg kell adnunk egy könyvtárat, amelybe a Kylix kerül (teljes telepítés esetén nagyjából kétszáz megabájtnyi hely szükséges). Ebben a könyvtárban a Kylix futási környezetet alakít ki magának saját függvénykönyvtárakkal.
Ez a módszer linuxos szemmel furcsának tűnhet, de megvan a maga haszna. A Kylix túléli a nagyobb programkönyvtárcseréket is, mivel a saját könyvtárait használja, viszont a vele fordított programoknak a valós rendszeren kisebb-nagyobb gondjai lesznek. Kellemetlen, hogy az üzembiztos Debian-rendszerben jelenleg nincs olyan csomag, amely tartalmazza a libqtinf.so-t. Mivel a fejlesztett program futtathatóságához erre a függvénykönyvtárra szükség van, kénytelen voltam a Kylix könyvtárából kimásolni a /usr/local/lib könyvtárba.

A fejlesztőrendszer

A Borland szokásához híven most is elegendő leírással látta el fejlesztőit. A doboz az alábbi három angol nyelvű könyvet tartalmazza: Developers Guide, Quick Start és Language Guide, amelyek a CD-n PDF formátumban is megtalálhatók. Véleményem szerint ezeknek, valamint a Súgórendszernek (*2. kép)* és az elemek forráskódjának birtokában bármely fejlesztés közben felmerülő gond orvosolható. Nálam a rendszer első indításakor egy *Generating font matrix. Please wait…* feliratú párbeszédablak jelent meg és valami megette a teljes processzoridőt. Egy darabig bírtam cérnával, aztán mivel úgyis fontos söröznivalóm volt, otthagytam. Mivel visszatértemkor még mindig dolgozott, kezdtem gyanút fogni… Az strace segítségével hamar kiderült, hogy végtelen ciklusban kering, így lezártam az ablakot.
A hiba okát feltehetőleg a saját beszerzésű betűkészletek jelentették. Az ablak lezárása után a Kylix hibátlanul elindult. Azóta telepítettem

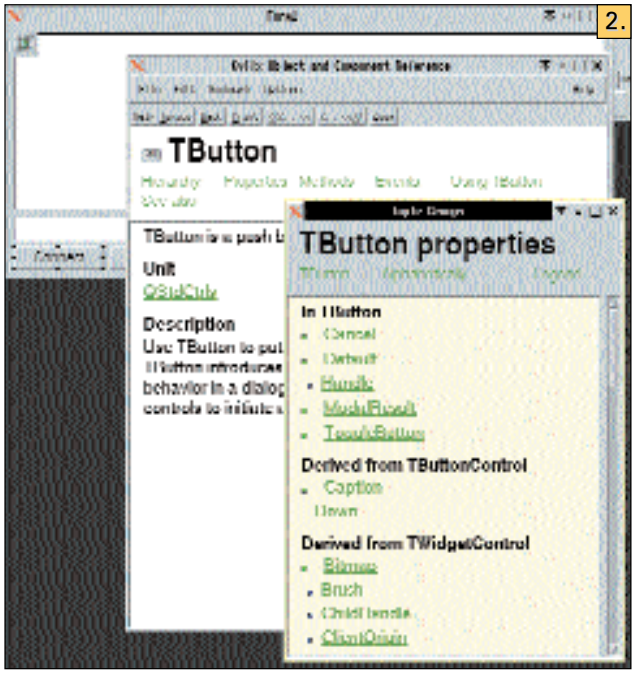


már SuSE 7.1-re és Debian Potato alaprendszerre is. Egyik alatt sem jelentkezett az előbbi hiba, futásuk között pedig látszólag semmi különbség sincs. Egy dologra viszont a fenti gond felhívta a figyelmet. A Kylix ugyan valódi Linux alatt futó program, de bizonyos Win32-höz kapcsolódó szolgáltatások Windowsról való átemelésére a Borland a Wine-t használta. A Kylix általános sebessége megfelelő (egyedül a Súgórendszer lassú kissé), az általa fordított programoknak pedig egyáltalán nincs szükségük a Wine-ra.

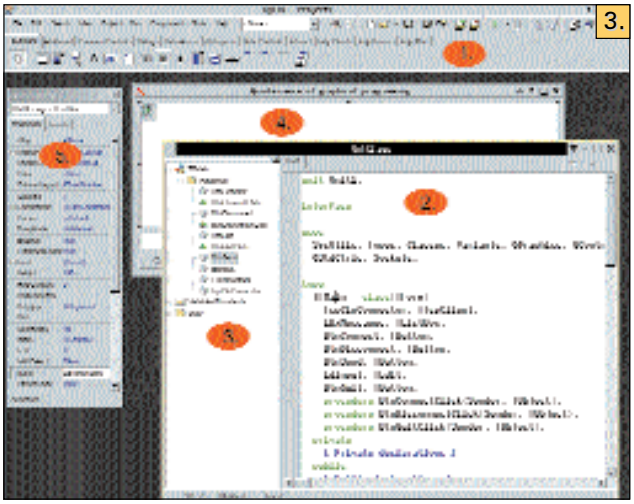
A Kylix felülete

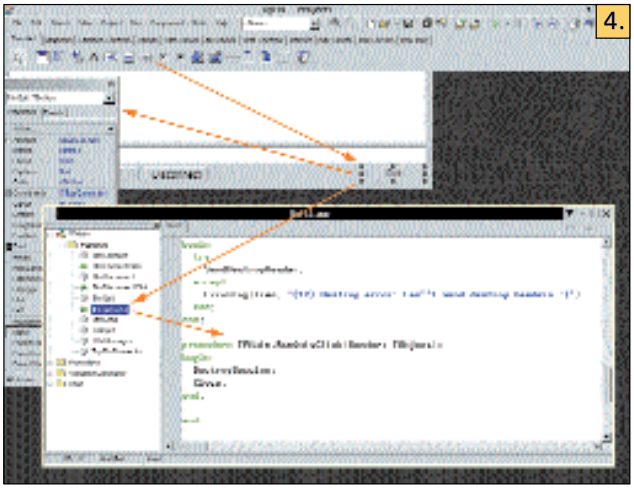
A rendszert futtatva a *3. képen* látható kép tárul a szemünk elé. Öt alapvető rész különíthető el, melyeket az ábrán számokkal jelöltem. Az öt rész a következő:

1. a menüt, az eszköztárat és az elempalettát tartalmazó ablak;
2. a forráskód módosítására használható szerkesztőablak;
3. a Code Explorer, amely gyors tájékozódást tesz lehetővé a szerkesztett alkalmazás forráskódjában;



2.


3.


4.

4. az alkalmazás felületének megtervezésére használható Form Designer;
5. az alkalmazás elemeinek finomhangolását lehetővé tevő Object Inspector.

