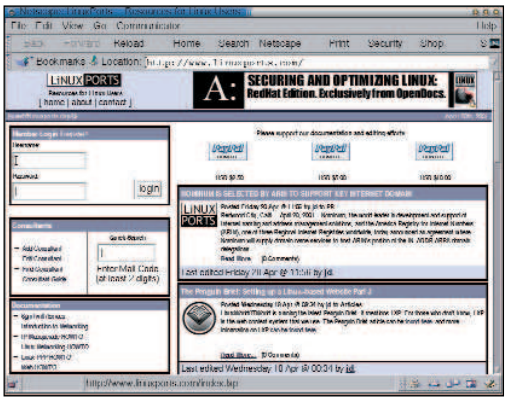
Tanácsadás magyar módra

„Az, ami vagy, már jutalom és büntetés is azért, ami vagy.”
Popper Péter Belső utakon

Szombat van és lapzárta. „Ne haragudj, hogy most zavarlak, de tudod, a cég bevállalt egy munkát, egy négygépes hálózatot kell összeraknunk, a központi gépre kell egy Linux, egy DOS-os könyvelőprogramhoz fájlkiszolgálónak…” – volt osztálytársam hív, egy számítógépeket forgalmazó cégnél dolgozik. „Nem is tudtam, hogy te is Linux-hívő lettél!” – ébredt fel bennem a gyanú. „Én? Dehogy! – válaszolta.
– Csak rám osztották ki a feladatot.”
Szóval itt tartunk. Bár még sokan nem értenek a Linuxhoz, azért mindenhol elkezdik használni. De vajon mennyi ideig tart majd, míg a titkárnők is megismerik, megszeretik, és legalább annyira otthon érzik magukat benne, mint más rendszerekben? Többen fel is háborodnak e gondolatra, mondván: „a Linux nem titkárnőknek való!”. Szerencsére már sok helyen bizonyították e kijelentés idejétmúltságát – ennek a sok helynek pedig van egy nagyon fontos közös vonása: főállású Linux-hívő dolgozik a közelben.



Hamarosan viszont a Linuxra nem kell majd annyit figyelni. Nem fordul majd elő, hogy a telepített OpenOffice bizonyos szolgáltatásai nem működnek, vagy esetleg az új változat telepítése kínszenvedés még gyakorlott rendszergazdáknak is. És ez az idő nincs is olyan messze.
De erre az időre fel kell készülnünk! A cégek oldaláról egyre többen jelentkeznek majd, hogy Linuxot akarnak használni, de nincs lehetőségük főállásban alkalmazni egy profi linuxost. És itt lépnek be a képbe a szaktanácsadók, távüzemeltetők és a megoldásszállítók. Lapunk jelenlegi számában az első kategóriával foglalkozunk, a szaktanácsadókkal.
És ez bizony nagyon kényes kérdés. Sokan mondják, hogy nem is kell foglalkozni a Linux reklámozásával, eladásával, hiszen egy szabad rendszerről van szó, aki használni akarja, majd letölti magának. Ez könnyen megoldható egy egyetemista számára, de egy üzletembernek, akinek minden óra drága, elfogadhatatlan, hogy ne legyen egy olyan embere, akire rábízhatja a Linuxhoz kapcsolódó összes feladatot. Egy üzletember nem akar maga bajlódni a rendszer üzemeltetésével, és ha a termelés zavartalansága érdekében költségei vannak, azt sokszor szívesen kifizeti.
Ehhez viszont olyan emberek kellenek, akik ténylegesen meg tudják oldani a felbukkanó feladatokat. És ez több lépésből álló folyamat. Először is meg kell érteni a gondot, beszélni kell tudni a felhasználók nyelvén. Másodszor tényleges szakmai felkészültséggel kell rendelkezni. És harmadszor, a munkánkkal arányos díjazást kell kérni. Ha ezeket végiggondoljuk, el tudjuk dönteni, hogy inkább egy főállást vállalunk el, vagy vállalkozóként kezdünk el tevékenykedni, mint szaktanácsadó. Akárhogy is, Magyarországon ez utóbbihoz nagyobb felkészültség kell, de több olyan kisebb, egy-három fős vállalkozás létezik, amely ebből él. Sajnos többször annyi kókler csapat is van: szakmai hozzá nem értésük ellenére komoly pénzeket kérnek el olyan szolgáltatásért, ami éppen csak azért működik, mert a dobozos ilyen vagy olyan Linux alapértelmezésként nyújtja azt. (Én például törlöm azokat a Címtárból, akikről folyamatosan rossz véleményt kapok, ugyanis igyekszem „megvédeni” a világot a silány szolgáltatást nyújtóktól.)
Véleményem szerint, amíg le nem tisztul ez a szakma is, és a „köznép” számára is könnyen azonosítható rendszer nem jön létre a minőségi szakemberek felismerésére, addig a szakembereknek kétszeresen nehéz a feladata. Nem hiszem, hogy jó megoldás volna valamilyen központosított vizsgarendszerhez kötni a „szaktanácsadó” képesítést, de igenis elkelne egy-két olyan fórum, amelyik kiáll a szakma érdekéért, és ha silány szolgáltatásra panaszkodik egy vevő, akkor (a helyzet elemzése után) kimondja, hogy Ikszipszilon Káefté csapnivaló lókötő, és semmi köze a Linuxhoz. Ehhez viszont egy ilyen fórum mögött lényegesen erősebb, népesebb és szakmailag képzettebb tömeg kell, hogy álljon. Előbb-utóbb csak kialakul egy erős mag, olyan szakembergárda, akikre a piaci szereplők is nyugodtan támaszkodhatnak.
Vajon hány olyan cég működik kis hazánkban jelenleg, amely hivatalosan is vállal linuxos rendszerfelügyeletet? Nos, a Consultants-HOWTO szerint összesen egy.
Véleményem szerint, ha valaki linuxos szakembernek vallja magát, legalább azt tegye meg, hogy feliratkozik egy-két ilyen listára, megmutatja magát! Erre a listára elméletileg bárki bármikor feliratkozhat (lásd a *Kapcsolódó címeket*). Sőt, ha valaki úgy érzi, hogy hajlandó és képes ilyen szolgáltatást nyújtani, iratkozzon fel honlapunkon a Címtárba!
Remélem, összegyűlik annyi cím, hogy érdemes lesz a Linuxvilág következő számában legalább egy oldalt megtölteni velük. Biztos vagyok benne, hogy tömérdek magánvállalkozó és pár fős betéti társaság dolgozik ebben a témakörben, olyanok, akik szakmai felelősséggel végzik munkájukat. Egyszerűen csak nem tudunk egymásról.

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

Kapcsolódó címek

A Consultants-HOWTO kereshető változata
➲ http://www.linuxports.com

Feliratkozás a Tanácsadók listájára
➲ http://www.linuxports.com/ldp/ldpadd_company.phtml

Programvadászat

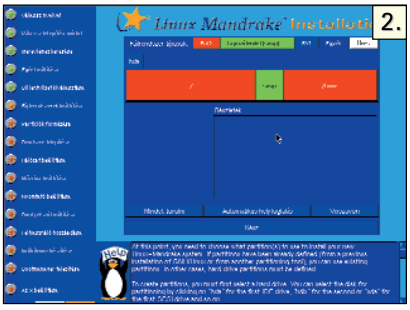
Mandrake Linux 7.2

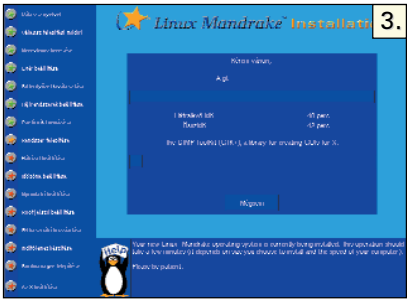
Magazinunk mellékleteként a két korongból álló Mandrake Linux 7.2 Download Editiont tesszük közzé. Ez a Linux-változat a RedHat irányzatot képviseli, csomagkezelése RPM-alapú, a rendszermag és a csomagok i586 processzoron működnek hatékonyan, így nem futtathatók 486 vagy ennél kisebb processzorokon (bár van 486-osra fordított Mandrake változat is). A sok kényelmes és áttekinthető szolgáltatásának köszönhetően kiválóan alkalmas első Linuxnak.

A Mandrake telepítése

Az első lemez indítható, így ha számítógépünk képes CD-s rendszerindításra, nagyon könnyű helyzetben vagyunk. Amennyiben nem tudunk CD-ről indítani, akkor el kell készítenünk a telepítőlemezeket. Ezt a Windows/DOS alatt a CD-n található `rawrite` programmal tehetjük meg, Linux/Unix alatt pedig használhatjuk a `dd` parancsot. A rendszer elindulása után választhatunk a grafikus vagy a szöveges telepítés között. A grafikus telepítő könnyen áttekinthető és kényelmesen használható, a karakteres viszont sokkal kisebb étvágyú, így ha memóriaszűkében vagyunk, válasszuk az utóbbit.

A grafikus telepítőben a bal oldalon látható csillagokkal tudjuk nyomon követni az eddig elvégzett és a még hátralévő teendőinket: ahol a csillag zöld, azzal már elkészültünk, ahol piros, az még hátra van; a narancssárga az éppen folyamatban lévő feladatokat mutatja. Első lépésként válasszuk ki a telepítő nyelvét (*1. kép*). A magyar is a választható nyelvek között szerepel, bár jó néhány segítséget még csak angol nyelven olvashatunk. Ennek ellenére mind a kezdők, mind a megfelelő nyelvismerettel nem rendelkezők már nagyobb önbizalommal foghatnak hozzá a telepítéshez. A nyelv kiválasztása és a felhasználói szerződés elfogadása után következik a telepítési mód választása. A *Javasolt telepítési mód* a teljesen kezdőknek ajánlott, ekkor a rendszer mindent megpróbál helyesen beállítani, valamint a két lemezről egy előre elkészített programcsoportot telepít fel, ennek segítségével már szinte minden feladat elvégezhető (internetezés, szövegszerkesztés stb.). Az *Egyedi* kiválasztása sokkal nagyobb teret enged rendszerünk beállításához, a *Haladó* segítségével pedig az egész rendszert testre szabhatjuk. A merevlemez felosztásához

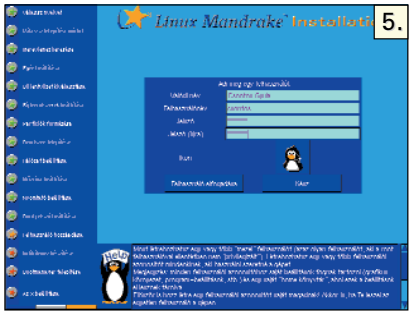




kérhetjük a telepítőtől, hogy az elérhető szabad területet használja, vagy itt is a *Haladó módot* választjuk, és magunk állíthatjuk be a lemezrészeinket (partícióinkat). Ha nincs különösebb igényünk, akkor nyugodtan hagyjuk a telepítőre ezt a feladatot. Ekkor három lemezterületet hoz létre: a /, a /home és a swap, azaz csereterületet (*2. kép*). Miután ezzel végeztünk, válasszuk ki a telepítés méretét. A *Javasolt* 700 MB-ot foglal el a merevlemezünkön, a *Minimális* pedig 300 MB-ot. Ezután hosszadalmas folyamatként a programok telepítése következik (*3. kép*), ennek időtartama a számítógép teljesítményétől és a memória méretétől függ.

Hálózat beállítása

Választhatunk a modem, az ISDN, az ADSL, a kábeles csatlakozás és a helyi hálózat



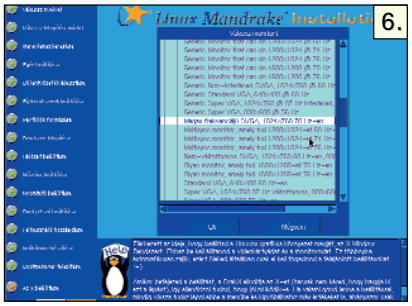
(LAN) közül. Amennyiben gépünk nem kapcsolódik másik géphez, akkor a hálózatot le is tilthatjuk. Mindegyik beállítási feladathoz kényelmes és egyszerű felületet kapunk, így nem jelenthet gondot a beállítás még a teljesen kezdők számára sem. Azonnal ki is próbálhatjuk a beállításainkat, amennyiben nem működnének, lehetőségünk van rögtön kijavítani, majd újra megpróbálni (*4. kép*).

A rendszer testre szabása

Ha a telepítés nyelvéül a magyart választottuk, akkor a program az időzóna beállításánál önműködően Budapestet választja ki. Ezután meg kell adni a rendszergazdai jelszót (a rendszergazda rendelkezik az összes jogosultsággal a rendszerben). Következő lépésként egyszerű felhasználókat adhatunk a rendszerünkhöz. Erre azért van szükség, hogy a mindennapi munkánkat ne rendszergazdaként végezzük, mert a teljes rendszerben az összes jogosultsággal rendelkezvén esetleg óvatlan pillanatunkban véletlenül tönkretehetjük Linuxunkat (*5. kép*). Az ikont nyugodtan cseréljük ki, így minden felhasználó saját „arcképet” kaphat. Miután befejeztük a felhasználók hozzáadását, készítsük el az indítólemezt. Erre akkor lehet szükségünk, ha a merevlemezünk elsődleges rendszerindító területe (az MBR, mely a rendszerindításért felelős) megsérül, vagy például a Windows telepítésekor a program kíméletlenül felülírja. Kiválaszthatjuk, hová szeretnénk telepíteni az indításkezelőt, az elsődleges lemezterületre, vagy a linuxos lemezterületünkre.

Az X beállítása

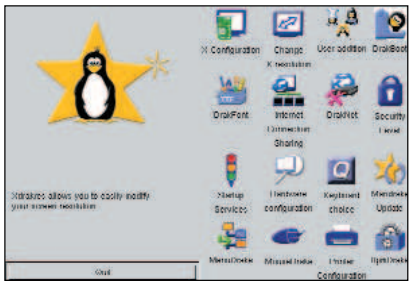
Elsőként a VGA kártyánk típusát kell kiválasztanunk, majd az ehhez szükséges fájlokat a telepítő a lemezről felrakja. A képernyő beállítása (*6. kép*) után a munkafelület és az alapértelmezett felhasználó megadása következik (*7. kép*). Az alapértelmezett



felhasználót csak akkor adjuk meg, ha a gépen rajtunk kívül senki sem dolgozik, mivel a grafikus felület a megadott felhasználóval indul el. Munkafelületek közül választhatjuk a KDE-t, a GNOME-ot, az IceWM-et stb. (*8. kép*). Ezután rendszerünket újraindítva a Grub indításkezelőben tudjuk elindítani a kívánt rendszert.

Összegzés

A Mandrake Linux 7.2 könnyen telepíthető és egyszerűen beállítható rendszert kínál a felhasználóknak. Telepítője áttekinthető és kényelmesen kezelhető (a magyar leírás sajnos még sok helyről hiányzik). Nálunk a gépben lévő összes alkatrészt sikerült helyesen felismernie, így mindenkinek nyugodt szívvel ajánljuk, hogy próbálja ki telepítését. Nagyon kellemesnek találtam a magyar nyelvű súgóoldalak telepítését



(*9. kép*). Ezeket a man szó és a program nevének begépelésével hívhatjuk elő.
A rendszer karbantartását legkönnyebben a DrakConf programmal végezhetjük el. Ez tulajdonképpen a beállításokhoz egy vezérlőpult, melynek segítségével minden fontos beállítást végrehajthatunk.
A menüpontok és jelentésük:

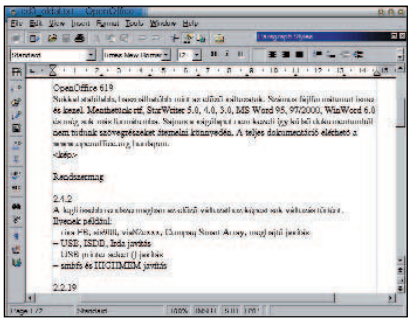
- XFDrake: X11 beállításai
- Xdrakres: X11 felbontás beállítása
- Userdrake: felhasználók karbantartása
- DrakBoot: az indulási feltételek megváltoztatása
- DrakFont: betűkezelő
- DrakGW: átjáró beállítása
- DrakNet: Internet-beállítások
- Draksec: a biztonsági szint kiválasztása
- DrakXServices: az induláskor elinduló szolgáltatások kiválasztása
- HardDrake: alkatrész-beállítások
- KeyboardDrake: billentyűzet beállítása
- Mandrake Update: a telepített csomagok frissítése
- MenuDrake: menübeállítások
- MouseDrake: egérbeállítások
- PrinterDrake: nyomtató
- RpmDrake: általános csomagkezelő

XFree86 4.0.3

Ez a változat az előzőhöz képest csupán hibajavítást és kódtisztítást tartalmaz, így nem teljes változat, hanem frissítésként jelent meg. Ezért elengedhetetlen egy működő 4.0.2-es X megléte a telepítésre kiszemelt gépen.
Kijavították az ideiglenes fájlok által okozott biztonsági hibát, valamint a Neomagic és a S3 Virge lapkakészletek meghajtóinak a hibáját. Az `XFree86-configure` parancs most már működik a SiS és i810-es lapkákkal is, javítottak a TrueType betűkezelésen, ezenkívül támogatja a GeForce 3-ast az nVidia meghajtóban.