A Kylix a Linuxon jelenleg rendelkezésre álló eszközökkel ellentétben nemcsak a rendszer felületének megtervezését teszi lehetővé, hanem az eseményekhez tartozó forráskód is azonnal szerkeszthető. A fejlesztés folyamatát mutatja be a *4. kép*, mely a különböző részek összefonódását szemlélteti. A fejlesztő az elempalettáról összeválogatja a szükséges elemeket, és a Form Designer segítségével kialakítja a leendő ablakok tervét. Az ablakra kerülő felületelemek jellemzőit az Object Inspector segítségével lehet beállítani. A szükséges eljárások és függvények megírására a szerkesztőablakban van lehetőség, majd a felület objektumaihoz köthető eseménykezelőket kell megírni. Az eseménykezelők szerkesztését szintén az Object Inspectorból a legegyszerűbb kezdeményezni, mert itt az adott felületelemhez tartozó összes fontosabb eseménykezelő megadható. Az általánosan használt ablaktervek és egyéb, többször használt elemek elmenthetők egy mintakatalógusba (Object repository), ahonnan újból felhasználhatók. A Form Designer szokatlan jellemzője, hogy az X Window System alatt eddig megszokott felülettervezőkkel ellentétben kizárólag állandó méretű objektumokkal dolgozik (mint tudjuk, X alatt a grafikus objektumok általában az ablakkal együtt önműködően méreteződnek), így az ablak átméretezése esetén a felület részeinek átméretezéséről szükség esetén magunknak kell gondoskodnunk.

Fő cél: a hatékonyság

A Borland sokéves tapasztalata jól tükröződik a Kylix felépítésében és működésében. Számos olyan, Linuxon eddig elérhetetlen eszközzel segíti a fejlesztőt, amelyek a hatékonyságot akár a többszörösére is növelhetik. Nézzünk meg – a teljesség igénye nélkül – néhány olyan eszközt, amelyek megkönnyítik a fejlesztést! A programozónak nagyobb projektek esetén nem szükséges fejben tartania akár több ezer objektum- és eljárásmeghatározást, az összevont fejlesztőrendszer ugyanis kérésre kiegészíti azokat. A nagy tudású beépített nyomkövető meggyorsítja és megkönnyíti a hibakeresést, illetve -elhárítást.

A helyzetérzékeny súgó segítségével a rendszerbe épített elemek használatát villámgyorsan el lehet sajátítani. Az alábbi forráskód szerkesztését segítő eszközök összefoglaló neve CodeInsight.

Code Completion

Ha beírunk egy osztálynevet ponttal kiegészítve, a rendszer felajánlja az adott osztály tulajdonságait (properties), tagfüggvényeit (methods) és eseményeit (events) (lásd *5. kép*). Egy eljárás, függvény vagy tagfüggvény nevét beírva a rendszer megadja annak értéklistáját.

Code Parameters

Egy tagfüggvény nevét és egy nyitó zárójelet beírva megkapjuk a tagfüggvény értékeinek formai követelményeit.

Code Templates

A CTRL+J lenyomására a rendszer felajánlja a mintatárában található gyakran használt programrészleteket. A mintatárat természetesen mi magunk is bővíthetjük.

Tooltip Expession Evaluation

A program nyomkövetése közben, amikor az éppen áll, bármely változóra mutatva megjelenik annak pillanatnyi értéke.

Tooltip Symbol Insight

A kód szerkesztése közben bármely azonosítójára mutatva megjelenik annak ismertetése.

A fejlesztőt segítik a fordítási hibáknál előbukkanó tippek, amelyek – láss csodát! – tapasztalataim szerint a leggyakrabban rá is mutatnak a valós hibára. A fordító nagyon gyors, hibaüzenetei jól érthetők. A forráskódban való gyors mozgás elősegítésére a rendszer kereszthivatkozásként tudja kezelni az azonosítókat, és böngészőhöz hasonló módon az adott hivatkozás meghatározásához ugrik. A visszajutás meggyorsítására mind a Forward, mind a Back gomb igénybe vehető.
Néhány további tulajdonság címszavakban:

- A fejlesztőket beépített teendőlista (To-Do list) segíti.
- A szerkesztőablakban a különböző nyelvi elemek könnyebb megkülönböztetésük végett eltérő színekkel jelöltek.
- Az eszközpaletták a kényelem érdekében teljesen átszerkeszthetők, illetve bármely ablakba beilleszthetők (docking system).
- A billentyűk kiosztása átszerkeszthető. A számos beépített kiosztás között az Emacsra jellemző is megtalálható.
- Függvénykönytárak és többszálú programok nyomkövetési lehetősége.
- Nyomkövetés közben képes naplót készíteni a fontosabb eseményekről, így lehetővé teszi a későbbi elemzést.

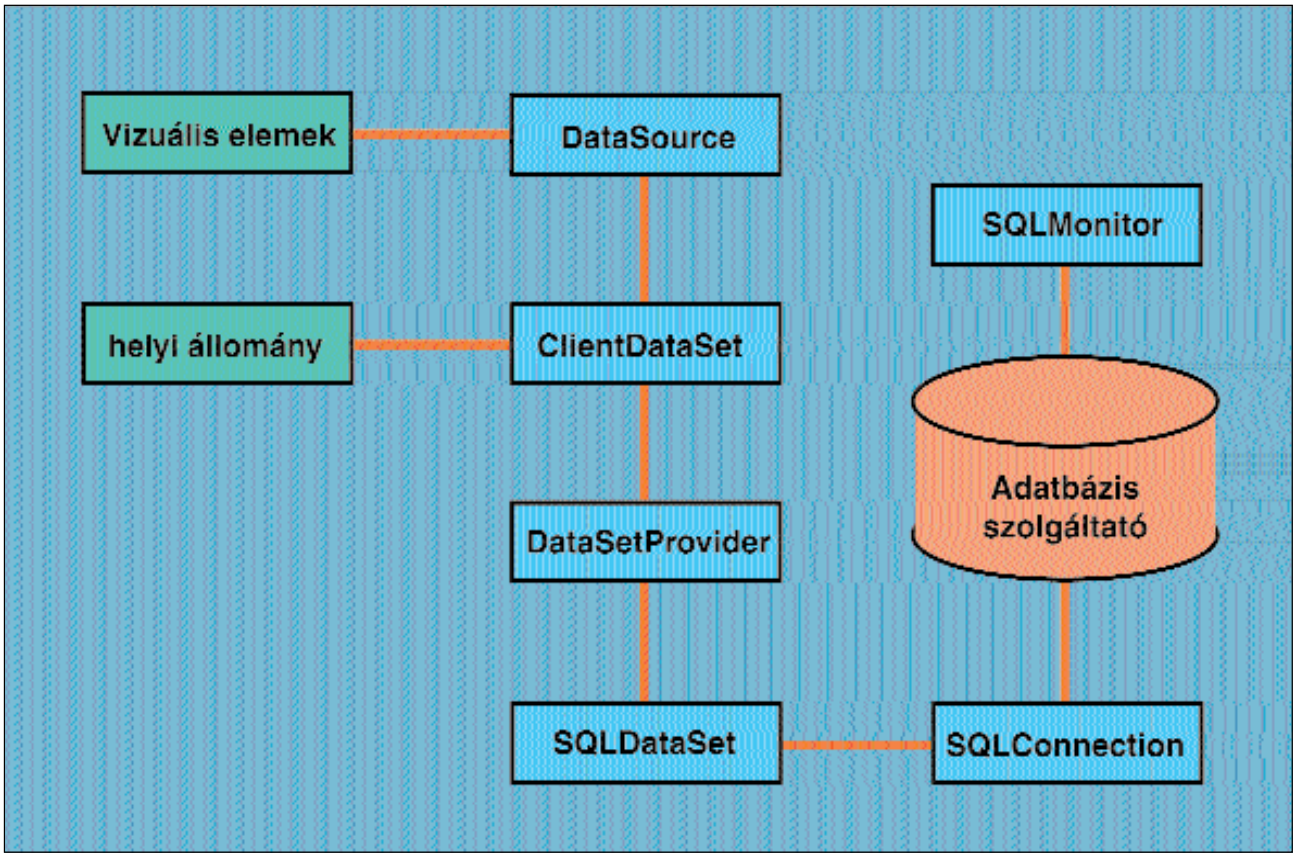

5.