OpenOffice 619

Sokkal megbízhatóbb és használhatóbb, mint az előző változatok. Számos fájlformátumot ismer és kezel. Menthetünk rtf, StarWriter 5.0, 4.0, 3.0, MS Word 95,



97/2000, WinWord 6.0 és még sok más formátumba. A vágólapot sajnos nem kezeli, így egy külső dokumentumból nem tudunk szövegrészeket átemelni. A teljes leírás elérhető a ➲ www.openoffice.org honlapon.

Rendszermag

2.4.2

A legfrissebb rendszermagban az előző változathoz képest sok változás történt. Ilyenek például:

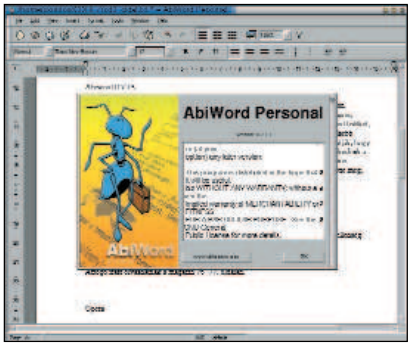
- Riva FB, sis900, via82cxxx, Compaq Smart Array, meghajtójavítás
- USB-, ISDB-, Irda- javítás
- USB printer select () javítása
- smbfs és HIGHMEM javítás

2.2.19

Korongunkon helyet kapott még a 2.2-es sorozat legújabb tagja a 2.2.19-es rendszermag is, mindazok számára, akik még nem békültek meg a 2.4-es sorozattal.
➲ http://www.kernel.org

Abiword 0.7.13

Az első, legszembetűnőbb – egyben leglekevésbé lényeges – változás a bejelentkező

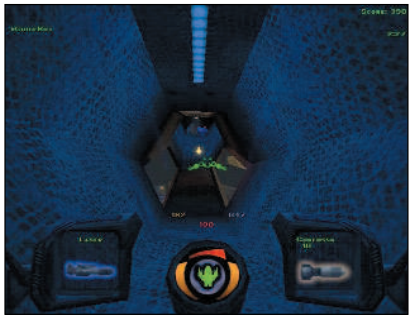


képernyő grafikájának a megváltozása. Javítottak a LibXML értelmezőn, működik a TTF nyomtatási lehetőség Unixon, képes beolvasni és menteni a HTML-listákat,

kibővítették a segítséget és a leírást, működik a PostScript nyomtatás, és javítottak a Word import/exportszűrőkön. A tapasztalataim azt mutatják, hogy nagy előrelépés történt az előző változathoz képest, viszont még mindig nem működnek a billentyűzet számpadján lévő karakterek, nem tudtam külső forrásból másolni, valamint egy rtf-ben elmentett fájlt újra megnyitva, minden formázás nélkül, teljesen összehúzva jelenítette meg.
➲ http://www.abiword.com

Descent 3 bemutatópéldány

Ez egy térben játszódó akciójáték, melyben mozgásunkat csak a falak korlátozzák. Repülhetünk fel, le, miközben menekülünk az ellenség elől, vagy amikor ő próbál menekülni előlünk (Magazinunk 76–77. oldalán átfogó írást olvashatnak erről a játékról.)
Legkisebb gépkövetelmény:
2.2.x-es rendszermag, legalább 200 MHz-es Pentium osztályú processzor, XFree86 3.3.x vagy újabb grafikus felület, OpenGL- vagy Glide-megfelelő VGA kártya, legalább 4x-es CD-ROM, 32 megabájt memória, OSS-megfelelő hangkártya, glibc 2.1, a kezeléshez billentyűzet, egér, de leginkább a botkormány ajánlott. Hálózati játékhoz TCP/IP kapcsolat szükséges.
Jó játékot!
➲ http://www.lokigames.com



Opera

Az Opera böngésző legújabb 5beta7-es változata is megtalálható a lemezen. Tapasztalataim szerint ez sokkal megbízhatóbban működik, mint elődje, de azért lesz még mit csiszolgatni rajta a fejlesztőknek.
Újdonságai és a hibajavítások:
PNG fájlkezelési hiba javítása, JPEG megjelenítési hibák javítása, összeomlás után a következő indításkor az előző állapot visszaállítása, jobb Javascript-támogatás, nagyméretű szövegek vágólapra másolása során sem fagy le.
➲ http://www.opera.com

Frissítések

Természetesen felkerültek a lemezre hibajavítások és frissítések is a különböző Linux-változatokhoz, így ismét naprakész rendszer tulajdonosai lehetnek azok is, akik nem rendelkeznek internetkapcsolattal.

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

Szupereladások a szuperszámítógépből

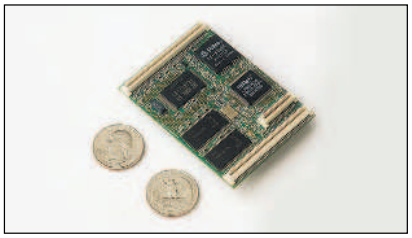
Az IBM S80 a valaha létezett legkelendőbb felső osztályú Unix-kiszolgáló – 16 hónap alatt háromezer fogyott belőle.

A United Airlines légitársaság nemrég vette át a háromezredik S80 kiszolgálót, ennek segítségével a cég nemzetközi pénzügyi rendszerét fogja működtetni.

A chicagói légitársaság bevételkezelő alkalmazásainak futtatásához három S80 kiszolgálót vásárolt.

Linuxos hitelkártya?

A 486CORE Module kicsi, beépített PC-megfelelő számítógép, mely minden olyan elemet tartalmaz, ami szükséges a DOS, a Windows CE, a Linux vagy a VxWorks operációs rendszer futtatásához. Hitelkártyához hasonló mérete lehetővé teszi, hogy bármilyen beágyazott rendszerben eredményesen használhassák.



Főbb jellemzői:

- teljes PC-megfelelés
- kis méret (54x75 mm)
- 486SX-megfelelő ElanSC400 processzor (33, 66 vagy 100 MHz-en)
- 8–32 megabájt DRAM
- 1–136 megabájt flashlemez
- CGA grafikus vezérlő LCD kijelzőkhöz
- PCMCIA vezérlő.

Az ára 98 dollártól kezdődik (ezer darab rendelése esetén).

➲ http://www.compulab.co.il/486core.htm

Wine

Megjelent a Wine legújabb változata. Segítségével windowsos programokat futtathatunk Linux/Unix alatt. A szabványos Windows rendszerhívásokat alakítja át Linux/Unix rendszerhívásokká. Sikeresen futtatták már a Microsoft Word 6, a Brice 4, a pasziánsz, a RealPlayer 5.0, a Lightwave 5.5 programokat. Az összes program kereshető adatbázisát a ➲ http://www.winehq.com/Apps/query.cgi oldalon találhatjuk meg. Jelenlegi állapotáról a ➲ http://www.winehq.com/News/status.html oldalon olvashatunk.

OpenSSH 2.5.2

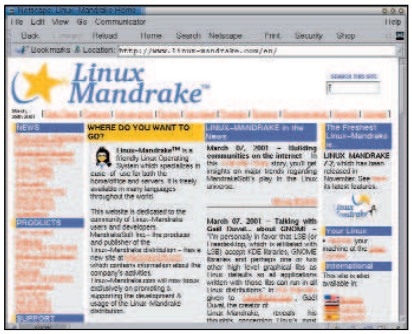
Megjelent az OpenSSH 2.5.2, mely támogatja az SSH1 és SSH2 protokollokat. Számos felhasználó, aki telnet, rlogin, ftp és hasonló programokkal dolgozik, nem is tudja, hogy jelszava titkosítás nélkül megy

át az Interneten, és ezt bárki lehallgathatja egy megfelelő gépről. Az OpenSSH titkosítja a gépek között zajló teljes forgalmat a jelszavakkal együtt, ezzel is csökkenti a támadás veszélyét és lehetőségét. Az OpenSSH csomag magában foglalja az ssh-t, mely a telnet és rlogin parancsokat váltja ki, valamint az scp-t, ez az rcp-t cserélte le a fájlmásolásnál és az sftp-t, mely az ftp-t teszi biztonságosabbá.

➲ http://www.openssh.com

Mandrake Linux

A CD-mellékletünkön közzétett Linux-változathoz néhány hasznos címet gyűjtöttünk. Reméljük, segítségükkel sikerül megkönnyíteni a linuxozást.



➲ http://www.linux-mandrake.com/
➲ http://www.mandrakeforum.com/
➲ http://www.mandrakeuser.org/

Matrox

Ha Matrox kártyánkhoz keresünk meghajtót – amit hivatalosan sem a Matrox, sem pedig az operációs rendszerünk fejlesztői nem támogatnak –, akkor itt érdemes körülnézni. Ugyanitt találhatunk BeOS és OS/2 meghajtókat is.

➲ http://www.matroxusers.com
➲ http://www.matrox.com

Compaq-adomány a magyar felsőoktatásnak

A Compaq Magyarország egy-egy Compaq gyártmányú szuperszámítógépet adományozott az Eötvös Lóránd Tudományegyetemnek és a Budapesti Műszaki és Gazdaságtudományi Egyetemnek. Mindemellett további húsz, 64 bites Alpha processzorra épülő Compaq kiszolgálóra pályázhatnak a hazai felsőoktatási intézmények.

A Compaq a világ egyik vezető cége a szupergépek kutatása és fejlesztése területén.

Az elmúlt évben kiépített nyolc szuperszámítógép közül hétre a Compaq kapott megbízást. A közelmúltban pedig a világ legnagyobb szupergépének kiépítésével bízta meg az amerikai energiaipari minisztérium, ezenkívül az emberi géntérkép összeállítását is a cég szuperkiszolgálói segítették elő.

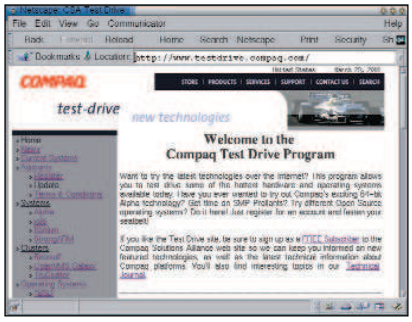
Néhány érdekesebb alkalmazási hely, ahol szintén ilyen számítógépeket használnak:

- A francia atomenergia-tanács. A számítógép segítségével másodpercenként öttrillió művelet elvégzésére nyílik lehetőség, ez a számítási teljesítmény lehetővé teszi az atomkísérletek számítógépes szimulációját.
- Compaq Alpha szupergép képezi az alapját annak a Kanadai Egyetemközi Multimédiás Rendszernek, mely Calgary, Alberta, Lethbridge és Manitoba egyetemeit köti össze.
- Az Ausztráliai Szakmai Számítástechnikai Társulás szinten a Compaqot bízta meg, hogy megépítse Ausztrália legnagyobb szuperkiszolgálóját. Ennek segítségével megfelelő számítási teljesítményt tudnak nyújtani a különböző iparágak kutatói számára.

➲ http://www.compaq.hu
➲ http://www.compaq.com

Rendszerpróba vér nélkül?

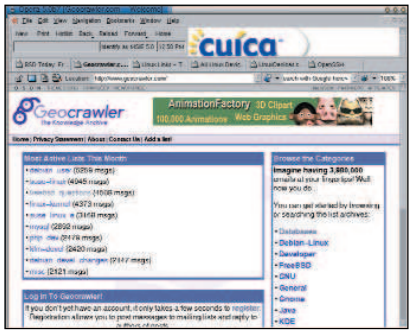
A Compaq Testdrive lehetőséget teremt operációs rendszerek vagy akár programok kipróbálására, ezzel jócskán megkönnyíti a döntést, hogy mit válasszunk. Az operációs rendszerek közül kipróbálhatjuk a Linuxot, a FreeBSD-t, a NetBSD-t, az OpenVMS-t, a Tru64 Unixot és a Microsoft Windows 2000-et.



A nagyobb programok közül pedig az Alpha2Arm, az Applixware Office, az Informix DS2000, a Citrix MetaFrame, az Oracle 8i, az Oracle9iAS Portal, a PAWZ Demo és a Tarantella is választható. Ha valakinek mindez nem lenne elég, akkor akár fürtökkel (Beowulf, OpenVMS és Galaxy TruCluster) is kísérletezhet, mindenféle befektetés nélkül.
A kintről jövő kapcsolatot a gépek előtt felállított tűzfalak korlátozzák. A Compaq Testdrive használatához feliratkozás szükséges.
➲ http://www.testdrive.compaq.com/

Geocrawler

Hatalmas, témakörönként rendezett és kereshető tudásbázissal rendelkező weboldal. Közel négymillió levelet tartalmaz, így mindenki találhat segítséget az éppen megoldásra váró feladataihoz. Az elmúlét hetekben a legnépszerűbb tíz tárgykör: a Debian-felhasználók, a SuSE Linux, a freeBSD



kérdések, a Linux rendszermag, a SuSE Linux-e, a MySQL, a PHP-dev, a kfm-devel, a debian-devel-changes és a misc. Angol nyelvet értő kezdőknek és haladóknak egyaránt ajánlható.
➲ http://www.geocrawler.com

PostgreSQL 7.1 RC 1 kiadás

Az RC-kiadások mindig a próbaváltozatot követik, és az utolsó változatok a végleges kiadás előtt.
Javítások ebben a változatban:

- az outer join (külső összekapcsolás) teljes támogatása
- végre megszűnt a soroknál a nyolc kilobájtos hosszúsági határ
- előírással dolgozó naplózás teljesítményének javítása
- jobb támogatás az összetett lekérdezésekhez
- több összetett lekérdezés támogatása
- javítottak a sebességén
- a JDBC és az ODBC felületen
- a biztonsági mentés és visszatöltés lehetőségein
- teljesebb lett az ANSI SQL92 támogatás.

A 7.1-es több mint 120 fejlesztő kódját tartalmazza.
A fájlok letölthetők az
➲ ftp://ftp.postgresql.org/pub/dev/ címről.
A leírás a következő oldalon érhető el:
➲ http://www.postgresql.org/devel-corner/docs/postgres/

Progeny 1.0

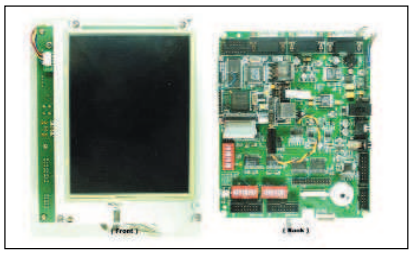
Kiadták a Progeny Linux 1.0-s változatát. Debian-alapokon nyugszik, és sok kiegészítést tartalmaz. Grafikus telepítőjének köszönhetően könnyedén üzembe állíthatjuk. A telepítő felismeri a gépben lévő alkatrészeket és felkészíti azok használatára a rendszert. A programok közül a legújabb megbízható terjesztések kerültek ebbe a Linux-változatba, ilyenek például az XFree86 4.0.2, a glibc2.2, és az USB támogatás. *Ian Murdock* szerint ezzel a lépéssel a Debian sokkal nagyobb eséllyel indul a vállalati felhasználói területen, mint eddig.
A dobozos változat április 23-tól vásárolható meg, egy hónapos telefonos telepítési támogatás és három hónapos Progeny Service Network támogatás jár hozzá, ez magában foglalja a biztonságos programfrissítést és a levélben nyújtott tanácsadást is. Lehetőség van arra, hogy a vásárló kiválassza a termékhez járó támogatás fajtáját, megfelelő ellenértékért.
➲ www.progeny.com

NetBSD

A NetBSD 1.5.1-es foltkiadás (patch release) bétaállapotú, így hamarosan megjelenik a végleges kiadás. Sokat változtattak a megbízhatóságán és számos hibajavítás is belekerült. Jelenleg Alpha és i386-os számítógépekre érhető el ftp-n keresztül, de hamarosan elérhető lesz a többi processzorra is. A tervek szerint a végleges 1.5.1-es változat május elején jelenik meg.