- Nyomkövetés közben a töréspontok csoportokba foghatók össze, majd szükség esetén együtt élesíthetők vagy hatástalaníthatók.
- Mintaalkalmazásokat tartalmaz a programozási módszerek gyors elsajátításának elősegítésére.
- A Server Developer-változat olyan elemeket is magában foglal, amelyek segítségével az Apache webkiszolgálóhoz illeszthető internetes alkalmazások könnyen fejleszthetők.
- Az SQL Monitor az SQL-t használó alkalmazások kipróbálását, nyomkövetését és finomhangolását teszi lehetővé.
- A gyors hálózatos fejlesztéshez sok hálózati protokollra létezik előre elkészített elem.

A fenti felsorolás csak ízelítő abból a rengeteg lehetőségből, amit a Kylix kínál. Minden leendő fejlesztőnek javaslom a fejlesztőrendszer leírásának figyelmes tanulmányozását.

Adatkezelő alkalmazások fejlesztése

A jelenleg fejlesztett alkalmazások jelentős részének valamilyen adatbázist is kezelnie kell. A nyilvántartó rendszerek után egyre gyakrabban bízzák mérőrendszerek adatait, internetes médiák közléseit adatbázis-keresőkre, jó tulajdonságaiknak köszönhetően (megbízhatóság, kereshetőség, összetett lekérdezések és sorolhatnám). Mivel egy rendszernek akár többféle adatforrást is tudnia kell kezelni, szükséges egy olyan eszköz, amely a különböző típusú adatforrások elérését valamilyen szinten szabványosítja. Erre a feladatra a Borland korábban a BDE-t alkalmazta, de a rendszer kellemetlen tulajdonságokkal is rendelkezett. A Borland a rendszer tulajdonságainak javítására a dbExpresst fejlesztette ki. A Kylix adatkezelése már a dbExpressre épül, ez a korábbi, Windowson fejlesztett alkalmazások átírása során gondot is okoz (lásd még később).
A dbExpress kifejlesztése során a következő szempontok voltak a legfontosabbak:



- a rendszer méretét és erőforrásigényét tekintve kicsiny legyen,
- legyen a lehető leggyorsabb,
- tegye lehetővé a könnyű fejlesztést,
- legyen operációsrendszer-független,
- könnyen lehessen új meghajtókat írni hozzá.

A rendszer meghajtóprogramjai kicsik és gyorsak, mert szolgáltatásaik szándékosan behatároltak. A dbExpress-szel a fejlesztő az SQL adatbázis-kezelők összes lehetőségét közvetlenül kihasználhatja. Az adatbázis-kezelő alrendszer néhány tulajdonságának megértéséhez szükséges betekintést nyernünk a rendszer elvi működésébe is. Az adatok a következő úton jutnak el az alkalmazás felületére (lásd az *ábrán*):

- Az SQLConnection-elem összekapcsolja az alkalmazást az adatbázishoz megfelelő dbExpress meghajtóval.
- A dbExpress valamely adatkezelő eleme szükség esetén az adatbázis-kiszolgálón lekérdezést vagy tárolt eljárást hajt végre.
- A DataSetProvider-elem a ClientDataSet kérésére a szükséges parancsokat az előző elemek valamelyikével végrehajtatja, és a kapott adatokat továbbítja a ClientDataSet-elemnek.
- A ClientDataSet a kapott rekordokat a memóriában tartja, melyek rajta keresztül megjeleníthetők vagy módosíthatók. A módosítások a memóriában tárolódnak és az ApplyUpdates tagfüggvény hívásával elküldhetők a DataSetProvidernek. Ez az elem ilyenkor indítja el a tranzakciót a kért módosítások véglegesítésére. Hiba esetén a tranzakció törlődik és a ClientDataSet értesítést kap a hibáról.

A fenti módszer előnyei:

- Lehetővé válik, hogy a tranzakciók idejét a lehető legkisebbre szorítsunki, így csökkenthetjük az adatkiszolgáló terhelését.
- Szükség esetén a rendszer eszközöket ad összetett lekérdezések (többtáblás) szerkesztésére is.
- Mivel a ClientDataSet a memóriában tárolja a lekérdezett adatokat, azok gyorsan újrarendezhetők és szűrhetők az SQL kiszolgáló felesleges terhelése nélkül.
- A ClientDataSet önmagában szolgál néhány SQL-hez hasonló szolgáltatással, ez szintén csökkenti a feldolgozási időt és az SQL kiszolgáló terhelését. A memóriában lévő sorokon összegzéseket lehet végezni, erre a következő függvények használhatók: Sum, Min, Max, Count, Avg. Az SQL WHERE feltételéhez hasonlóan lehet elvégezni a sorok szűrését.

Természetesen a memória használatának megvannak a határai. Ha a lekérdezés eredménye túl nagy mennyiségű adat, az SQL kiszolgáló és az ügyfél közötti hálózat erősen túlterhelődhet. Ha azonban egy lekérdezés ésszerűen felépített és adatait többször is fel lehet használni, a fenti lehetőségek nagyon hasznosak. Jelenleg a rendszer Desktop Developer változata a MySQL és az InterBase, a Server Developer-változat az Oracle és DB2 SQL adatbázis-kezelőket támogatja, mivel azonban a MySQL és InterBase-meghajtók forráskódja mintaként adott, könnyen fejleszthető a rendszerhez saját megbízható, hatékony meghajtómodul. Tudomásom szerint a PostgreSQL-hez ezt a meghajtót jelenleg is fejlesztik. Helyi gyorsítótáblák létrehozására a Borland a MyBase adatbázisrendszert alakította ki. Ez a memóriatáblák használata során nagy sebességet tesz lehetővé, és a könnyebb felhasználhatósághoz a bináris mellett képes XML formátumban is állományba menteni az adatokat.

Fejlesztés több operációs rendszerre

A korábban Delphiben fejlesztett programok átvitele Linux alá nem zökkenőmentes. Aki ismeri a Delphi-rendszert, az tudja, hogy úgynevezett VCL (Visual Component Library) elemekre épült. Ezek között sok olyan is akad, amelyek a Windows valamely egyedi lehetőségére épülnek (OLE, MTS stb.). Mivel ezek Linux alatt nem állnak rendelkezésre, így a velük készült alkalmazások átültetése nehézségekbe ütközhet, rossz esetben a program bizonyos részeit akár teljesen át kell írni. Míg Windows alatt a programok az alacsony szintű műveletek elvégzéséhez gyakran hívják segítségül a Win32 rendszerhívásokat, ezek Linuxon nem léteznek. Az ilyen hívásokat használó program Linux-rendszeren módosítás nélkül nem futtatható. A programok Windows alatt beállítási adataikat gyakran a Rendszerleíró adatbázisban (Registry) tartják, míg Linuxon nincs a Registryhez hasonló központi beállítási adattár. Az itt említett példák mutatják, hogy milyen nehézségek merülhetnek fel a több felületre történő fejlesztéskor. Sokfajta kisebb-nagyobb különbség miatt az eddig fejlesztett programok átvitele Linuxra gyakran igen költséges lehet. További gondok forrása lehet a Delphi 5-ig elterjedt adatelérési eljárás, a BDE (Borland Database Engine). Mivel ez a rendszer nem létezik Linuxra, így a vele fejlesztett programokat át kell írni.
A Kylixban a Delphi 5-ig használt elemek jó részét újraírták. Az új rendszer neve CLX (Component Library for X-Platform). A CLX objektumait úgy tervezték, hogy azok módosítás nélkül vagy a lehető legkisebb módosítással a különböző operációs rendszerek között átvihetők legyenek. A Delphi 6-os változatában is megtalálhatjuk ezeket a CLX-elemeket, így már lehetőség nyílik rá, hogy több operációs rendszeren futó programokat fejlesszünk, de csak akkor, ha az átvihetőséghez szükséges szabályokat betartjuk.

Összegzés

Amennyiben a fejlesztendő program grafikus vagy adatfelülettel érintkezik, mindenképpen javaslom a Kylix használatának megfontolását. Minden cégnek alapvető érdeke, hogy fejlesztései hatékonyak legyenek. A rendszer összevont fejlesztőfelülete gyors és kényelmes munkát tesz lehetővé, használatának elsajátítása akár önállóan is viszonylag egyszerű. A fejlesztők és a felhasználók közös érdeke, hogy a programba a lehető legkevesebb hiba kerüljön, és egy esetleges hiba felfedezése esetén gyorsan ki lehessen deríteni az okát, és el lehessen hárítani a bajt. Ez egy szerényebb tudású rendszerrel nehézkesen kivitelezhető. A Kylixnak megvannak ugyan a maga kisebb gyermekbetegségei, de hiszem, hogy ezeket ki fogja nőni. Végre a grafikus fejlesztőeszközök új nemzedéke indult útjára Linuxon is.

Mátó Péter (atya@andrews.hu) korábban több évig lelkes Delphi-fejlesztő volt. Miután főállású Linux-tanácsadó lett, kisebb grafikus fejlesztésekhez hosszasan keresett hatékony grafikus fejlesztőeszközt. Eddig sikertelenül.

Kapcsolódó címek

A Kylix-rendszer kifejlesztője a Borland Software Corporation
➲ http://www.borland.com
A Kylix-rendszer
➲ http://www.borland.com/kylix
Kiegészítő eszközök és elemek
➲ http://borland.com/kylix/resources/kylixtools.html
Nyitott forráskódú fejlesztőeszközök (Zeos library)
➲ http://www.zeoslib.org

Maya 3 már Linuxon is!

A világ legkomolyabb háromdimenziós játékfejlesztő, filmes és hatáskészítő programja, a Maya 3 végre Linuxon is elérhető.

Az Alias/Wavefront 2001. március 22-én bejelentette: a Maya 3 programcsomag fejlesztése RedHat Linux operációs rendszerre sikeresen befejeződött. A felhasználók különböző csoportjait megcélozva három programcsomagot dobtak a piacra – a Maya Buildert, a Maya Complete-et és természetesen a Maya Unlimitedet. Ezek jelenleg a RedHat 6.2-es és ennél későbbi változatokon futnak.
Azért esett a választás a RedHatre, mert ez a változat az egyik legelterjedtebb és -támogatottabb Linux-változat, számos cég és stúdió ezt használja. A Maya linuxos változatának megjelentetésével az Alias/Wavefront tulajdonképpen a játékfejlesztő és filmes, illetve utómunkálatokat végző stúdiók hatalmas igényét elégítette ki.

Maya

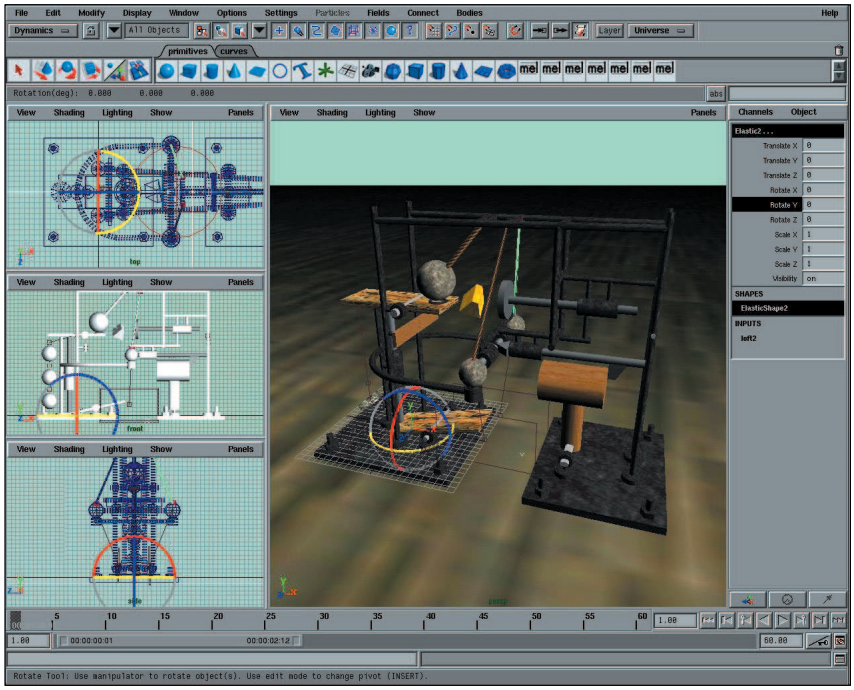
Az Alias/Wavefront ➲ www.aliaswavefront.com, a képi hatások és animációs programok fejlesztése terén piacvezető cég 1995-ben hozzálátott egy új programnemzedék megalkotásához, amellyel az ötletes felhasználás és az eredményesség közötti korlátot szándékozott áttörni.
A program fejlesztése során a következő kulcsfontosságú célokat tűzték ki. *Egyszerűen kezelhető felületet kívántak alkotni*, hogy a hagyományos animációval foglalkozók szintén könnyen kezelhessék a programot, ne kelljen még a számítógép használatának rejtelmeiben is elmélyülniük. Az Alias/Wavefront a Mayához rengeteg olyan szakmai trükköt használt fel, amelyek *a rendszer gyorsaságát a végletekig kihasználják*. Az objektumközpontú C++ programozás, valamint az OpenGL-es megvalósítás adja a legnagyobb teljesítményt mind a számolási, mind a leképezési (rendering) folyamatokhoz (*1. kép*).
Az előbbi tulajdonságok elérésére a következő felépítést választották. A Maya gyakorlatilag egy C++ban fejlesztett valós idejű parancsfeldolgozóként fogható fel. Ez azt jelenti, hogy szinte az összes eszköz a saját, MEL nevű nyelvén íródott. Ez megkönnyíti a fejlesztést, másrészt (a programozási ismeretekkel magas szinten nem rendelkező) felhasználók számára is páratlan rugalmasságot nyújt. További rugalmasságát a Dependency Graph, a Maya csomópontalapú szerkezete adja, amely rendkívül megkönnyíti a tájékozódást és a vezérlést.
A komolyabb fejlesztők rendelkezésére áll még a Maya C++ API-ja, ennek segítségével a programot olyan képességekkel ruházhatják fel a fejlesztők, amilyenekkel csak akarják. Az átgondolt felület segítségével akár a kezdők is gyorsan elsajátíthatják a program kezelését.
A készítők a Maya tökéletesítése során azt a több mint tíz évnyi tapasztalatot is felhasználták, amit a három cég – az Alias, a TDI és a Wavefront – az animációs programok fejlesztése során szerzett.
A programcsomag első változatát csupán a legnagyobb filmes cégek IRIX-es munkaállomásainak felhasználói élvezhették. Pár hónappal később azonban már megjelent az NT-változat, PC-s guruk számára. Megjegyzem, akkoriban (1988) egy erőgépnek számító masinán lehetett csak futtatni: Pentium 166 MHz, OpenGL és 96 MB RAM kellett hozzá. Azóta az Alias/Wavefront már a négyes változatnál tart, maga mögött hagyva hat kiadott sorozatot, amelyekben mindig egy-egy új lehetőség jelent meg, vagy egyes régi eszközöket finomítgattak, csiszolgattak.
A cégcsoport háromféle Maya-csomagot szállít Linuxra, a felhasználási céltól függően. Természetesen ezek árban rendkívül eltérnek egymástól (csomagonként, illetve országonként az ár 2995 dollártól csaknem húszezer dollárig változhat).