Tynux, az első Linux-alapú okos telefon

A Conversay cég piacvezető szerepet játszik a beszédfelismerésben a mobil és a hagyományos internetes eszközök területén. Együttműködési megállapodást kötött a Seoul-based PalmPalm Technology nevű



vezető programgyártó céggel. Elsőként mobileszközökre hangolták a Linuxot, hogy okos telefonokban, internetes és vezeték nélküli eszközökben használják.
A Conversay a beszédfelismerő programját adja a „Tynux" beágyazott linuxos rendszerhez. Az eszköz ötvözi a mobiltelefonok és a PDA-k tulajdonságait. Különféle multimédás kiegészítéseket is tartalmaz, mint például MP3-lejátszást, illetve animációk, képek és filmek lejátszását.

Új IBM-megoldások

Az e-üzlet manapság szinte minden vállalkozást érint. Legvonzóbb tulajdonságai a hatékonyság és a takarékosság, valamint a közvetlen kapcsolattartás lehetősége a vásárlókkal, legyen szó akár cégekről, akár magánszemélyekről. Ehhez kínál az IBM teljes körű megoldást a közép- és kisvállalkozások számára az iSeries 400 termékcsaládjával. A cég megoldásainak legfőbb haszna, hogy mind a vasat, mind a programokat egy helyről kapjuk, így valóban minőségi szolgáltatásban részesülünk.

Az internethasználók tábora napról napra növekszik, és mára már szinte mindenfajta mobileszközről elérhetjük szolgáltatásait. Ennek köszönhetően a leendő vásárlóerő is hatalmas ütemben fejlődik. Az IBM nemrég termékbemutatót tartott. A rendezvényen láthattuk az üzletek közötti kapcsolat (B2B) működését a Long Hair Production (LHP) előadásában. Szemléltették, hogyan lehet mobiltelefonról autóalkatrészt rendelni, visszaigazolást kérni és utánanézni, hogy ki és mikor szállítja az árut. Az IBM megoldásainak köszönhetően ez a szolgáltatás hamarosan bárki számára elérhetővé válik a megfelelő eszközök birtokában.

Az elektronikus dokumentumvezérlésről elhangzott előadáson bemutatták az IBM Content Manager OnDemand családot. Ennek segítségével könnyedén tárolhatjuk és rendszerezhetjük elektronikus dokumentumainkat, legyen szó akár képről, filmről, hanganyagról vagy hagyományos szöveges dokumentumról. A dokumentumok felvitele önműködő rendszeren keresztül történik. Ezenkívül faxrendszer- és lapolvasó-támogatásának köszönhetően papíralapú irataink tárolását is könnyedén megoldhatjuk.

➲ www.ibm.hu

Vékony ügyfélnek: új magyar mini PC

Ma már minden olyan munkahelyen, ahol egynél több számítógépet használnak, valamilyen hálózatba kötik a gépeket. A hálózatok egy vagy több központi számítógépből (kiszolgáló) és a hálózatra csatlakoztatott személyi számítógépből (munkaállomás) állnak. A gépek szinte kivétel nélkül hagyományos PC-k, szürkésfehér dobozban, merevlemezzel, hajlékonylemezes meghajtóval és CD-olvasó egységgel. A hálózati munkahelyen használt hagyományos PC-k nagyon, sok helyet foglalnak, csúnyák és sokszor zajosak. A Portocom cég az egynél több számítógépet használó irodák kedvében kíván járni, és azok számára, akik a rendelkezésre álló helyet a lehető leghatékonyabban kívánják kihasználni, piacra dobott egy saját fejlesztésű, vékony ügyfélként használható gépet, a mini PC-t.

A mini PC-t nagy sorozatban gyártott szabványos alkatrészekből építették fel, ez teszi lehetővé a kedvező árfekvést, miközben a nagy számítógépgyártók hasonló gyártmányaival megegyező teljesítményt nyújt. A vékony ügyfél minden olyan hálózatba jól beilleszthető, amely kiszolgálóként Windows NT 4.0 Terminal Servert, Windows 2000 Servert, vagy valamilyen unixos kiszolgálót tartalmaz. Egy mozgó alkatrész nélküli, merevlemezként működő adattároló eszköz behelyezésével pedig bármely hálózatba beilleszthető vagy akár különálló munkaállomásként is használható.

A mini PC méretei: 26x26x6,5 cm. A halk processzorventilátoron kívül másik ventilátor nincs a gépben, ezért a mini PC zaja elenyésző. Hagyományos hajlékonylemez- és CD-olvasó egyáltalán nincs a kiépítésben, és a merevlemez is csak választható kiegészítő. A mini PC-kbe ezzel szemben szabványos alaplap került, ezért rugalmasan beállítható rendszer jön létre, melynek másik nagy előnye a kedvező ára. Az alaplapok tartalmazzák a hang- és videovezérlőt is, ezért a termék egyaránt alkalmas a hétköznapi irodai, valamint az egyszerűbb tervezési feladatok elvégzésére.

A mini PC-ket a vevők igényeinek megfelelően egyedileg állítják össze. Processzort Celeron 366-tól Pentium III 800-ig, memóriát pedig 32 megabájttól 512 megabájtig lehet választani. A mini PC lételeme a hálózat, ezért egy alacsony kiépítésű, RTL8139-es vezérlőt tartalmazó 10/100 megabites hálózati kártya található benne.

A hálózati kártyába boot-EPROM helyezhető, amellyel a mini PC merevlemez nélküli hálózati munkaállomásként használható. Későbbi fejlesztési célokra, vagy nagyobb sorozatú egyedi megrendelések kielégítésére még egy szabad PCI csatlakozó áll rendelkezésre.

➲ http://www.portocom.hu

Linuxon futtatott népszerű játékoldal

A RedHat Linux/Apache-ra épülő NeoPets.com honlap néhány hónapos működés után már nyereséget könyvelhetett el, és havi több milliárd találattal büszkélkedhet. A húszéves és ennél fiatalabb korosztályt megcélzó honlapon létrehozhatjuk saját, gondozást és szeretetet igénylő kisállatunkat. Ezenkívül találunk itt állandóan változó világokat is, játékokkal, történetekkel, pályázatokkal és szórakozással. A PC Data Online legfrissebb adatai szerint a NeoPet havonta 2,1 milliárd lehívást és 2,3 milliárd felhasználót vonz, akik összesen 7 óra 48 percet tartózkodnak a honlapon, mely így az Internet legnépszerűbb helyévé vált.

A tavaly augusztusi adatok alapján a NeoPets még az Excite, a Lycos vagy az Amazon oldalainál is látogatottabb. De ami még ennél is fontosabb, hogy a látogatók szívesen maradnak. Például egy átlagos AOL-felhasználó havonta 35 percig bámulja a szolgáltató oldalait, a Yahoo-t használók böngészőjének címsorában pedig egy hónapban 3 óra 22 percig szerepel a yahoo.com.

A NeoPets 7 óra 48 percével messze maga mögött hagyja vetélytársait, sőt, még a Disneyt is tízszer leköröz a laplekérések tekintetében.

A NeoPets egy kollégiumi szobából indult, a készítők az indítási hadjárat keretében küldtek néhány levelet más hasonló honlapoknak. Az első nap kétszáz látogató érkezett, de ez a szám nemsokára elérte a napi kétszázezret. Ekkor létrejött az irányító testület, mely a NeoPets bővítéséhez szükséges üzleti alapok kidolgozását kapta feladatul. A gyarapodó csapat a Pixelgate-hez (kaliforniai internet- és tárhelyszolgáltató) költöztette a kiszolgálót. „Néhány napig nem lehetett elérni bennünket, de a visszatérés utáni első három napban már hatszázezren töltötték le az oldalt" – mondta *Doug Dohring*, a NeoPets elnök-vezérigazgatója. A vállalat először két Linux/Apache kiszolgálót használt, ezt később ötre bővítették. Egy PIII/600-as tárolja a képeket és kétprocesszoros, 0,5-1 GB memóriával rendelkező PIII/600-as gépek működnek webkiszolgálóként.

A látogatók számának további növekedése következtében a cég MySQL adatbázis-kezelő rendszere képességeinek határára érkezett. Ekkorra már általánossá vált, hogy naponta tízmillióan nézték meg az oldalt. Ismét szükségessé vált tehát némi átalakítás.

A vállalat a kaliforniai Santa Clarában lévő Web Zone Inc. és a coloradói Broomfieldben működő Level 3 nevű cégekkel állapodott meg, így lehetségessé vált a sávszélesség bővítése. A NeoPets ekkor újabb alkalmazottakat vett fel és ötven RedHat/Apache, illetve két további MySQL kiszolgálót vásárolt. Ezenkívül vettek egy Sun kiszolgálót is, mely az Oracle adatbázis futtatását végzi. Az Oracle-re történő áttérés után a látogatók napi száma a negyvenmilliót is elérte.

A cégnél jelenleg használt felépítés RedHat 6.2-es és Apache kiszolgálókból, valamint Oracle adatbázisokból áll, de a mai napig használnak MySQL-t is a sokféle feladat megoldására.

Az Oracle bevezetése ellenére a NeoPets továbbra is a Web egyik legnagyobb Apache-felhasználója maradt. Bár a hatalmas forgalom kiszolgálásához mindenképpen szükség volt az Oracle-re, a NeoPets vezetői szerint a nyílt forrású fejlesztés jobb minőséget, magasabb fokú megbízhatóságot jelent, és a cég továbbra is gondosan figyeli a PHP, az Apache és a MySQL fejlődését, hátha egyszer e három rendszer együttese képes lesz végleg kiváltani az Oracle-t.

„Olyan embereket keresünk, akik e nyílt forrású fejlesztéseket képesek saját munkájukkal új síkra helyezni – nyilatkozta *Bill McCaffrey*, a cég szakmai igazgatója. – Én bízom abban, hogy ha a megfelelő embereket sikerül elérnünk, akkor ezen alkalmazásokat olyan tökéletesre fejleszthetjük, hogy a legforgalmasabb honlapoknál is megállják a helyüket."

A NeoPets elhatározta, hogy webfejlesztőket, nyílt forrású programok fejlesztéséhez programozókat, rendszergazdákat és tanácsadókat vesz föl a nyárra várható újabb hatalmas forgalomnövekedés kiszolgálására.

Drew Robb

Régi vicc új köntösben

Egy ember vidéken autózik, megáll egy bárányokkal teli legelő mellett, kiszáll, odalép a pásztorhoz, és azt mondja neki:

– Szép jó napot, pásztor uram! Van egy ajánlatom a maga számára: megpróbálom kitalálni, hogy hány báránya van. Ha sikerül, elviszek egyet, ha nem, akkor magáé a Porschém.

Gondolkozik a pásztor, ez úgyse sikerülhet az idegennek, szóval tuti nyerésre áll.

– Jól van – szól a pásztor.

– 137 – mondja az idegen.

– Rúgja meg a ló, igaza van! – szól a pásztor, és becsületesen átnyújtja a bárányt. Az idegen mosolyogva begyömöszöli a csomagtartóba, beszáll, és éppen indulna, amikor a pásztor bekopog az ablakán.

– Nekem is van egy ajánlatom a maga számára. – mondja a pásztor – Ha kitalálom, hogy mi a maga foglalkozása, akkor enyém az autó. Ha tévedek, akkor viheti az összes bárányt.

– Áll az alku. Na, ki vele.

– Maga tanácsadó.

– Hihetetlen! Honnan a csudából találta ki?

– Nem volt nehéz. Hívatlanul jött, elmondta azt, amit már tudok és még fizetnem is kellett érte!

➲ http://www.sapconnection.com/cons_humor.html

Nyílt (video)kártyákkal

A gépgyártó cégek komoly pénzeket költenek a nem módosítható bináris programokra. Pedig a cégeknek elegendő lenne műszaki leírásokat közreadni, mert ezek alapján a nyílt fejlesztés hívei önként elkészítik a meghajtóprogramokat, és ehhez ingyenes támogatást is nyújtanak. *Bakonyi Ferenccel* beszélgettünk, aki maga is részt vesz e munkában. Az általa írt linuxos meghajtóprogramokról, ezek fórumáról, valamint a nyílt fejlesztésekről kérdeztük.

Mi szükséges ahhoz, hogy valaki részt vehessen a linuxos fejlesztésekben?

B. F.: A legkevesebb a felhasználói szintű Linux-ismeret, valamint tudni kell C nyelven programozni. Ezenkívül kellő elszántság és belső késztetés sem árt, hogy olyan programot hozzunk létre, ami saját magunknak megfelel. Az külön öröm, ha munkánk mások hasznára is válik.

Hogyan kezdted el a fejlesztői munkát?

B. F.: A Linux egy jól kidolgozott rendszer, ami folyamatosan fejlődik. Nem tökéletes, mert nem tud mindent – hiányosság, hiba akad benne bőven. Én belefutottam egy ilyen hiányosságba, és úgy döntöttem, magam oldom meg ezt a nehézséget.

Mi történt?

B. F.: Az otthoni linuxos gépem nem ismerte fel a hangkártyámat, ami új termék volt és meghajtóprogramom sem volt hozzá. Némi utánajárás után kiderült, hogy ez a kártya nem sokban különbözik egy korábbi típustól, és a régi meghajtóprogramban pár programsor megváltoztatása elég volt ahhoz, hogy működjön. A módosítást elküldtem a program készítőjének, aki jónak találta, és figyelembe vette a változtatásaimat.

Hová küldted a javaslatodat?

B. F.: A nyílt fejlesztések fő jellemzője, hogy a programforrás és a leírás teljesen nyitott, bárki szabadon belenézhet, átírhatja igénye szerint. A dolog önszerveződő: a változtatás után a programot meg is tarthatjuk magunknak, de közzé is tehetjük. A fejlesztő a program forrásában általában megadja az elektronikus levélcímét vagy a honlapot, ahol megtalálható a munkája. A legtöbb projekthez saját levelezőlista is tartozik, ahová a javaslatokat, kérdéseket el lehet küldeni. A hasonló érdeklődésű emberek így egyből meghányják-vetik a változtatásokat, a kérdéseket és természetesen a gondok jellegétől függően hamar válaszolnak.

A nyílt forrással nem lehet visszaélni?

B. F.: Az egész világ láthatja a munkádat: ha hiba vagy hátsó szándék lenne benne, gyorsan észrevennék. A nyílt fejlesztésekkel szemben a zártak programhibáit, szándékos hátsó ajtóit nagyon nehéz felderíteni, mert nem láthatsz bele a program forrásába, hiszen ezt tiltja a felhasználási szerződés is. A nyílt rendszerben azonban hamar észre lehet venni a biztonsági rést, amin keresztül feltörhető a gép. Ugyanis minden program *tartalmaz* hibákat, amelyeket rosszindulatúan kihasználva akár betörhetnek gépekre, és adatokat is eltulajdoníthatnak. A nyílt rendszerben rövidre zárhatók az ilyen esetek, mert mindenki „belülről" láthatja a működő programot, és ugyanolyan gyorsan be is foltozhatja a lyukakat. Például a Linux rendszermag 2.2-es változatában, szinte rögtön a kibocsátás után találtak egy súlyos hibát: egy rossz szándékú parancstól lefagyhatott a gép. A megoldás napokon belül megjelent, és így helyrehozták a tévedést. Zárt rendszereknél egy ilyen eset hónapokig, sőt, évekig is elhúzódhat!

Melyek a saját fejlesztéseid?

B. F.: A hangkártya sikerén felbuzdulva átnéztem a gépemet, és láttam, hogy számos alkatrésznek nincs teljes támogatása, azaz nincs kihasználva az összes képessége. Ha lehetne jobban is, miért ne hozzuk ki a legtöbbet? A divatos termékekhez sokan fejlesztenek, de van azért még kiaknázatlan terület. Például a vadonatúj vagy a vitatható eszközök, melyekhez nincs megfelelő leírás. 1999-ben támadt az az ötletem, hogy otthonra kétmonitoros rendszert építek ki, hogy mindkét monitorral egyszerre használhassam a gépemet. A következő módszerrel éltem: a Riva 128-as VGA kártyát és a régi Hercules kártyámat kombináltam és próbáltam kétféleképpen használni. Egyrészt csak szövegesen: de így nem lehetett a terminált visszafelé görgetni a Hercules monitoron, hiába volt hozzá linuxos meghajtóprogram, nehézkes volt vele dolgozni. Másfelől X-Window alatt kíséreltem meg egyszerre használni a két monitort, és kiderült, hogy ezt csak úgy lehet, ha a Riva-kártya mindössze 16 színű üzemmódban működik, alkatrészszintű gyorsítás nélkül. Hamar rátaláltam egy létező projektre, a framebuffer alrendszerre, ami a nyílt fejlesztésű Linux-mag része volt. A framebuffer lényege, hogy grafikus képernyőn jeleníti meg a szöveges konzolt, ezért más-más felbontásban használhatjuk a virtuális terminálokat, X-et is futtathatunk rajta. Ráadásul rendszerindításkor a jól ismert pingvin logó fogadja a felhasználót. A különböző videokártyákhoz különböző fb meghajtóprogramok vannak. Ezek kombinálhatók, és így több kártyát is lehet egyszerre használni. Balszerencsémre sem a Herculeshez, sem a Riva 128-hoz nem létezett még ilyen program, így hát elhatároztam, hogy magam írom meg ezeket. Először a Hercules kártya meghajtóprogramjával, a hgafb-vel kezdtem, mert ez egy régi, jól leírt modell, egyszerű programozással. És amíg a Herculeshez írom a programot, gondoltam, addig megismerkedem a Linux-mag különböző alrendszereivel, és majd kellő tapasztalattal készíthetem el a Riva kártya meghajtóprogramját.