Maya Builder

A „legegyszerűbb” program a Maya Builder, amelyet játékfejlesztőknek szántak. Hatékony poligonmodellezési és mintázatkezelő rendszert építettek bele. Ezenkívül teljes elérést biztosít a Maya API-hoz, valamint a MEL-hez (Maya Embedded Language) – C++ alapú programozható felületet nyújtva.
A Maya Builder rendkívül gazdaságos eszköz a sok munkaállomást igénylő játékfejlesztő cégek számára.

Maya Complete

A következő csomag a Maya Complete. Ezzel a programmal már világszínvonalú animációkat és képi hatásokat készíthetünk,



1. kép A Maya felülete

sok-sok gyakorlás után. Csodálatos részecskerendszerével például rövid idő alatt vihart, esetleg robbanást állíthatunk elő. A *Rigid-soft body* segítségével szilárd, illetve lágy testek ütközéseit, kölcsönhatásait utánozhatjuk a fizika törvényeinek megfelelően, vagy éppen azokat teljesen megcáfolva.
Ha például filmünkön egy falevél lehullását szeretnénk végigkísérni, csak kapcsolatba hozzuk a levelet a tömegvonzással, a széllel, és attól kezdve programunk szinte mindent elvégez helyettünk.

Maya Unlimited

A Maya csúcsának is lehetne nevezni a Maya Unlimitedet. Mint nevéből is kiderül, a fejlesztők arra törekedtek, hogy ne legyen olyan nehézség, amit e csomaggal ne lehetne megoldani. Mindazt tartalmazza, amit az előző két program, ráadásul olyan modulokat építettek bele, mint például a *Cloth*, amellyel ruhák, anyagok, vásznak mozgását tudjuk valósághűen bemutatni és akár animálni is; vagy a *Fur*: ez haj, szőrzet, esetleg fű készítésére szolgál.
Az utóbbi csomagokban még seregnyi nagyszerű dolgot találunk, például az *Artisant*, amellyel egyszerűen modellezhetünk, illetve festegethetjük tárgyainkat; valamint a *Paint Effects*-et, ami egyben két-, illetve háromdimenziós festőprogram. Itt több száz ecset közül választhatunk (a tengeri növényektől, az emberi ujjakon át a villámokig), és ezeknek körülbelül háromszáz különböző beállítását változtathatjuk meg, illetve animálhatjuk időben. Nem elhanyagolható

2. kép Paint Effectsszel festett és animált kép

programrész a Maya Live sem, ez a filmes utómunkában elterjedten használt (3D camera tracking matchmove) megoldás (*2. kép*).