Nehéz munka volt?

B. F.: C-ben kellett programoznom, ami azért nem volt könnyű, mert évekig nem dolgoztam benne, pedig előtte „profi" voltam. Pár hónapig ügyködtem a hgafb-n, főként hétvégéken. Természetesen más programozók forrásait is felhasználtam, mert ugye szabadon átvehettem programrészleteket – hála a nyílt fejlesztéseknek. Amikor már működött a meghajtóprogram, létrehoztam egy honlapot, ahol közzétettem a leírását, a honlap elérhetőségét pedig elküldtem levelezőlistákra .

Mekkora volt az érdeklődés iránta?
B. F.: Napokon belül több száz találat érte az oldalt és záporoztak a visszajelzések, javaslatok. Programrészleteket is kaptam, miként írjam át, hogy mindenkinél jól működjön. A nyílt közösség lökést adott a fejlesztésnek, ellenőrzések és hibajavítások formájában. Egy cseh programozó olyan új lehetőséget is felfedezett, amire nem is gondoltam az elején; saját karakterkészleteket is lehetne használni a monitoron, például egyedi karaktereket: a magyar, cseh stb. ábécé betűkészletéből. A fő célom a két monitor együttes használata volt, az embereket azonban legjobban a kutatásom „mellékterméke", a különleges karakterhasználat fogta meg. A cseh fiúnak annyira tetszett, hogy egyenesen *Linus Torvaldsnak* küldte el a meghajtóprogramot, aki azon nyomban beleillesztette a fejlesztői rendszermagba. Ezzel a lehető legszélesebb nyilvánosságot érte el hgafb meghajtóprogramom. Pár nappal a felvétel után már jöttek is a további hozzászólások.
Sok javításra szorult a program?
B. F.: Pár apró javítás kellett, de a mai napig él és működik. Sikerként könyvelhettem el. A figyelmem ezután a Riva 128-as kártya felé fordult, és örömmel láttam, hogy már más is dolgozik rajta. Igaz, a rivafb-vel megelőztek, de az a program eléggé kezdetleges volt, az én kártyámon például egyáltalán nem működött. Nekifogtam, belejavítgattam és a módosított változatot ismét a framebuffer fejlesztői közösségnek és a rivafb szerzőinek továbbítottam, akik örömmel vették a közreműködésemet és beillesztették a programba a javítást. Azóta ez a program sok változáson ment keresztül, még többen segítettek a fejlesztésében, természetesen én is folyamatosan küldtem a javaslataimat.
Jelenleg mi foglalkoztat a Linuxszal kapcsolatban?
BF: A Riva kártyát szeretném képessé tenni a beépített videó ki- és bemenet használatára. A feladat azért nehéz, mert a kártyát gyártó cég a műszaki leírásokat szigorúan titkos anyagként kezeli és nem ad ki adatokat. Szép lassan azért halad ez is, a tv-kimenet nemsokára működni fog, de akit érdekel, bekapcsolódhat a fejlesztésbe, lendíthet az ügyön!

Kapcsolódó címek

A framebuffer alrendszer honlapja
➲ http://www.linux-fbdev.org/

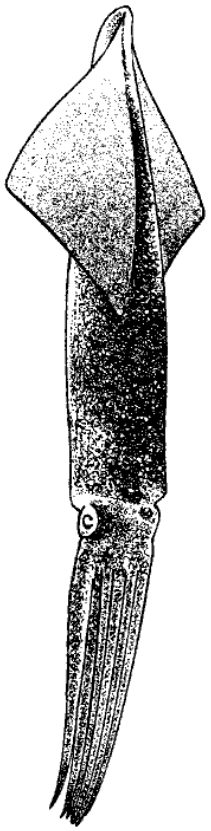
Hercules framebuffer meghajtó
➲ http://drama.obuda.kando.hu/~fero/cgi-bin/hgafb.php

Riva tévéprojekt
➲ http://fero.koli.kando.hu/rivatv/
➲ http://sourceforge.net/projects/rivatv/

Az alkalmazott Calimariért járó Linux Journal-díj nyertese...

A nyílt forrású fejlesztés nagyszerű üzleti modell, de *Eric Raymond* példája kivételével a hirdetések területén eléggé gázos.
Na jó, nézzük meg kicsit közelebbről a dolgot. Sok vállalat büszke nyílt forrású fejlesztéseire, de szerkesztőként elmondhatom, hogy vajmi kevés azon nyílt forrású fejlesztések számra, ahol igazi PR-részleg dolgozik a háttérben. Mire is kellenének? Például arra, hogy a hozzám hasonló szerkesztőknek elmeséljék, éppen mivel foglalkoznak. Természetesen, ha elfelejtjük a nyilvánvaló dolgokat megemlíteni, akkor azonnal lehordanak a sárga földig. Például amikor tavaly tévesen jelentettük be, hogy a Borland-féle InterBase lesz az első nyílt forrású adatbázis-kezelő rendszer, a PostgreSQL-t készítő csapat dühös levelekkel bombázott. (Bár ha jobban belegondolok, nem is jött olyan sok levél.)
Vegyük például a proxy gyorstárazást, ez rendkívül hasznos nagy forgalom esetén, de a megszállottak kivételével keveseket hoz lázba. Az egyik megszállott mondta nekem, hogy a Squid ➲ http://www.squid-cache.org/ a nyílt forrású proxykiszolgálók királya.
Nos, nekem úgy tűnik, hogy egy nagy marék (főként fizetős, zárt forrású) proxykiszolgáló létezik, melyeket a Lucent, a Novell, az IBM, a Cisco, a Microsoft és a többi nagy hal forgalmaz. Áruk nullától hat számjegyűig terjed, a Squid e tekintetben a sor legelején foglal helyet. A GYK szerint a Squidet rengeteg ftp-oldalról le lehet tölteni.
A termék versenyképes – szó szerint. Az IRCache nevű csapat, a National Science Foundation és mások által kifejlesztett Polygraph ➲ http://www.polygraph.ircache.net/ segítségével rendszeres felméréseket végez. Az IRCache honlapján is olvasható eredmények rengeteg oldalból állnak. Ezekből kitűnik, hogy a Squid sokszor vezet, itt-ott lemarad, de mindig az élmezőnyben foglal helyet.
A legegyértelműbb eredmény *Matthew P. Barnsonnak* a Slashdoton olvasható 5. szintű hozzászólásából derül ki: „Elmondhatom, hogy sok tapasztalatot szereztem a Novell ICS, a Network Appliance NetCache és a (Solarison futó) Squid rendszerekkel és mindegyikük tökéletesen működött. A Squid 2.3-stable1 változat fordítása, telepítése és beállítása gyerekjáték volt."
Később amikor találkoztunk kijelentette: a Squid a Harvest projektből kialakult, figyelemre méltó és ingyenes proxygyorstár, mely új mércét állított a gyorstárak területén. Ezért az árért a Squid igencsak frankó.
Egy másik nyílt forrású fejlesztésről is szólt Matt: „az Apache webkiszolgálót külön nem említi a felmérés, pedig tapasztalataim szerint e rendszer kitűnő választás a gyorstárazás mellé, hiszen ugyanaz a kiszolgáló, amit akár a kollégiumi szobánkból is üzemeltethetünk, jelentősen meggyorsíthatja a hálón való böngészést."
Matt biztosan tudja, hogy miről beszél, hiszen a világ egyik leglátogatottabb honlapjánál (iMALL.com) dolgozik. Igyunk tehát a Squid csapat egészségére, s egyúttal hadd üzenjek a többi nyílt fejlesztésen dolgozónak: ti is legalább ilyen részletesen számoljatok be arról, hogy éppen mivel foglalkoztok. Köszi!

Doc Searls



A Microwindows múltja, jelene és jövője

Eleinte csak szórakozásból terveztem, fejlesztettem a Microwindowst, mert imádom, amikor kis rendszerek nagy dolgokra képesek.

Greg Haerr a Century Software vezérigazgatója és a Microwindows Project alapítója. A projekt beágyazott rendszereken és eszközökön futó grafikus ablakkezelő rendszer fejlesztésével foglalkozik.

Rick: *Mikor találkoztál először a Linuxszal, és mi késztetett arra, hogy grafikus felületet írj a Linuxhoz?*
Greg: A linuxos múltam 1993-ra nyúlik vissza, akkor használtam az egyik legelső i386-os terjesztést, az Yggadrisilt. Emlékszem, hogy a csomag egy hajlékonylemezt és egy CD-t tartalmazott, ami újdonság volt abban az időben. UMSDOS fájlrendszerről indult – ez lehetővé tette, hogy a meglévő DOS-os fájlrendszer szabad területét használva fusson a Linux –, elindította a grafikus felületet, megjelenített egy bejelentkezési képernyőt és lejátszotta a Star Trek zenéjét. A képernyő felbontását három beállítás között lehetett kapcsolgatni az ALT-+ billentyűkombinációval… A Linux már a kezdetek kezdetétől megmutatta, hogy milyen jól tud alkalmazkodni a sokféle gépfelépítéshez, és azt, hogy a felhasználó segítsége (beállítások megváltoztatása) nélkül is meg tudja ezt tenni.
Több mint 25 éven át programoztam, vagyis jóval azelőtt kezdtem a programozást, mint hogy üzletember lettem. Annak ellenére, hogy hét éve követem a Linux fejlődését, csak az utóbbi két évben vettem részt néhány fontosabb fejlesztésben. Mindenképpen megérte.
Rick*: Miből gondolod, hogy a Linux jó választás a beágyazott rendszerekhez?*
Greg: Ennek több oka is van. Ezek alátámasztják azt a feltevést, hogy a Linux roppant gyorsan fog fejlődni a jövő évtizedben. Először is, a 32 bites processzorok ára végre elég alacsony, sebességük és áramfogyasztásuk is megfelelő ahhoz, hogy olcsó beágyazott rendszerekhez használhassuk őket, például tenyérgépekben, webpadokban és egyéb érintőképernyős eszközökben. A Linux szinte a kezdetek kezdetétől több processzortípust támogatott, például a fejlett RISC processzorokat is. Így a Linux és a beágyazott rendszerek közös útja szinte természetesnek vehető. Másodsorban a programfejlesztők szeretnék átültetni az asztali gépeken már nagy sikert aratott felületet hordozhatóbb eszközökre. A vezeték nélküli hálózatos eszközökben sem kell újra feltalálni a spanyolviaszt, amikor olyan sok közkedvelt alkalmazás is felhasználható már.
Végül a Linux, mivel ingyenes és nyílt forrású, lehetővé teszi, hogy a gyártók tovább mérsékeljék a költségeket, a legjobb megvalósítást lehetővé tevő fejlesztések és eljárások megosztásával, használatával. Ez egyaránt nyereség a gyártó és a fogyasztó számára.
Rick*: Hogyan viszonyul a Linux a beágyazott rendszerekben használatos „hagyományos" operációs rendszerekhez? Olyanokra gondolok, mint a VxWorks, a pSOS, vagy a VRTX.*
Greg: A Linux legnagyobb előnye a fejlesztési szakaszban mutatkozik meg, ugyanis a fejlesztők ugyanazt az operációs rendszert használják az asztali gépükön a fejlesztés során, mint ami azon az eszközön van, amire az alkalmazást szánják. Ez azt jelenti, hogy annyira tisztában vannak a céleszköz lehetőségeivel, mint amennyire az asztali gépükével. Például a Century Software új, a Compaq iPAQ-hoz készített Microwindows operációs környezet és fejlesztői eszköz esetében teljes egészében elő tudtuk állítani a célkörnyezet pontos mását, ezért a program fejlesztése az eszköz fejlesztésével egy időben folyhatott. Vannak természetesen más előnyök is, főleg azok számára, akik már befektettek a unixos megoldásokba, legyen Sun, DEC, HP vagy IBM a választás. Ha ugyanis fejlesztettél valamilyen alkalmazást unixos környezethez, és ki akarod használni az új, hordozható eszközök nyújtotta lehetőségeket, kimondhatatlanul könnyebb helyzetben vagy, ha az alkalmazásodat Linuxra kell átültetni, mintha ugyanezt Windows CE-re kellene megtenni. És természetesen, mivel a webkiszolgálók többsége Linuxot futtat, az ügyféloldalon is megjelennek a linuxos alkalmazások.
Rick*: Miért fogtál bele a Microwindows fejlesztésébe?*
Greg: Eleinte csak szórakozásból terveztem, fejlesztettem a Microwindowst, mert imádom, amikor kis rendszerek nagy dolgokra képesek. Az egész azonban gyorsan felfutott, a fejlesztést a felhasználói igényekhez kellett igazítani, számos olyan lehetőséget is beépítettem, melyeket addig csak a fejlettebb rendszereken lehetett elérni.
A Microwindows támogatja például az élsimított szöveget, a TrueType és Adobe Type 1 betűtípusokat és a külön csatornára alapozott átlátszóságot (alpha blending), ezeket például az X is csak most kezdte támogatni.
Ha a Microwindows méretét összehasonlítjuk az X és a hozzá tartozó programkönyvtárak méretével, a különbség óriási a Microwindows javára. Az X fejlesztői keményen dolgoznak azon, hogy a rendszer méretét csökkentsék. A két rendszer összetettsége között azonban mindig óriási lesz a különbség. A beágyazott rendszerek esetében mindig igyekszünk a lehető legjobban kihasználni a gépet, mivel elsősorban ezért hozzuk létre a készüléket. Ez azt jelenti, hogy amikor a grafika gyorsításához ki akarod használni a gép lehetőségeit, vagy amikor a programnak le kell kezelnie valamilyen furcsa érintőképernyős külső egységet, egy nagyságrenddel könnyebb dolgod van a Microwindows esetében. Emellett a Microwindows Nano-X protokollja nagymértékben hasonlít az X-hez, a különbség a színkezelésben van, ezért a Microwindows alá könnyebb alkalmazást fejleszteni.
Rick*: Milyen más megoldásról tudsz, ami esetleg versenyezhet a Microwindows rendszerrel?*
Greg: Valószínűleg két nagy vetélytársunk van, már ha azokat a cégeket nézzük, amelyek valójában létrehozzák az újításokat ezen a területen. A TrollTech mellett ott van a Javában megvalósított PocketLinux is. A Century, a TrollTech és a Transvirtual tekinthetők vetélytársainknak is, bár merőben másképp próbálják az alkalmazásaikat a piacra dobni, mindegyikük más területre összpontosít.

A Transvirtual PocketLinuxa egy aranyos Javaás megvalósítás, ami fut framebufferen, viszonylag kevés alkalmazás érthető el hozzá, de ezek az alkalmazások minden felhasználási területet lefednek. Ez mind szép, ha van Javád. A PocketLinuxon nem futtatható olyan alkalmazás, amit nem Javában írtak meg, tehát nem alkalmas általános célokra. A TrollTech Qt/Embedded rendszere hasonló helyzetben van, de népesebb felhasználótábor számára jelenthet megoldást. A Qt/Embedded projekt célja, a Qt könyvtárral egyetemben, egy Windowshoz hasonló keretrendszer létrehozása az alkalmazások számára. A Qt-s alkalmazásokat pillanatok alatt át lehet ültetni egy framebufferes beágyazott eszközre. Ez nagyon szép, de a Qt/Embedded alatt csak Qt-s alkalmazások futtathatók. Ez sok kódolást és egy összetett rendszer kezelését jelenti. A Microwindows a leggyorsabb mind közül, és kiválóan alkalmas általános alkalmazásfejlesztéshez, itt nem lehet arra számítani, hogy az alkalmazások kinézete megegyezik, hogy ugyanazt az elemkészletet használják, vagy ugyanazt a Java-megvalósítást. Ez az általános környezet jól jön, ha általános alkalmazásfejlesztésről van szó, ahol sok külső cég által készített terméket kell használni, vagy ha több eltérő eljárást kell alkalmazni együtt, az API vagy a fejlesztéshez felhasználható nyelvek korlátozása nélkül.

Rick: *Útnak indítottál egy böngésző projektet is, a ViewML-t. Milyen más lehetőségek vannak a ViewML mellett, és hogyan viszonyulnak ezek az általad fejlesztett programhoz?*

Greg: Természetesen elfogult vagyok, mert a ViewML a Century Software nyílt forráskódú beágyazott böngészője. A ViewML elfut 800 k ROM-mal és 2,1 MB RAM-mal, és kiválóan működik a kis eszközökkel. Eléggé korlátozottak a képességei, a kevés memória határokat szab. Nemrégen, Ausztráliában egy lelkes csapat átültette a teljes Mozilla böngészőt Microwindowsra, és jól működik. Tehát az igényektől függően elérhető a kis méret és a teljes kiépítésű böngésző is, kinek mire van szüksége. Az Opera nagyon jól vizsgázik HTML-megfelelőségből, és létezik már egy még ki nem adott változat Microwindowsra. A Microwindows munkakörnyezet szempontjából az volt a fontos, hogy a ViewML jól és gyorsan fusson a kis PDA eszközökön és a nagyobb gyorsabbakon egyaránt. Ha azonban nem felel meg az ügyfél számára és van elég memóriamodul, még mindig felteheti a Mozillát.

Rick: *Milyen kihívásokkal találkoztál a ViewML fejlesztése közben? Hogyan léptél túl a nehézségeken? Ki kellett hagynod bizoyos tulajdonságokat a ViewML-ből?*

Greg: A lehető legkisebb méret kulcsfontosságú volt, mint az is, hogy a böngésző minden lapot helyesen tudjon megjeleníteni. A KDE csapat kfm KHTML elemére esett a választás. A ViewML projekt létrehozott egy beágyazott böngészőt, úgy, hogy közben a KHTML kódjában egy sort sem változtattunk meg, így nem nagyon tudtuk elrontani. Írtunk egy QT→FLTK osztályátalakító réteget, ami lehetővé tette, hogy az egész grafikus felületet és az elemen megjelenő vezérlőket kevesebb, mint 100 k-ban valósítsuk meg. A végeredmény a ViewML. A legtöbb munkát az átalakítóréteg véglegesítése jelentette. Azt is mindig szem előtt kellett tartanunk, hogy a böngészőnek elég gyorsnak kell lennie a mindennapi használathoz. Még mindig dolgozunk a TrueType betűtípusok megjelenítésének tökéletesítésén.

A jövőre nézve, azt tervezzük, hogy felhasználjuk a KDE Konquerer v2.0 html programelemet, ami támogatja a HTML 4.0-t és a JavaScript 1.4-et. Ennek a megvalósítását is egy átalakítórétegen keresztül tervezzük.

Rick: *Visszatekintve, milyen volt vezetni egy nyílt forráskódú projektet?*

Greg: A Microwindows Projekt biztosan nem lenne az, ami ma, ha nem használhatott volna fel számos nyílt fejlesztést, mint például a méretezhető betűmegjelenítés támogatását, ha nem lett volna lehetőség a kipróbálásra és a hibakeresésre számtalan rendszeren, és ha nem tudtuk volna megvitatni a kulcsfontosságú kérdéseket a levelezőlistán. Nagyon élveztem, hogy vezethettem ezt a fejlesztést. A Nyílt Forráskód Közössége sokat segített azzal, hogy a megjelent terméket, majd az ahhoz megjelent további alkalmazásokat azonnal kipróbálták. Hidd el, ha valami nem működne jól, arról tudnék…

Rick: *Milyen új bővítéseket vagy fejlesztéseket tervezel a Microwindowsba?*

Greg: Sokkal több, a piacon már jelen lévő vagy megjelenni készülő PDA-ra lesz elérhető a Microwindows ingyenes bináris terjesztése. Ez felkelti majd a fejlesztők érdeklődését, hiszen még több hasznot húzhatnak abból, hogy az alkalmazásuk több felületen futhat.

Nagyon érdekesnek ígérkezik a megjelenésünk a webpadok területén. Egy olyan felépítés áll rendelkezésünkre, amely lehetővé teszi, hogy a fejlesztők és a felhasználók ugyanazokat a grafikus alkalmazásokat használják számos lapos képernyős eszközön. Az alkalmazásnak más lehet a kinézete, attól függően, hogy 240x320-as vagy 600x800-as felbontásban használják, vagy hogy két-, háromdimenziós vagy tévészerű vezérlőket valósítanak meg a fejlesztői könyvtárak.

Rick: *A Linuxszal ellentétben a Microwindowsra nem a GNU General Public License (GPL) vonatkozik. Szólnál pár szót a Microwindows felhasználási feltételeiről?*

Greg: A Microwindows első fejlesztőivel összhangban úgy döntöttünk, hogy a termék felhasználását az MPL fogja szabályozni, ami kevesebb megkötéssel jár, mint a GPL. Ez azt jelenti, hogy a Microwindowst fel lehet használni zárt forráskódú fejlesztésben is, vagy olyan esetben, ha a meghajtó forráskódja nem tehető közzé. Lehet, hogy ez ellenkezik a szigorú értelemben vett szabad program szellemével, de én inkább gyakorlati szempontból közelítem meg a dolgot. Az az igazi küldetésem, hogy elősegítsem a grafikus felületű beágyazott eszközök elterjedését, ezért olyan rendszert kellett fejleszteni, aminek engedékenyebb a felhasználási szerződése.

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

A tanácsadónktól hallott mondatok jelentése

A projekt koordinációs megbeszélése folyik.
Leültünk kávézgatni a srácokkal.

Az előzetes próbák alapján nem következtethettünk ilyen végeredményre.
Kipurcant a nyavalyás, amikor a kapcsolóhoz nyúltunk.

A teszt eredményei kiemelkedő hatásfokról adnak tanúbizonyságot.
Mi is alig hisszük el, de tényleg működik.

Az egész koncepciót el kell vetnünk.
Az egyedüli srác, aki értett hozzá, kilépett.

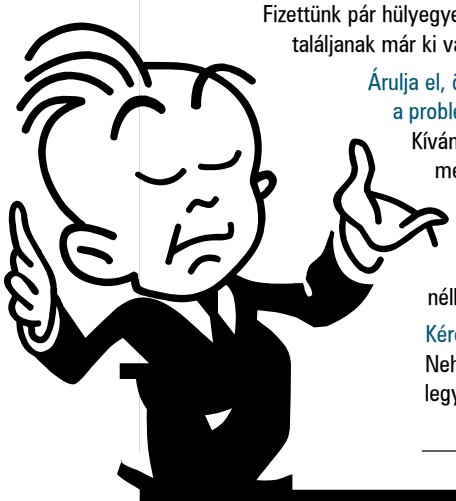
Már folyamatban van.
A helyzet teljesen reménytelen, de azért úgy teszünk, mintha okosak lennénk.

Meg fogjuk vizsgálni a kérdést.
Na ne, már így is elég bajunk van.

Ossza meg velünk bátran a gondolatait.
Amíg olyanokat mond, amik összecsengenek saját nézeteinkkel, addig kíváncsiak vagyunk a véleményére.

Ez egy vadonatúj eljárás/rendszer.
Az előző rendszer egyetlen elemét sem használhatjuk fel újra, tehát mehet a kukába az egész.

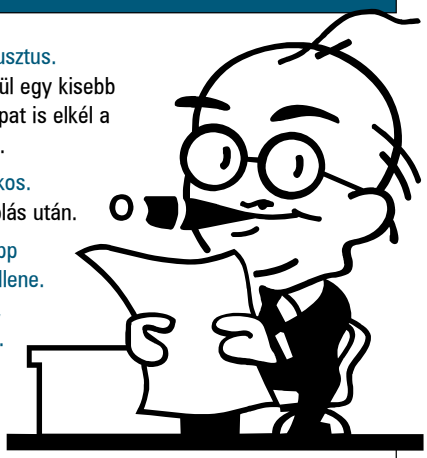
Mindenre kiterjedő vizsgálatot folytatunk, teljesen új megközelítéssel.
Fizettünk pár hülyegyereknek, hogy találjanak már ki valamit.



Árulja el, ön hogyan látja a probléma lényegét?
Kíváncsi vagyok, hogy megint milyen hülyeség jutott az eszébe.

Nagyon masszív.
Ne próbáljuk emelő nélkül mozgatni.

Kérem, írja ezt alá.
Nehogy már csak én legyek felelős érte.



Rendkívül robusztus.
Az emelőn kívül egy kisebb költöztető csapat is elkél a mozgatásához.

Energiatakarékos.
Csak kikapcsolás után.

Ma mindenképp beszélnünk kellene.
Az irodámban, négyszemközt.
Már megint elszúrtam valamit.

Pehelykönnyű.
Egy kisebb emelő már megteszi.

A vásárlók elégedettsége szavatolt.
Már a huszonkilencedik határidőt is annyival túlléptük, hogy a szerencsétlen vásárlóknak jó lesz bármi, csak jöjjön már.

Kevés karbantartást igényel.
Majdnem lehetetlen javítani.

Többévi munka gyümölcseként…
Egyszer csak (mi sem tudjuk miért) működni kezdett.

Fantasztikus technológiai áttörés.
Úgy-ahogy működik, de csak a beavatottak értik, hogy mi a fene ez az egész.

Nincs szükség karbantartásra!
Javítani lehetetlen.

Számos különböző megközelítést ki kell még próbálnunk.
Csak találgatni tudunk.

A szabvány előírásait követjük.
Mindig így csináltuk, és jó volt.

Nem kaptam meg az ön által küldött e-mailt.
Úgy egy hete nem néztem meg a leveleimet.

➲ http://www.sapconnection.com/cons_humor.html

Tíz dolog, amit sosem fogunk egy tanácsadótól hallani

10. Igaza van, ezért tényleg elég sokat kérünk.
9. Fogadjunk, hogy kibírom egy hétig anélkül, hogy olyanokat mondanék, hogy „szinergikus”, meg „értéknövelt”!
8. Mi lenne, ha a projekt sikerességétől függő mértékben kapnék fizetést?
7. Az egész tervet egy könyvben olvastam.
6. Valójában az egyetlen különbség köztünk és az XY cég között az, hogy mi többet kérünk a tanácsadásért.
5. A tudásom nem elégséges ahhoz, hogy erről értelmesen beszéljek.
4. Megvalósítás? Hááááát, én csak a hosszú jelentések írásához értek.
3. Ne nekem köszönje, hanem Lacinak a piackutató részlegről.
2. A gond az, hogy itt túl kevés emberre túl sok munka jut.
1. Nekem úgy tűnik, hogy itt nincs hiba.

➲ http://www.sapconnection.com/cons_humor.html

Érdekes helyek

Részeges Linux
➲ http://i-want-a-website.com/about-linux/oct98.shtml#Drinking-Game5

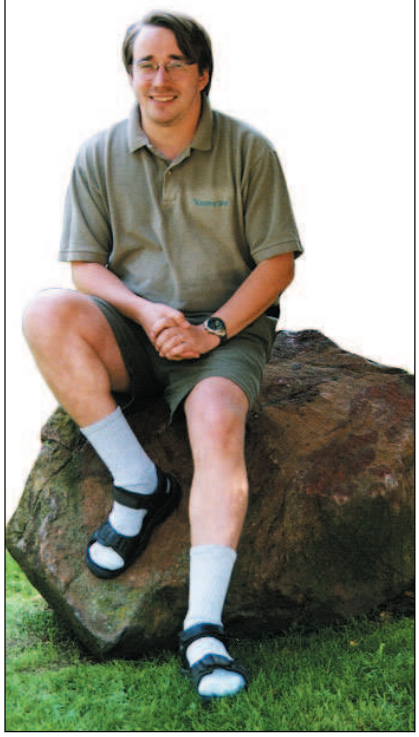
Stiledot
A Slashdot és a Stile Project e furcsa gyermekét a BBspotnak köszönhetjük
➲ http://www.bbspot.com/SiteBlender/index.html

A Linux már megint okosabb...
➲ http://www.bbspot.com/News/2000/12/smart.html

Andrew Leonard ingyenes fejlesztése a Salon.com-nál
➲ http://www.salon.com/tech/fsp/about/index.html

A LJ története

Linus Torvalds (a LJ 1995. márciusi számában): egyszer majd elkészítem a 2.0-s rendszermagot…
Valaki a Novelltől: nem kell csodálkozni azon, hogy a Linux ilyen jól működik, hiszen Linus tízezer felhasználó visszajelzéseire számíthat.
Az immár családos Torvalds úr már jócskán a 2.x-es rendszermagot készíti, háta mögött nyolcmillió felhasználóval.



Linux-index

1. Az IBM z900 (régebben S/390) nagygép áramfogyasztásának költsége naponta (USA): 32 dollár
2. Az IBM z900 alapgép ára (USA): 750 ezer dollár
3. Az 900-on párhuzamosan futtatott Linux példányok száma (USA): 41 ezer
4. Azon cégek százaléka, amelyek alkalmaznak Linuxot, illetve legalább egy linuxos rendszer üzembeállítását tervezik (USA): 68%
5. Tavaly az Egyesült Államokban az eladott informatikai eszközök száma: 7 440 000
6. Az informatikai eszközeladások száma 2005-ben az Egyesült Államokban várhatóan: 51 800 000
7. Az informatikai eszközök eladásának növekedése az Egyesült Államokban: 47,4%
8. Tavaly a világszerte eladott informatikai eszközök száma: 29 millió
9. Az informatikai eszközeladások száma 2005-ben világszerte várhatóan: 305 millió
10. Az informatikai eszközök eladásának éves növekedése világszerte: 59,8%
11. A letöltött, vagy másképpen szállított BSD-alapú Apple OS X béták száma 2001. január 1-jéig: százezer
12. Az OS X fejlesztési folyamatot visszajelzésükkel segítők száma: hetvenezer
13. Naponta a Monster.com-ra elküldött önéletrajzok száma: 38 ezer
14. A Monster.com-on nyilvántartott állások száma: 12 000 000
15. Az egyetlen beteg állat által is megfertőzhető állati eredetű takarmány (húsliszt) súlya: 15 000 kg
16. Az esélye annak, hogy egy amerikai állampolgár ma gyorsétteremben ebédel: egy a négyhez
17. Világszerte a VA Linux ellen indított polgári peres eljárások száma 2001. januárig: öt
18. Ezek közül tévesen „Linux” néven hivatkoztak a cégre: négyen
19. Általános cégek – például a Milberg Weiss –, ellen hasonló okból beadott keresetek száma az elmúlt évtizedben: 200
20. Egy teljesen Linux-alapú megoldás becsült költsége egy IBM nagygépen: 7 millió dollár
21. Ugyanennek a megoldásnak a becsült költsége a fenti példával azonos összteljesítményű Sun gépparkkal: 55 millió dollár

Forrás:
1–2: CRN
3: LinuxPlanet
4: CIO
5–10: eTforecasts
11–12: Apple
13–14: ZDNet
15–16: *Eric Schlosser* Fast Food Nation
17–18: Linux Weekly News ➲ http://www.lwn.net
19: ➲ law-phoenix.com
20–21: Infoworld

Láttuk, hallottuk, kipróbáltuk...

Az Axico Kft. jóvoltából szerkesztőségünkben időzött egy Tandberg Data SLR5 szalagos egység és egy Adaptec 3200S SCSI RAID kártya, három 18 GB-os IBM merevlemezzel. Az előbbi termék a biztonsági mentések a második pedig a sebesség, illetve a biztonságos adattárolás nagymestere.