A Maya üzembe helyezése

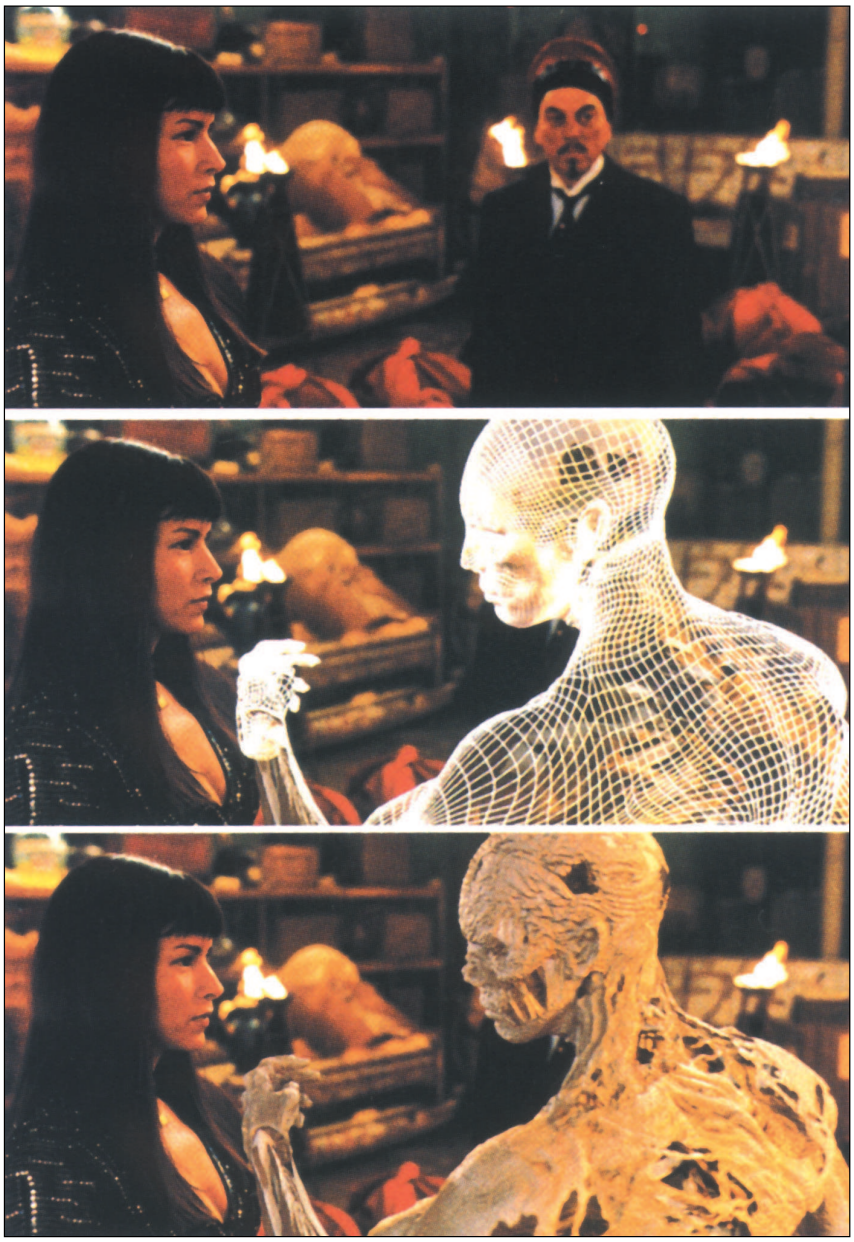
A program telepítése rendkívül egyszerű feladatnak bizonyult. A korongon három RPM fájlt találunk, amelyekből körülbelül tíz perc alatt futtatható állapotúvá tudjuk varázsolni a Mayát.
Más a helyzet a Linuxszal. E kiváló programcsomaggal végezhető eredményes munkához szükségünk lesz az árnyalt és a mintázatot mutató nézet használatára, a fények profi alkalmazására. Ehhez egy viszonylag izmos grafikus kártya kell, amely Linux alatt megfelelően működik. A „házilag” összekovácsolt gépeknél még a RedHat használatakor is szinte csak az nVidia kártyák (TNT, TNT2, GeForce 1, 2, 3) jöhetnek szóba.
Aki viszont komolyabban foglalkozik 3D-s grafikával, beszerezhet egy nagyobb munkaállomást, például HP FX10-es vagy sgi Visual Workstationt, ezeket már RedHattel és komoly grafikai rendszerrel látták el (Quadro2 Pro, esetleg Fire GL2 grafikus processzorok).

Visszhang

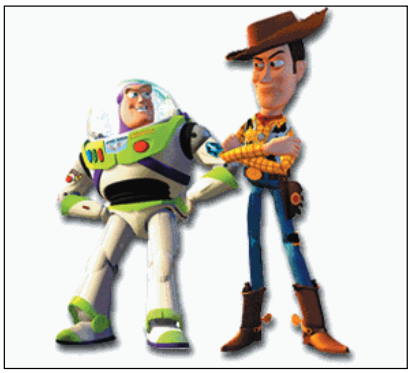
„Megörültünk, amikor megtudtuk, hogy ezután a Mayát Linuxon is használhatjuk” – mondta *Michael Babcock*, a Henson stúdió játékfejlesztője. „Teljes hálózatunk linuxos, és elégedettek vagyunk a megbízhatóságával, a rugalmasságával és a biztonsági szolgáltatásaival. Most, hogy a Maya rendkívül erős karakteranimációs eszközei közvetlenül elérhetők Linuxon is, a Henson Digital Performance Studio lehetőségei teljesebbé váltak.”

3. kép Részlet az ILM által készített Csillagok háborújában látható versenyből

„A játékfejlesztők örömmel fogadták a hírt, hogy ezután Linuxon is hozzáférnek a Mayához” – nyilatkozta Makato Osaki, a CSK Fejlesztési Központ AM2 részlegének (a SEGA vezető részlege) igazgatója. „A bétateszt során rendkívül elégedettek voltunk a linuxos Maya teljesítményével és biztonságával, amit a Unix-alapú rendszermag szavatol.”
A Maya rendkívüliségére a legjobb bizonyíték az, hogy a legnagyobb filmes és utómunkálatokat végző stúdiók saját fejlesztésű programjaik mellett egyre gyakrabban választják a Maya nyújtotta lehetőségeket.
A leghíresebb mozifilmes és videós cégek, amelyek az Alias/Wavefront termékeit használják: az ABC, a Blue Sky/VIFX, a CBS, a Cinesite, a CNN, a Digital Anvil és a Digital Domain (filmjeik: Apollo 13, Armageddon, Titanic, X-Men) és nem utolsósorban a Dream Quest Images, a Dream Pictures Studios, a Dreamworks (Z, a hangya), valamint az Industrial Light & Magic (Star Wars Episode I – *3. kép;* A múmia visszatér – *4. kép*).
A referenciák között említhetjük még a Matteworld Digital, a Metrolight Studios, a Manex Visual Effects (Mátrix, Mission Impossible 2, Truman Show), illetve az NBC, a Pacific Data Images és a Pixar cégeket (Toy Story – Játékháború – *5. kép*, Egy bogár élete). Nem feledkezhetünk meg az olyan klasszikusokról sem, mint The Walt Disney



4. kép Industrial Light & Magic – A múmia visszatér



5. kép Buzz Lightyear és Woody

6. kép Bigyó felügyelő

7. kép A Sacra Coronában a mayás angyalok hozzák az égből a koronát

8. kép Kép az Alcatrazból

Soa Lee – Monna

Company, a Sony Pictures Imageworks, a Santa Barbara Studios, valamint a Stardust (Stuart Little, Harry Potter, Godzilla, Anaconda, Speed, Bigyó felügyelő – *6. kép*). A játék- és multimédiás fejlesztő cégek között megtalálható a CAPCOM, az Electronic Arts, az Interplay, a NAMCO, a Nintendo, a SEGA, valamint a Virtual Worlds Entertainment.

Mayás rendezvények

Tavaly decemberben a Planetáriumban rendezték meg a 3December nevű eseményt, a Leonardo SNS szervezésében.
Az Alias/Wavefront a budapestihez hasonlóan még húsz másik nagyvárosban szervezett 3D-s találkozót, ahol a Maya legújabb szolgáltatásaival ismerkedhettek meg az érdeklődők. Bemutatkoztak még a hazai animációs stúdiók és játékfejlesztő cégek is. Többek között a Front Film nevű cég, amely vezető szerepet tölt be a magyar filmes és utómunka területén. Ők gyártották például a Sacra Coronában látható a látomás-jelenetet (*7. kép*), illetve a lovasok digitális sokszorosítását ➲ http://frontfilm.hu.
A játékfejlesztő cégek közül a Philos Laboratories (a Theokracy linuxos változatának készítője) mutatta be legújabb fejlesztését, az Alcatrazt, amely teljes egészében 3D-s motorra épülő stratégiai akciójáték lesz. Fontos szerep jutott a Mayának a karakterek mozgásának, mintázásának kialakításában, továbbá a helyszínek is ezzel készültek