Tandberg SLR5

A Tandberg SLR5-ös meghajtó belső változatát kóstolgattuk. A kapcsolók megfelelő beállítása után azonnal, könnyedén használatba vettük.
A ➲ www.tandberg.com honlapon körülnézve megfelelő mentőprogramokat találtunk, melyek közül mi a NovaNet8 bemutatópéldányát választottuk. A mentést végző program beállítása után elsőként a próbagépen található összes anyagot mentettük. A teljes mentés egyetlen menüpont kiválasztása után haladéktalanul elkezdődött, és a NovaNet8 program ablakában lépésről lépésre nyomon követhettük a folyamatot. A gépen található anyag több mint 3,5 GB volt. Ekkora adatmennyiséggel kellemes, de hosszadalmas feladatnak tűnt a teljes mentés, majd a hiánytalan visszaállítás végrehajtása. A program az adatok mentése közben 14 hibát jelzett, ezek olyan fájlokat jelöltek, amelyek a mentés elkezdése után még változtak. A teljes visszaállítást is sikeresen végrehajtotta, bár egy könyvtár jogosultságai megváltoztak.
A próba során a hálózati mentés szolgáltatás is tökéletesen működött. A felhasználókat könnyedén felügyeltük: csoportokat hoztunk létre, illetve hálózati számítógépeket adtunk hozzá és távolítottunk el könnyedén.
Tapasztalatainkat összegezve minden olyan vállalkozásnak ajánlhatjuk ezt az egységet, ahol viszonylag kis adatmennyiségekkel napi mentést kell készíteni.
A Tandberg Data nagyobb kapacitású meghajtókat is gyárt, ezek közül jelenleg az SLR100 a legnagyobb, mely száz gigabájt adat tárolására képes egy kazettán. Beszerezhetők Autoloader változatban is (egy meghajtóba több kazettát lehet behelyezni, így legfeljebb 800 gigabájt adatot tárolhatunk). A meghajtók linuxos támogatása nagyon jó.
Összefoglalva az eddigieket a soros rögzítési módszert alkalmazó Tandberg SLR meghajtók hosszú távon biztosíthatják adataink mentését és szükség esetén azok visszaállítását.

Adatok

4/8 GB kapacitás
256 kB gyorstár
SCSI 2-es csatolófelület
380 kB adatátvitel másodpercenként
55 másodperces átlagos fájlelérési sebesség
Ár: 174 750 + áfa
Tandberg és Imation SLR5-ös kazettákkal használható.

Az Adaptec és a három IBM

A csomagolás kibontása után a kártyán az első szembetűnő furcsaság a 64 bites PCI-csatlakozás volt. Szerencsére lehet használni 32 bites PCI csatolóban is. Telepítése nem okozott gondot sem a Debian, sem a RedHat Linux alatt, a termékkel érkező CD-n található meghajtók jól használhatók hozzá. Debian alatt szintén a lemezen lévő fájlt vettük igénybe a rendszermagfoltozáshoz és az újrafordítás után működésre bírtuk. A RAID tömbök beállításának legegyszerűbb módja, ha a kártya BIOS-át használjuk. Választhatunk a RAID 0, RAID 1 és a RAID 5 tömbök között, így már indítás után egy egységnek látjuk a három merevlemezt. A RAID 0 hatalmas adatátviteli sebességet nyújt, azonban adataink védelmét nem támogatja. A RAID 1 szintén elég gyors és a legnagyobb védelmet nyújtja lemezmeghibásodás esetén is. A RAID 5 a leggazdaságosabb adattárolást biztosító tömb, ha a három merevlemez közül egy tönkremegy, a másik kettőn adataink veszteség nélkül megmaradnak. A rendszer indulása után a RAID tömbön létre kellett hozni a fájlrendszert, ehhez mindenhol az fdisk programot használtuk. Ezután egy zavartalan formázás következett és máris készen állt a RAID a próbafutásra.

A három merevlemez teljesítménye külön-külön sem elhanyagolható, azonban ezzel a kártyával szinte csodákra képesek. A RAID 0-s tömb 51 GB tárhelyet biztosított. A 15 MB/mp-es önmagáról önmagára másolás, azt hiszem, magáért beszél. A RAID 5-ös tömb 35 GB méretű volt, az adatátviteli sebesség csökkent ugyan, viszont az adatainkat nagy biztonságban tudhattuk. Mindazon vállalkozások számára jó választás lehet, melyeknél a költségek megtérülnek a sebesség vagy a biztonság terén.

Adatok

64 bites PCI csatolófelület
kétcsatornás Ultra160 SCSI vezérlő
64 bites 100 MHz Intel i960NR mikroprocesszor és
32 MB ECC SDRAM gyorstármemória (128 MB-ig bővíthető).
Ár: 289 060 Ft + áfa
➲ www.axico.hu
➲ www.adaptec.com

Új termékek

briQ a Total Impacttól

A Total Impact kiadta PowerPC-alapú hálózati számítógépét a briQ-t. Méreteit tekintve 14,57 cm széles, 1,625

cm magas és 22,6 cm vastag, és megközelítőleg 83 dkg nyom. A briQ egy azonnal munkába állítható motorral van felszerelve, és az ipari szabvány által meghatározott félmagas meghajtóhelyre illeszthető.
A briQ PowerPC G3 vagy G4 processzort használ és testre szabható olyan alkalmazásokhoz, mint a tűzfalak, az útválasztók, a biztonsági eszközök és a webkiszolgálók. Néhány jellemzőjük: 1 MB L2 gyorstár, 100 MHz 64 bites rendszersín, legfeljebb 512 MB SDRAM, kettős 10/100 ethernet támogatás, legalább 40 GB merevlemez, RS-232 csatoló, alacsony energiaigény és távvezérelhetőség.
Adatok: Total Impact, Inc.,
295 Willis Avenue, Suite E,
Camarillo, California 93010
telefon: 1-805-987-8704
e-mail: sales@totalimpact.com
➲ http://www.totalimpact.com/

Aduva Manager

Az Aduva Inc. nemrég mutatta be az Aduva Manager névre keresztelt internetalapú szolgáltatást, amely önműködő linuxos rendszerfelügyeletet nyújt, és eközben figyelemmel kíséri a frissítést, a foltozást, illetve a rendszer bővítése közben fellépő összeférhetetlenséget vagy biztonsági hézagokat.
A hálózatról letölthető ügyfél sorra veszi a felhasználó gépén található legfrissebb összetevőket, majd elvégzi a szükséges frissítéseket. További képességek: ütemezett feladatok támogatása, figyelmeztetés a fontos eseményekre, titkosított kapcsolatok és folyamatos önjavítás. Az Aduva Manager folyamatosan elérhető az Aduva kiszolgálókról, és magáncélra ingyenesen felhasználható.
Adatok: Aduva, Inc.,
2595 East Bayshore Road,
Palo Alto, California 94303,
telefon: 1-650-858-8650
e-mail: info@aduva.com
➲ http://www.aduva.com/

Linux-támogatás HP-nyomtatókhoz

2001. januárjában a LinuxWorldön a Hewlett-Packard huszonnyolc PostScript LaserJet és üzleti tintasugaras nyomtatójának teljes körű Linux-támogatását jelentette be. További tizenhat, nem-PostScript lézernyomtatóhoz és tintasugaras nyomtatóhoz áll rendelkezésre alaptámogatás. A felhasználók az összes olyan szolgáltatást elérhetik, mint például a kétoldalas nyomtatás (duplexing), tálcaválasztás és a papírkezelési beállítások. A teljes körű támogatáshoz telepíteni kell egy frissítést a nyomtatórendszerhez, amely elérhető a ➲ http://www.hp.sourceforge.net/ címen. A jövőben az összes PostScript HP lézernyomtató teljes körű Linux-támogatottsággal jelenik meg.
Adatok: Hewlett-Packard Company,
3000 Hanover Street, Palo Alto,
California 94304-1185
telefon: 1-650-857-1501
➲ http://www.linux.hp.com/
➲ http://www.hp.hu/

K2 NAS

A Big Storage, Inc. K2 NAS rendszere egy vállalkozások számára készült hálózati tárolóeszköz, amely minden szinten hibatűrő elemekkel rendelkezik, ideértve a beépített hálózati és tárolóprocesszorokat. A K2 egytől negyven terabájtig terjedő tárhellyel rendelkezhet. A hibák kiküszöbölése érdekében kettőzött kiszolgálóprocesszorokkal, alkatrészszintű RAID vezérlőkkel, illetve a fájlrendszerek biztonsági mentéséhez és naplózásához használható pillanatkép-kezelővel szállítják. A rendszert rövid beállítással közvetlenül be lehet kapcsolni bármelyik hálózatba. A webalapú grafikus kezelőfelület könnyű kezelhetőséget és rendszerfelügyeletet tesz lehetővé, együttműködik Windows98/2000, Mac és Unix-rendszerekkel.
Adatok: Big Storage, Inc.,
19 Heron Street, San Francisco,
California 94103
telefon: 1-800-864-3789
e-mail: techsales@bigstorage.com
➲ http://www.bigstorage.com/

A Zend Technologies PHP termékvonala

A Zend Technologies bemutatta PHP-hez kapcsolódó termékvonalát, amely a vállalkozási PHP-piacot célozza meg. Honlapukon a teljes anyag elérhető. A kínálatban az alábbi termékek szerepelnek:

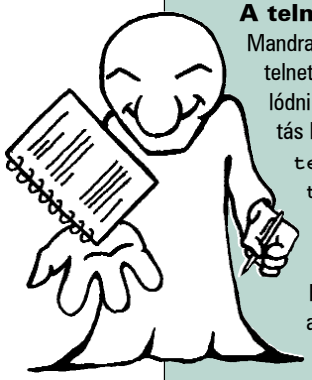
- a Zend Cache nevű, testre szabható programgyorstárazó egység, amely memóriában tartja a programok előfeldolgozott változatát;
- az Encoder Unlimited, amely magasabb szintű biztonságot tesz lehetővé azáltal, hogy a forráskód kiszolgáltatása nélkül lehet terjeszteni a programokat;
- a Zend IDE eszközgyűjtemény, amely távoli hibakeresést, szövegszerkesztést, PHP és HTML színkiemelést lehetővé tevő eszközöket tartalmaz;
- a Zend LaunchPad, amely minőségbiztosított, frissített PHP-letöltést tesz lehetővé egy grafikus felületű egységen keresztül;
- az Online Service Support, mely webalapú, folyamatosan elérhető szolgáltatás.

Üzleti- és magánszemélyek számára összeállított változat egyaránt létezik.
Adatok: Zend Technologies Ltd.,
PO Box 3619, Ramat Gan 52136, Israel
e-mail: info@zend.com
➲ http://www.zend.com/

ADSL lapkakészlet

Az Agere Systems piacra dobott egy ADSL lapkakészletet, melyet kifejezetten otthoni útválasztókhoz és kisebb hálózatokhoz ajánlanak. Az ajánlat egy PCI-csatolós ADSL hálókártyát tartalmaz, az Agere DSP-ügyfelével. A lapkát egy csatolókártyára ültették, ezen keresztül tartja a kapcsolatot a számítógéppel. Egyedi átviteli képességekkel bír, például be van építve egy ATM-et támogató réteg, ismeri az ATM és az ethernet fölött futó PPP protokollt, valamint az ATM-csomagok feldarabolásának és összeépítésének módszerét.
Adatok: Agere Systems, Inc.,
555 Union Boulevard,
Room 30L-15P-BA, Allentown,
Pennsylvania 18109-3286
telefon: 1-800-372-2447
e-mail: docmaster@micro.lucent.com
➲ http://www.agere.com/

A hónap szakmai tanácsai



A telnet nem működik?

Mandrake 7.1-est használok. A telepítés óta sem telnettel, sem pedig FTP-vel nem tudok kapcsolódni a helyi géphez. A telnet localhost utasítás kiadásakor a következő üzenetet kapom:

```
telnet: Error message-unable
to connect toremote host:
connection refused.
```

Pierre, pierre_poo@hotmail.com

Felelőtlenség megtanítani az embereket arra, hogyan állítsák be telnet és ftp-kiszolgálóikat. Tépd szét azokat a régi könyveket, amelyek a biztonságra fittyet hányva erről a két protokollról szólnak (ne add be őket a könyvtárba sem, mert a gyerekek esetleg megtalálhatják), és telepíts ssh-t. A legtöbb változat már tartalmaz könnyen telepíthető ssh-csomagokat. Ha nem linuxos gépről szeretnél belépni Linux-rendszeredbe, akkor nézd át *Rick Moen* ssh programlistáját a következő címen:
➲ http://linuxmafia.com/pub/linux/security/ssh-clients
Don Marti, dmarti@linuxjournal.com

A hangkártya beállítása

Átolvastam a hangkártya beállításával kapcsolatos összes README, FAQ és HOWTO fájlt, de még mindig nem látom tisztán, hogy mindenképpen újra kell-e építenem a rendszermagot. A operációs rendszer indításkor felismeri az SB 3.01-es kártyát, de később hibaüzenetet kapok, miszerint a kártya már vagy túlságosan régi, vagy pedig módosítani kell a beállításait. Egyszerűbb lenne, ha vennék egy olyan hangkártyát, amelyik barátságosabban viselkedik a Linuxszal?
Greg McNichol, gc@mcnichol.net

Ez elsősorban a kártyától függ. Ha ez egy ISA-alapú hangkártya, akkor könnyen lehet, hogy nem fog működni, hacsak nem valamelyik népszerű kártyáról, például a SoundBlasterről (Nem SoundBlaster-megfelelő kártyáról!) van szó. Ha azonban a kártya a PCI sínhez csatlakozik, akkor ennek az üzenetnek nem szabad megjelennie. Általában véve a PCI eszközök nagyszerűen működnek Linux alatt. Ez alól csak nagyon kevés kivétel van, azok sem hangkártyák, hanem SCSI vagy hálózati kártyák. Először is nézz utána annak, hogy nem létezik-e olyan illesztőprogram, amit kifejezetten a kártyához írtak, mivel az eszköz vezérlését mindig az első olyan illesztőprogram kapja meg, amely azt *gondolja* magáról, hogy képes ellátni a feladatot. Ez galibát okozhat abban az esetben, ha nem SoundBlaster vagy annak megfelelő kártya van a gépünkben, de az illesztőprogram azt gondolja róla.
Ha ilyen jellegű hibát tapasztalsz, akkor építsd újra a rendszermagot, és keresd meg a hangkártya-beállításoknál azt a kártyát, amit ténylegesen használsz.
Ha még nem próbáltad ki, akkor első körben adj egy esélyt az alsa programcsomagnak is.
Chad R. Robinson, crobinson@rfgonline.com

Rossz billentyűzetkiosztás

FoxPro 2.6 for Unixot futtatunk SCO Server Release 5 környezetben. A felhasználók Windows alól érik el a Tiny Term program segítségével ezt az alkalmazást. A felhasználókat szeretnénk áthelyezi linuxos környezetbe, számos terminált kipróbáltunk már (például a telnetet úgy, hogy a TERM értékét scoansira változtattuk), de nem sikerült megfelelő billentyűzetkiosztást megvalósítani.
Jorge C. Oneto, jconeto@jco.wamani.apc.org

Az SCO szerint sok Linux-változat rossz scoansi terminálbeállításokat tartalmaz. Azt hiszem, hogy az SCO szívesen elküldené a hibajavító termcap bejegyzést. Ahhoz, hogy pontosan meg lehessen állapítani, mi okozza a hibát, részletesebben körül kellene írni a jelenséget.
David Brown, david@caldera.com

FoxPro alkalmazást kell átültetned SCO Unix-környezetből Linuxra. Ahhoz, hogy százszázalékosan scoansi-megfelelővé tedd a terminálemulációt, kézzel kell módosítanod a terminfo bejegyzést.
Miután módosítottad a terminfo fájlt (ez valószínűleg a vt100 vagy az ansi), le kell fordítanod a tic fordító segítségével. A fordítóról a `man tic` parancs kiadásával részletes leírást kaphatsz. Célszerű megnézni a scoansi emulációért felelős fájlt igazi SCO-rendszerben, hogy pontosan tudd emulálni Linux alatt a billentyűzet működését. (A fájl szerzői jogi védelem alatt áll, ezért nem szabad közvetlenül lemásolni.) A billentyűzettel kapcsolatos nehézségek megoldására használhatod még az `xmodmap` parancsot is, amelyben beállíthatod, hogy milyen sorozatokat küldjön a billentyűzet az X terminálnak az egyes billentyűk lenyomása esetén.
Felipe E. Barousse Boue, fbarousse@piensa.com

Az egérgörgő beállításai

Linux alatt beállítható-e a görgős egér (Microsoft Intellimouse), hogy ugyanúgy működjön, mint Windows alatt?
Ray, wrathstocks@ragingbull.com

Természetesen! Az imwheel parancs segítségével megoldható. Ha szeretnéd megtudni az utasítás pontos formai követelményeit, akkor add ki az `imwheel -?` parancsot. További tájékoztatást a
➲ http://jcatki.dhs.org/imwheel címen kaphatsz.
Paulo Wollny, paulo@wollny.com.br

Igen. Minden szükséges adat megtalálható a
➲ http://www-sop.inria.fr/koala/colas/mouse-wheel-scroll/ címen.
Marc Merlin, marc_bts@valinux.com

A görgős egér használatát a /etc/X11/XF86Config fájlban is beállíthatod. Keresd meg a Pointer részt, és állítsd be a protokollt imps/2-re. Ezután írd be a következő két sort:

```
Option    "Emulate3Buttons"    "off"
Option    "ZAxisMapping"    "4 5"
```

Ha ezzel megvagy, indítsd újra az X-et.
Paul Christensen, pchristensen@penguincomputing.com

Rossz jelszó!

Mi szabályozza a jelszavak összetételét? Hogyan érvényesíti a rendszer a jelszavak hosszára és a nem alfanumerikus karakterek kötelező használatára vonatkozó szabályokat?
Westley L. Hespeth, hespethw@all-speth.com

A legtöbb Linux-telepítés tartalmaz egy cracklib nevű csomagot. A pam (pluggable authentication mode) ennek felhasználásával ellenőrzi a jelszavak érvényességét. A szabályokat általában a pam_cracklib.so nevű fájl tartalmazza.
David Brown, david@caldera.com

FTP kapcsolódási kísérlet nem engedélyezett gépekről

Állandó IP-címmel kapcsolódom az Internethez, és engedélyezetlen FTP-hozzáférést tapasztaltam a rendszeremben annak ellenére, hogy a hosts.deny ALL:ALL beállítást tartalmaz.
A másik dolog, amit sűrűn tapasztalok, hogy egy ismeretlen felhasználó folyamatosan kapcsolódni próbál az 1994-es tcpd kapuhoz. A próbálkozások így néznek ki:
Nov 21 11:58:10 ns1 tcpd[1994]: warning: can't get client address: Socket operation on non-socket
Nov 21 11:58:10 ns1 tcpd[1994]: connect from unknown
Nov 21 11:58:10 ns1 tcpd[1994]: warning: can't get client address: Socket operation on non-socket
Nov 21 11:58:10 ns1 tcpd[1994]: connect from unknown
Hogyan lehetne betömködni ezeket a lyukakat?
Dave Price, davep@support-one.com

Úgy tűnik, hogy a tcpd egyszerűen csak naplózza az elutasított kapcsolódási kísérleteket, vagyis a hosts.deny megfelelően ellátja feladatát.
A gép biztonságossá tételére egyébként sokkal jobb és rugalmasabb megoldást kínál az ipchains. A részletekkel kapcsolatban érdemes ellátogatni a ➲ linuxdoc.org webhelyre.
Paul Christensen, pchristensen@penguincomputing.com

Második CD-ROM-meghajtó használata KDE-ben

Második operációs rendszerként telepítettem a gépemre a 2.4-es Caldera-változatot, ami nagyszerűen működik is. Ezután bekötöttem egy második CD-ROM-olvasót (ATAPI) is a gépbe. Windows 98 alatt mindkettő látható, Linux alatt azonban csak az első. Hogyan tudom közölni a rendszerrel, hogy egy másik CD-meghajtó is van a gépemben? Egyáltalán lehetséges ez a KDE-ben?
Abderrahmane Meskine, ameskine@finances.gov.ma

Először is ki kell derítened, hogy melyik eszközhöz kapcsolódik a CD-ROM. Jelentkezz be rendszergazdaként, majd futtasd le a következő parancsot (feltéve, hogy IDE-s eszközről van szó):

```
dmesg | grep "hd[abcd]"
```

Ezután keresd meg az eredményben a CD-ROM-meghajtó(ka)t. Miután már tudod, hogy melyik eszközhöz kapcsolódik, létrehozhatsz a KDE asztalán egy olyan ikont, amely a CD-meghajtóra mutat. Kattints a jobb gombbal az első CD ikonjára, és vizsgáld meg a beállításokat. Ezután hozz létre egy új eszközikont az imént látott beállításokkal, de most azt az eszközt add meg, amelyhez a második CD-meghajtó kapcsolódik. Az eszköz pontos nevét az imént kiadott dmesg parancs eredményéből tudhatod meg (az eszköz neve /dev/hdd vagy valami ehhez hasonló lesz).
David Brown, david@caldera.com

Netscape 6 telepítése RedHat Linux alá

Kezdő Linux-felhasználó vagyok. Telepíteni szeretném a Netscape 6-ot a RedHat 7.0-s rendszerem alá. Kicsomagoltam a Netscape-től letöltött fájlokat, és elhelyeztem azokat a /usr/My_Downloads/netscape6 könyvtárba. Mit kell még tennem? Megpróbáltam kétszer rákattintani a Netscape telepítőjére, de semmi sem történt. Milyen parancsot kell kiadnom ahhoz, hogy telepíteni tudjam a programot?
Alex, aqk13@hotmail.com

Miután letöltötted és kicsomagoltad a Netscape 6 tárolófájlt, lépj be a netscape-installer könyvtárba és add ki a

```
./netscape-installer
```

parancsot. Ekkor elindul a telepítővarázsló, eldöntheted, melyik összetevőket szeretnéd telepíteni, és hová akarod elhelyezni. Ahhoz, hogy a telepítést elvégezhesd, rendszergazdaként kell bejelentkezned.
Paul Christensen, pchristensen@penguincomputing.coml

PPP igen, levelezés nem

Amikor egy új felhasználót adok a rendszerhez, a felhasználó önműködően megkapja a jogot ahhoz, hogy leveleket küldjön, illetve fogadjon. Szükségem lenne egy olyan csoportra, amelyhez ha új felhasználókat adok, a felhasználók nem kapnak jogot a levelezéshez. Jó megoldás lenne az is, ha egyszerűen meg tudnám vonni a létrehozott felhasználóktól a levelezési jogot, ezzel kapcsolatban sem találtam azonban semmiféle leírást. Fontos megjegyezni azt is, hogy olyan azonosítókról van szó, amelyekhez csak PPP-n keresztül lehet hozzáférni.
Len Elyea, lelyea@ncmc.cc.mi.us

A bejövő levelekhez készíthetsz olyan aliast, amely egy nem létező felhasználóhoz irányítja a leveleket.
A kimenő levelezés letiltására azonban semmilyen jó módszert nem ismerek.
Ha a sendmail helyett eximet használsz, akkor készíthetsz olyan fájlt, amely azoknak a felhasználóknak a listáját tartalmazza, akik számára nem engedélyezted a levelezést. A listában szereplő felhasználók leveleiben átírhatod a feladót, illetve a címzettet tartalmazó sorokat úgy, hogy azok nem létező felhasználóra hivatkozzanak, majd a megfelelő ellenőrzés elindításával megakadályozhatod, hogy nem létező felhasználók levelet küldhessenek a rendszeredből.
Marc Merlin, marc_bts@valinux.com

Nincs olyan eszköz, melyhez az ember ne folyamodna, hogy megmeneküljön a gondolkodás fáradalmaitól.
T. A. Edison

A bazár visszatér

Úgy érzem, egy korszak véget ért. Amikor a gazdaság úgy dől össze, mint egy rossz sátor, nem nehéz ilyet mondani, de szerintem itt valami többről, valami fontosabbról van szó, nem egyszerűen csak arról, hogy a történelem ismétli önmagát. Úgy gondolom, ami most történik, az a valósághoz való visszatérés elterjedése – azaz a vágyálmok elvesztésének fordítottja. Ez abból is látszik, amit nem veszünk meg: a szükségtelenül gyors számítógépeket, a humortalan vígjátékokat a televízióban, a gyanúsan jól hangzó befektetési módokat, az üres kampányígéreteket, valamint a mindenféle zagyvaságokat hajtogató hirdetéseket.

Emögött több van, mint bizalmatlanság, tömeges unalom és a pénzkidobás elutasítása, bár ezek mind szerepet játszanak benne. Ennek gyökere a józan ítélőképesség, mely abból fejlődik ki, amikor olyan emberekkel beszélgetünk, akikben megbízunk, és nem a csordaszellemből, vagy a propagandagépezetből, ami megpróbálja irányítani és ellenőrizni azt, hogy mit gondolunk, eszünk, veszünk vagy szeretünk. Még fontosabb, hogy mindez a saját önbecsülésünkben gyökerezik.

Körülbelül két évvel ezelőtt részese voltam egy beszélgetésnek. Mi négyen, a beszélgetés résztvevői, egyre inkább úgy éreztük, hogy tudunk valamit, amit külön-külön nem nagyon tudunk összefoglalni. Végül, néhány hónap után egyikünk (*Christopher Locke*, azaz RageBoy) néhány egyszerű szóba öntötte, amire mindannyian gondoltunk: „Nem vagyunk sem ülések, sem szemgolyók, sem végfelhasználók, sem fogyasztók, és a hatalmunk messze túlnyúlik azon, amit megértesz. Ez van.”

Amint meghallottuk, tudtuk, hogy ezt újra el kell mondanunk, hangosabban és részletesebben, mert emberek milliárdjai nevében beszélünk, akik bár tudják ugyanezt, még nem kezdtek el róla beszélni. Az eredmény a Cluetrain Manifesto című írás lett.

Többen gondolják, hogy a Cluetrain főleg a piac erejének – a kínálattól a kereslet felé történő – eltolódásáról szól. Ez igaz, de a Cluetrain a piac igazi értelméhez való visszatérésről is szól, a piacról, melyre az egyik legjobb rokon értelmű szó a bazár – az a hely, ahol az emberek találkoznak, üzleteket kötnek, és kultúrát építenek.

A piacok már itt vannak évezredek óta, sokkal régebben annál, hogy elkezdtük volna ezt a szót termékcsoportokra, földrajzi térségekre, demográfiai egységekre és a keresletre magára alkalmazni. Ehhez meg kellett várni az ipar kialakulását, ami körülbelül kétszáz évvel ezelőtt kezdődött. Az ipari korszakban, a piacokról főleg a kínálat szempontjából volt szó. Amikor azt halljuk, hogy a szórakoztatóelektronika, az irodai felszerelés, Mexikó és a BMW-tulajdonosok „piacok”, akkor a kínálat oldalát halljuk, amint saját magával beszélget.

A kereslet oldala számára a piac ma is többnyire olyan helyet jelent, ahova vásárolni jár – a bazárt.

A nyílt forráskód mozgalmában *Eric S. Raymond* tette a legtöbbet A katedrális és a bazár (The Cathedral and the Bazaar) című írásával. Magáról a piacról kevés szó esik, de népszerűsíti a bazárt, mint gondolatképet. Rákerestem a „bazaar” szóra, a Google keresőjével, és 355 000 találatot kaptam. Fogadok, hogy legfeljebb fele ennyi lenne, ha Eric nem terjesztette volna az igét.

A nyílt forráskód mozgalmának már két feladata is van: meg kell magyaráznia saját magát, és azt, hogy miről is szólnak valójában a piacok. Az elsőről már tudjuk, hogy nem könnyű feladat. A második sem az.

Nemzedékek óta az üzletre és a piacra is az ipar, a gyártó szemszögéből néztünk – amit annyira eltorzított a keresletről való elvonatkoztatás, hogy a gyártó szinte már képtelen megérteni a szemtől szembe, a billentyűzettől billentyűzetig zajló élet igazi természetét, a bazárokat, ahová a piacok vágyakoznak. Amit mi fogyasztói társadalomnak hívunk, az igazából termelői társadalom, olyan fáradt világnézet, mely azt hiszi, hogy az idők végezetéig osztályozhatja és szervezheti a piacokat, a saját kénye és kedve szerint.

Próbálj meg egyszer nagy gyártótól számítógépet venni. Ha a legnagyobbakhoz mégy – például a *Dell* vagy a *Gateway* – készülj fel rá, hogy meg kell adnod az osztályodat. Nem, nem azt, hogy kiszolgáló, asztali vagy laptop. A te osztályodról van szó. Mi vagy te? – kérdezik. Fogyasztó, cég vagy oktatási, esetleg államigazgatási intézmény?

Ezek azok a terelőösvények, amelyeken át a gyártó a vásárlás felé terelgeti a vevőket. Elég idegesítő, ha inkább úgy gondolsz magadra, mint szakemberre, és nem mint osztályra. De ki vagy te? Pontosabban mi vagy te? A tömeggyártók számára fogyasztó vagy. Az a feladatod, hogy fogyassz. Hogy

Jerry Michalski tökéletesen jellemző szavát idézzem, „zabálóka” vagy. Ez egy olyan lény, aki csak azért él, hogy termékeket zabáljon, és pénzt ürítsen.
Naponta dollármilliók folynak át a Dell és a Gateway pénzvezetékein. Amennyire én meg tudom állapítani, nem hallani a

zabálókák oldaláról zokszót. Mindkét cégre tágra nyílt szemekkel néznek, hiszen példás gyártók. A termékeik szinte mindig a toplisták élén, vagy ahhoz közel találhatók. Ez elmondható a terméktámogatásukra és az ügyfélszolgálatukra is. Mi a gond?
Lehet, hogy a Dell és a Gateway szempontjából nincs is. De ebben nem vagyok biztos. Úgy gondolom, hogy a Dell és a Gateway egyszerűen megmaradt az általánosan elterjedt tömegmarketing jelképeinél, valószínűleg elkerülnék ezeket, ha tudnák, hogy mennyire fogyasztóellenesek.
Éppen erről gyűjtöttem adatot a múlt héten, amikor elmentem a helyi Gateway Country Store-ba. A láthatatlan, tárgyilagos újságírósapkámat viselve, megpróbáltam a helyzethez olyan kedvezően és nyílt szemlélettel hozzáállni, ahogyan csak lehetett. Nem volt könnyű. Az első dolog, ami felkeltette a figyelmemet amint beléptem a helyiségbe, egy hibaüzenetbe fagyott képernyő volt. Senki sem vette a fáradságot, hogy elhárítsa a hibát, pedig az üzletben több volt az eladó, mint a vevő. Biztos voltam benne, hogy a hibaüzenet már órák óta ott van, ahhoz viszont csak néhány másodpercnek kellett eltelnie, hogy egy eladó srác megszólítson. Fura beszélgetés következett.

– Milyen gépe van jelenleg?
– Van egy otthoni irodánk, néhány linuxos és maces állomással – magyaráztam –, DSL jön a házig, a gépek pedig etherneten, elosztó segítségével csatlakoznak egymáshoz, majd egy útválasztón mennek ki a hálóra. Arra gondoltam, hogy kellene ven-

nem egy ilyen mindenre jó windowsos gépet, ugyanazzal a programmal, amit a könyvelőm használ, multimédiás cuccok kipróbálására…
– Akkor önnek Windows 98 kell.
– Tényleg? 98? Nem Millennium Edition vagy 2000?
– A Windows 2000-en nem lehet játékokat vagy multimédiát futtatni.
– Tényleg? Miért nem?
– Nem működik a hang a hálózaton.
– Tessék?
– A Windows 2000 egy hálózati operációs rendszer. Nincs hangja, csak olyan meghajtókkal, amiket nem támogatunk. Ha saját hálózaton dolgozik, és nem akarja, hogy az alkalmazottak MP3-akat töltsenek le, valamint hallgassanak munkaidőben…
– Én mindig ezt teszem.
– Ha játékokat futtat vagy multimédiát, ezek fogyasztói dolgok. Ehhez fogyasztói operációs rendszer kell. Ha céges hálózatot akar kiépíteni, az üzleti eladásokkal foglalkozó munkatársam tud felvilágosítást adni.
Végül nem vettem semmit. Megfigyeltem, hogy az üzletben nem voltak túl sokan, annak ellenére, hogy a karácsonyi bevásárlási láz kellős közepén voltunk. Észrevettem, hogy a Gateway megtett minden tőle telhetőt, hogy elveit alkalmazza.
A vásárlókat elosztórendszerbe terelte, ahonnan tovább tudta őket küldeni a megfelelő értékesítési csúszdába. Semmi újdonság. Az egész nagyon lejárt lemeznek tűnt. Igen, a gazdaság kicsit gyengélkedik. De elmentem a *Fry's and Best Buy*-ba is, és az ottani srácok nem nézték csekély értelműnek a vevőket. Nem tűnt úgy, hogy az az elsődleges céljuk, hogy rendszerezzék a vásárlókat. Ez talán azért van, mert ők az igazi üzletet képviselik, nem csak egy értékesítési tervet – ez alatt azt érem, hogy ők a piacról szólnak, nem pusztán a marketingről. Az igazi piacon – a bazárban – a vevőnek van valódi választása. Minél több, annál jobb. Ezt szeretjük a legjobban a Fry's and Best Buy-ban, főleg azért, mert ugyanaz a termék ott is ugyanabba az alacsony árcsoportba tartozik. A választás lehetősége számít a kínálat oldalán is, nem csak a bőség a lényeg. Azok, akik a kínálat oldalát erősítik önállóan és becsülettel, azért teszik ezt, mert ezt választották. Meglessük, hogy mit tesznek a többiek, érdeklődünk iránta, ötleteket merítünk belőle, beszélünk róla, beszállunk mi is a munkánkkal vagy elmegyünk a másik irányba, és végezzük a saját teendőnket. Hagyományos piaci kifejezéssel élve, a saját boltunkkal foglalkozunk. Így a lehető legnagyobb a választási lehetőség a mi oldalunkon és a vásárló oldalán egyaránt. Így működik a bazár – vagyis az igazi piac. Működhet ez a nagyipari gyártóknál is? Azt hiszem, hogy igen, ha arra törekszenek, hogy a lehető legtöbb legyen a választási lehetőség mind a két oldalon, a kereslet és a kínálat oldalán egyaránt. Ezt azt jelenti, hogy meg kell nyílniuk. Elérhetővé kell tenniük azt a forrást, amitől értékesek, mint cég: tudniillik a saját alkalmazottaikat.
Régen az emberi munkaerő olcsó volt és a gépek drágák, most pont fordítva van – állapította meg *Don Marti*. Ebben az esetben, a gyártóknak azt kell felszabadítaniuk, ami a céget értékessé teszi a piacon, majd a piacra bízni a dolgot, tegye, amit amúgy is a legjobban tud.
Talán akkor elkezdenek majd arra figyelni, amit mi hajtogatunk, idekinn a programbazárban. És mi is jobban figyelhetünk rájuk. A folyamat már elindult.