Tsohan Wu – Elhagyatott kunyhó

Fű szuperplánban

Iránytű

Fenra csodálatos kastélya

Pinhead

Bingo

Shrek, az ogre

(*8. kép*) ➲ http://www.philoslabs.com.
A cég ígérete szerint az idén is megismétlik ezt a találkozót, amit egyetlen „3D-s koponya” sem hagyhat ki.
A program rendszerkövetelményei: RedHat Linux 6.2 vagy későbbi változat, Pentium 200 MHz, 128 MB RAM, 3D-s videokártya, 1024x768-as, minimum 32 bites felbontás, és a legfontosabb a háromgombos egér (egy ilyen vason a Maya éppen csak elindul).
Az otthoni linuxos gépek többségének 3D-s teljesítménye sajnos még mindig nem éri el az IRIX-os és a Windows NT-s számítógépekét, de biztosak lehetünk benne, hogy ez a gond a közeljövőben megoldódik. Aki foglalkozott valaha háromdimenziós grafikával vagy egyéb animációk készítésével, annak feltétlenül meg kell ismernie a Mayát. Olyan csúcsteljesítményt nyújtó programról van

Szamár és Shrek

A Pixar legújabb animációs kisfilmjének, a For the Birdsnek a főszereplője

szó, amely mindenki elképzelését képes valóra váltani. Vitathatatlanul a legkomolyabb és a leginkább felhasználóbarát a maga osztályában.
A CD-mellékleten mayával készült filmek és animációk találhatók a magazin/maya könyvtárban.

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

Kapcsolódó címek

- ➲ http://www.aliaswavefront.com
- ➲ http://www.ilm.com
- ➲ http://www.d2.com
- ➲ http://www.mvfx.com
- ➲ http://www.pixar.com
- ➲ http://imageworks.com
- ➲ http://www.philoslabs.com
- ➲ http://www.frontfilm.hu
- ➲ http://leonardo.sns.hu

Tanuljuk meg a PHP 4 használatát 24 óra alatt

Matt Zandstra, a Corrosive Web Design cég egyik vezető programozója újabb világhálós programozási könyvvel jelentkezett.

A szerző rajong a parancsnyelvekért: PHP, Java, JavaScript, Perl, Lingo és AppleScript nyelven fejlesztett már programokat, a HTML-t, a JavaScriptet, a Perlt és a PHP-t pedig tanfolyamokon oktatta is.

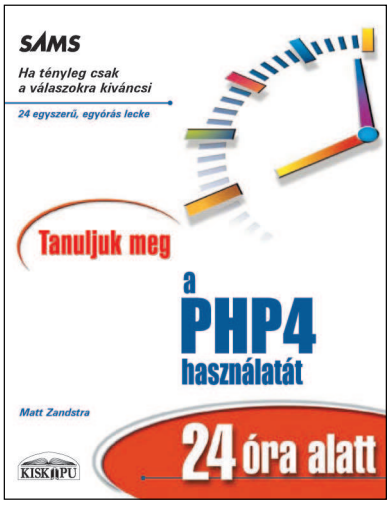
A Tanuljuk meg a PHP4 használatát 24 óra alatt – mint címe is mutatja – a Perl, az ASP és a Java mellett legnépszerűbb webes parancsnyelvről, a PHP-ről szól. A kezdőknek szánt sorozat célkitűzéseinek megfelelően nem teljes programozási útmutató, nem térképezi fel a PHP összes lehetőségét, nem ismerteti annak valamennyi eljárását, csupán elindít az úton, mindazonáltal elegendő anyagot tartalmaz ahhoz, hogy a PHP-t webes alkalmazásainkban felhasználhassuk, akár tapasztalt programozók vagyunk, akár nem.

A szerző célja, hogy megmutassa, mire is használható a PHP, hogyan juthatunk el a segítségével a személyes honlapok szintjéről a profi portálok megtervezéséig. Bár belépő szintű könyv, ne feledjük, szakembereknek, tevékeny és leendő webprogramozóknak szól, így a leckék megértéséhez legalább ilyen irányú érdeklődés szükséges, de az sem árt, ha a programozási alapfogalmakkal tisztában vagyunk. A kötet ugyan összefoglalja a leggyakoribb programozási szerkezeteket: a függvények, tömbök, objektumok használatát, de a bemutatott példák mélyebb megértéséhez programozói gondolkodásmód szükséges. A könyv nem az önmagukat az Interneten bemutatni kívánó laikusoknak íródott.

A PHP a kiszolgálói alkalmazások, a webes adatbázis-kezelő programok nyelve, így értelemszerűen ezek elkészítése áll a középpontban. A kötet végén két teljes fejezetet szántak egy működő példaprogram részletes bemutatására. A huszonharmadik óra a helyi klubok adattárának és „programfüzetének” feladatát ellátó alkalmazás megtervezésével foglalkozik, a felhasználók bejegyzéséhez és a klubok programjának beviteléhez szükséges elemeket tárgyalja. A huszonnegyedik óra a látogatók számára készült felület megvalósításához szükséges kódot tartalmazza, amely lehetővé teszi a klubok programjainak böngészését.

Amíg azonban eddig eljutunk, meg kell ismerkednünk a PHP történetével, a 4-es változat újdonságaival és telepítésével (első fejezet), a környezet beállításával (második fejezet), a rendszer alkotóelemeivel, illetve az adatbázis-kezeléshez nélkülözhetetlen eljárásokkal, az űrlapok használatával, a fájlok és adatok kezelésével.

A harmadik óra bemutatja, miként építhetünk PHP-kódot a HTML-oldalakba, illetve hogyan készíthetünk a böngészők számára kimenetet adó programot. A negyedik óra a PHP alapjaival ismertet meg. Az óra anyagát a változók, az adattípusok, a műveletjelek (operátorok) és a kifejezések képezik. Az ötödik óra a programok futását vezérlő elemek utasításformájával (nyelvtanával) foglalkozik. Az if és switch szerkezetek mellett megtanuljuk a for és while ciklusvezérlő elemek használatát is. A hatodik óra a függvények készítését és használatát tárgyalja, a hetedik a lista jellegű adatok tárolására használható tömb adattípussal foglalkozik, valamint a tömbök használatát segítő PHP 4 függvények közül ismertet néhányat. A nyolcadik óra a PHP 4 osztály- és objektumtámogatását mutatja be.



Szerző: Matt Zandstra
Terjedelem: kb. 540 oldal
Kiadó: SAMS
Magyar kiadás: Kiskapu Kft., 2001.
E-mail: kiskapu@kiskapu.hu
Honlap: ➲ www.kiskapu.hu
ISBN: 963 9301 30 2

A kilencedik óra a HTML-űrlapok használatát, vagyis a felhasználótól érkező adatok feldolgozását vezeti be. Ezt követik a fájlok és könyvtárak kezelésének lehetőségei, majd a PHP DBM-támogatása. (Ez azért lényeges, mert a DBM-támogatás a legtöbb operációs rendszeren elérhető.) A tizenkettedik óra az SQL alapjait tárgyalja, valamint bemutatja a PHP 4 MySQL adatbázisok kezelésére szolgáló függvényeit.

A tizenharmadik óra a külvilággal való kapcsolattartásé: a HTTP-kéréseket veszi szemügyre, illetve a PHP hálózati függvényeit taglalja.

A tizennegyedik óra a GIF, PNG és JPG képek készítését lehetővé tevő függvényeket, a dinamikus képkezelést mutatja be. A tizenötödik óra a dátumműveletekhez használatos függvényeket és eljárásokat ismerteti, míg a tizenhatodik visszatér az adattípusokhoz és az addig nem említett, de a hatékony használathoz elengedhetetlen további függvényeket mutatja be. A tizenhetedik óra a karakterláncok kezeléséhez jól használható PHP-függvényekkel foglalkozik, a tizennyolcadik pedig a bonyolultabb minták keresésére és cseréjére szolgáló szabályos helyettesítő kifejezéseket ismerteti. A tizenkilencedik óra a különböző programok és HTTP-kérések közötti adatáramlás néhány módját mutatja be. A huszadik óra az előző órában tanultakat a PHP 4-es beépített munkamenet-kezelő függvényeivel bővíti ki. A huszonegyedik óra a külső programok futtatását és kimenetük felhasználásának lehetőségeit tárgyalja.

A kész alkalmazások ellenőrzésével, a hibakereséssel külön fejezet foglalkozik – nem véletlenül, hiszen az ilyen bonyolult, a felhasználókkal interaktív kapcsolatot tartó programoknak bármilyen eshetőségre fel kell készülniük…

A gyors megoldásokat a sorozatban megszokott, külön ikonnal jelzett „Tippek” mutatják be, a részletes magyarázatokat a „Megjegyzésekben” találjuk, a „Figyelmeztetés” ikonnal ellátott részek pedig a leggyakrabban elkövetett hibákra hívják fel a figyelmet.

Rézműves László
1971-ben született Budapesten, eredeti szakmája szerint angoltanár, de az ELTE-n történelem és elméleti nyelvészet szakra, valamint afrikanisztika programra is járt. Rengeteg hobbija van: zenélés, filmezés, foci, illetve gomfoci, programozás. Jelenleg egyszerre dolgozik lektorként, nyelvi tanácsadóként, szerkesztőként, személyzetisként és fordítóként.

Programcsemegéző

SteelBlue

Ha valaha is használtad az Allaire ColdFusion nevű programját, talán érdemes vetned egy pillantást a SteelBlue-ra is. Ez az érdekes webes alkalmazás lehetővé teszi, hogy SQL lekérdezéseket programozz SQL kiszolgálóra, űrlapokkal dolgozz, vagy számos egyéb feladatot végezz el. A ColdFusionhöz hasonlóan a SteelBlue is számos kiegészítéssel látja el a HTML-nyelvet. Van azért egy nagy különbség köztük, mégpedig hogy a SteelBlue parancssorból is futtatható, így használatához nincs szükség webböngészőre. A komolyabb webes programozók számára nélkülözhetetlen eszköz. A futtatásához: webkiszolgáló, libpq, libstdc++, libm, libcrypt, glibc, SQL kiszolgáló (PostgreSQL, MySQL, msql vagy egyéb) szükséges.
➲ http://www.steelblue.com/

IP Flow Meter

Nemrég úgy éreztem, mintha valami felzabálná az internetkapcsolatomat, viszont nem tudtam rájönni, hogy ki vagy mi. A tcpdump futtatása kézenfekvő megoldás lett volna, de éppen nem volt kéznél egy nagyméretű merevlemez, és időm sem volt arra, hogy átrágjam magam a kiíratás eredményén. Egyszerűen tudni szerettem volna, hogy hol kezdjem a keresgélést, ugyanis ha megvan a kiindulópont, azt követően már át tudom nézni a protokollokat. Rátaláltam az IP Flow Meter programra, ami pontosan azt teszi, amire nekem szükségem volt. Az első kiíratást követő egy órán belül könnyedén szemmel tarthattam a rendszer ki- és bemenő forgalmát. A futtatáshoz szükséges: libpcap, glibc.
➲ http://www.via.ecp.fr/~tibob/ipfm/

w3perl

Olyan naplómegjelenítő programra van szükséged, amit még a korlátozott értelmi képességűek is azonnal megértenek? Megtaláltad, amit kerestél. A rengetegféle módon megjeleníthető ábrák jelentését még a hatéves gyermekek is felfogják. A telepítése is rendkívül egyszerű. Azt hiszem, ez az egyik legjobb webkimutatást készítő alkalmazás, amit valaha láttam. A futtatáshoz szükséges: Perl, webböngésző, webkiszolgáló, cron (igény szerint), Telnet vagy SSH (igény szerint).
➲ http://www.w3perl.com/

dhcpstatus

A dhcpstatus segédprogram egyedüli apró hátránya, hogy használatához a DHCP-rendszeren webkiszolgálóra is szükség van. Érdemes esetleg keresni egy kisebb Perl-webkiszolgálót, amely képes a Perl parancsfájlok futtatására – legalábbis jómagam ilyet fogok keresni. A dhcpstatus által megjelenített adatokat akár percekig is böngészhetnénk, keresgélhetnénk a leases.dhcp állományban. Természetesen mindezt a Webmin segítségével is megtehetjük, ha feltelepítettük, ám ez kicsit olyannak tűnik, mint szuperszámítógépet használni olyan feladathoz, amit az xcalc segítségével hamarabb meg tudnánk oldani. Futtatásához Perl, webkiszolgáló a DHCP kiszolgálórendszeren, és webböngésző szükséges.
➲ http://dhcpstatus.sourceforge.net/

fwipe

Az fwipe apró segédprogram, mely a kiszemelt állományokat biztosan törli. Első lépésként ötször teleírja az állományt nullákkal, másodikként ezt megismétli egyesekkel, majd törli az áldozatot. Mindehhez természetesen írási engedéllyel kell rendelkeznünk a kiválasztott állományhoz. Utólag bármilyen módszerrel is próbálják visszanyerni a fájlrendszerből, az eredmény csupa egyes lesz. A program olcsón elérhető biztonságot nyújt és használata is egyszerű. Futtatásához glibc szükséges.
➲ http://www.pobox.com/~lbudney/linux/software/fwipe.html

arb-scan.pl

Távoli reklámcsíkpásztázó, mely végigpróbálgatja a reklámcsíkokat a célállomáson, majd tetszőleges kiszolgálóról le is menti őket a későbbi felhasználáshoz. Segítségével ellenőrizhető, hogy a reklámcsíkok megfelelnek-e a cég szabályzatának. A levelezőkiszolgálókat is megvizsgálja, hogy támogatják-e az ellenőrzési és terjesztési műveleteket (ami kifejezetten rossz ötlet). A futtatáshoz szükséges: Perl, IO::Socket Perl-modul, Nmap, dig, finger.
➲ http://arbon.elxsi.de/

David A. Bandel
(dbandel@pananix.com) jelenleg Panamában él, Linux- és Unix-tanácsadással foglalkozik. Társszerzője a Que Special Edition: Using Caldera OpenLinux című könyvnek.