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

Az új népiesség

A Unix építőmesterei

A lakályos jelző a forráskód esetén azt jelenti, hogy a programozó több évvel később is előveheti az anyagot és képes átlátni, illetve biztonságosan módosítani.

Richard Gabriel

Nem beszélhetünk a programokról anélkül, hogy át ne vennénk a programgyártás nyelvezetét. Fejlesztőknek, tervezőknek és mérnököknek nevezzük magunkat. Szerkezetekről, terekről, objektumokról, keretrendszerekről, szintekről, rétegekről, összetevőkről és felületekről beszélünk. Az elkészült munkát azután úgy kezeljük, mintha valami ingatlan lenne, hiszen különböző címekhez kapcsolódó (web)helyeket építünk köré.

Mi folyik ebben a kusza nyelvezetű világban? Már jó ideje az a véleményem, hogy a nyelvhasználat azt jelzi, hogy a programipar kezd éretté válni – olyan iparág alakul ki belőle, amelyet képzett és elismert szakemberek határoznak meg, nem pedig az egyforrásos, úgynevezett „felületszolgáltatók". Az érett programiparban a Microsoft semmivel sem több vagy kevesebb, mint mondjuk a Georgia Pacific vagy a Kaufman & Broad.

Az építőiparban a fejlesztés teljesen nyilvánosan zajlik. Az építéshez használt anyagok és eljárások nem titkosak, éppen ezért a fejlesztéshez nagyon sok ember tapasztalatait fel lehet használni.

A programipar mindössze néhány évtizednyi múltra tekint vissza, ezzel szemben az építészet egyidős a kultúrával. Ez azt sugallja, hogy talán elleshetnénk néhány dolgot a régebbi iparágtól. Úgy tűnik, hogy *Gabriel Richard* is egyetért ezzel, amikor Patterns of Software című könyvében kihangsúlyozza annak fontosságát, hogy programjainkat lakályosra tervezzük. Az eszményi szerkezetet egy New England-i tanyaházhoz hasonítja: a végeredmény egy kissé kusza, de minden egyes részlet jól megfelel az igényeknek, és jól illeszkedik a többihez. A lakók könnyen megváltoztathatják a saját környezetüket, mivel azt olyan sablonok szerint építették, amely megfelel a családnak. Ezek a sablonok magukban hordozzák a kis darabokból építkezés lehetőségét. *Stewart Brand* a How Buildings Learn (Hogyan tanulnak az épületek?) című könyvében a New England-i tanyaházat a népies építkezés tökéletes példájának nevezi, és a következőt írja: „Az, hogy az építők és a lakók mit visznek át az egyik épületből a másikba, semmiféle szabályt nem követ, teljesen esetleges, az eljárás mégis mindig kellőképpen körültekintő és előrelátó."

Az építészettörténészek az 1850-es években vették át a nyelvészektől a „népies" kifejezést. A nyelvészetben ez a kifejezés egy adott terület saját nyelvjárására vonatkozik. Mindennapit jelent a szó mindhárom értelmében: elterjedt, általános és észrevétlen.

Az építészetben a népies épületeket az jellemzi, hogy szöges ellentétben állnak mindennel, ami tudományos, magasztos és művészi. Népiesnek nevezünk minden olyan épületet, amelyet nem profi építészek terveztek – más szóval a legtöbb épületet. A népies építészeti hagyományok elsősorban arra összpontosítanak, hogy miként lehet a több nemzedék során összegyűjtött tudást felhasználni az olyan hosszú távú gondok megoldásánál, mint amilyen például egy épület folyamatos karbantartása és fejlesztése. Ezzel szemben a profi építészet előszeretettel keres új megoldásokat a régi nehézségekre, s ez sokszor szerencsétlenséghez vezet.

A népies épületek folyamatosan fejlődnek. Miközben az új épületek nemzedékei egyre érettebbek és kifinomultabbak lesznek, megőrzik egyszerűségüket.

Használhatjuk-e a „népies" jelzőt a Unixra? *Neal Stephenson* az In the Beginning was the Comand Line (Kezdetben volt a parancssor) című könyvében a következőt írja: „A unixos gépek fájlrendszerei mind ugyanazt az általános szerkezetet követik. Egy gyatra operációs rendszer alatt könyvtárainkat bármilyen eszement névvel elláthatjuk, és oda tehetjük őket, ahová csak akarjuk. Unix alatt azonban a fájlrendszer legmagasabb szintjét (a főkönyvtárat) mindig ugyanaz a karakter, a / jelöli, és ez a könyvtár mindig ugyanazokat a felsőszintű könyvtárakat tartalmazza: /usr, /etc, /var, /bin, /proc, /boot, /home, /root, /sbin, /dev, /lib és /tmp.

Általában ezeknek a könyvtáraknak a belső felépítése is meglehetősen szabályos. A könyvtárnevek szinte mindig rövidítésekből állnak, és szinte sehol sem találkozunk nagybetűkkel. Ugyanis ezt a rendszert olyan emberek találták ki, akik képtelenek elviselni a rendetlenséget. A hosszú nevek hárombetűs rövidítésekké koptak le, pontosan úgy, mint a folyómederben csiszolódó kövek."

Ez a fajta kulturális hatás az, amelynek köszönhetően a Unix-hívők annyira bíznak a rendszerükben, és ez az, amely nyugodt, megrendíthetetlen, idegesítő felsőbbrendűséget kölcsönöz számukra.

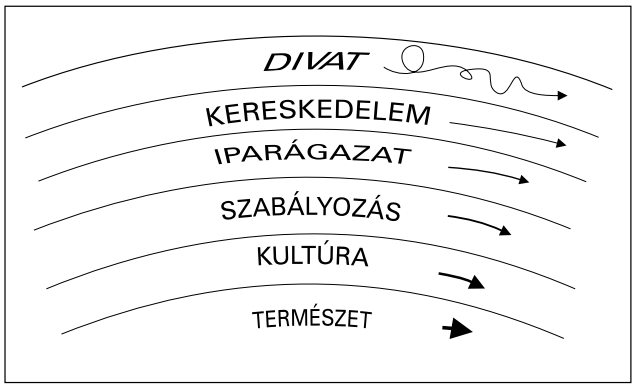
A Windows 95 és a MacOS olyan termékek, amelyeket a mérnökök egy-egy vállalat berkein belül fejlesztettek ki. Ezzel szemben a Unix nem nevezhető a szó szoros értelmében terméknek, mivel pontos összeállítása tulajdonképpen szájhagyomány útján alakult ki.

A népies építészet pontosan az ellenkezője annak, amit Brand „magazin építészetnek" nevez, mely szerinte az építészetet művészetként, nem pedig mesterségként kezeli. *Henry Glassi* így fogalmazza meg a különbséget: „Ha egy kényelmi szolgáltatás uralkodik, akkor művészetről beszélünk, ha egy gyakorlatias szolgáltatás veszi át az uralmat, akkor pedig mesterségről." Nehéz gyakorlatiasabb dolgot elképzelni, mint egy parancssor.

Stewart Brand szerint rengeteg a szakértelem az általa „Low Road" jelzővel illetett épületekben. Olyan építményekről van szó, amelyek nem feltűnőek, alacsonyak a költségeik, és egyáltalán nincs stílusuk. Brand szerint: „A világban a legtöbb munka Low Road épületekben zajlik… és még a gazdag társadalmakban is a legötletesebb alkotóerőt, főleg a fiatalos alkotószellemet, ilyen épületekben találhatjuk meg." Valószínűleg nem véletlen egybeesés tehát, hogy a mintaszerű Low Road épület példájaként a MIT 20-as épületét említi, ahol a harmadik emeleten lévő Vasútmodell klub adott otthont az 1960-as évek elején, a számítógépes betyárok első nemzedékének, akik egy sor forradalmi módszertani újítást indítottak útjukra.

Hajlamosak vagyunk azt gondolni, hogy a forradalmak gyorsan történnek, az a forradalom azonban, amely a MIT 20-as épületében kezdődött, megelőzte Moore törvényét, amelyet elsőként 1965-ben tettek közzé. Ezenkívül megelőzi magát a számítógépipart is majd egy teljes korosztállyal. Bárhogyan nézzük is, a betyárvilág hosszú ideig a számítástechnika alapjaként szolgált, hiszen a folyton változó divatok és forradalmak ellenére is viszonylag állandó maradt. Nem állítom, hogy a Unixnak nincsen kereskedelmi oldala, mindössze azt mondom, hogy kutltúrális alapjai mélyebben gyökereznek, mint a kereskedelmiek.

Az In The Clock of the Long Now című könyvében Stewart Brand a művelt világot hat rétegre osztja fel, amelyek az idő múlásával eltérő mértékben változnak. A felosztást ekképpen ábrázolja:



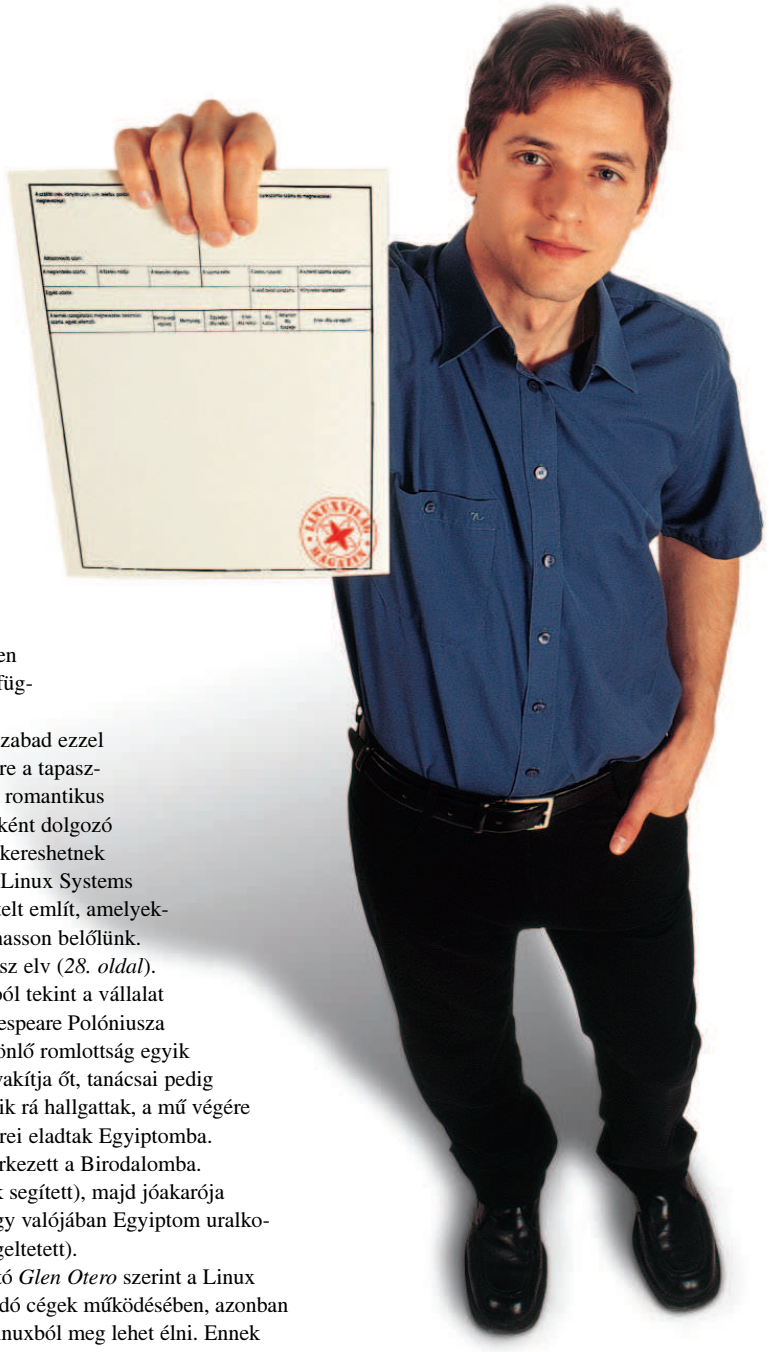
A alsó rétegek lassabban, a magasabban lévők pedig gyorsabban módosulnak. A gyors rétegek újításokat, a lassúak pedig megbízhatóságot hoznak. A tanulás így a folytonossággal társul. – állítja Brand.
A betyárvilágot és a programkereskedelmet egyaránt új és igen érdekes szemszögbe helyezi. A betyárvilág, amely egy szinttel a természet felett helyezkedik el, gondosan őrzi a programok mesterkéletlen értékeit. A GNU kiáltvány szerint például az ingyenes program olyan, mint a levegő. Egy szinttel feljebb megtaláljuk a szabályozást, a betyárvilág állandó kínlódását a felhasználási szerződésekkel. Ezek a szerződések egy réteggel feljebb ipari ágazatot alkotnak: olyan programok és protokollok ezek, melyek annál értékesebbek, minél több helyen vannak jelen. Talán nem túlzás azt mondani, hogy a betyárvilág természetes módon ráérzett arra, hogy mi tesz egy programot értékessé. Ezt az elgondolást láthatjuk megtestesülni az Internetben is, amely legalább három természetes értéket mondhat magáénak: senki sem birtokolja, mindenki használhatja és bárki továbbfejlesztheti.
Mindezek a lehetőségek beépültek az Internetbe (és az ingyenesen elérhető programfejlesztő eszközökbe), elsősorban anyagi okból. Ez az oka annak, hogy a GNU projekt azt mondja: „az embereknek lehetőséget kell adni arra, hogy szabadon használhassák a programokat minden olyan célra, amely a társadalom számára hasznos”.
A GNU eszközök és a többi ingyenes program mellett ezt az elvet követi a Linux, a Bind, a TCP/IP, a sendmail, az Apache, a Jabber és a SOAP is. Ezeket a programokat (javarészt) senki sem birtokolja, mindenki használhatja és bárki továbbfejlesztheti.
Ezek azok az értékek, amelyek nem az üzleti életből jönnek (és nem is jöhetnek onnan). Ezek nem kereskedelmi értékek. A program kultúrarétege azonban – amit kereskedelemnek nevezünk – attól a társadalmi háttértől függ, amely a betyárvilágból és általában véve a Unixból fejlődött ki. Nyilvánvalóan ide tartozik az Internet, és ezenkívül még sok más célszerű és igen elterjedt eszköz. Ezek ingyen vannak, mint a levegő.
Nagyon fontos, hogy megértsük azt, hogy milyen viszonyban állnak egymással ezek a rétegek, mivel számtalan félreértés és rossz döntés születhet úgy, ha például a kereskedelmi érdekek saját javukat szem előtt tartva próbálnak meg kialakítani egy iparágat, vagy amikor a szabályozás megpróbálja átalakítani a közösség igényei szerint a kereskedelmet.
Michael Polany azt mondja, hogy az átfogó tudatok (ilyen például a társadalom) a valóság különböző szintjeinek logikus együttesei, és léteznek olyan elvek – úgynevezett kerületi feltételek –, amelyek révén az egyes szintek biztosítani tudják azokat a feltételeket, amelyektől a felsőbb szintek függenek. *Phil Mullins* ezt a következő módon fogalmazza meg: az alacsonyabb szint olyan korlátokat szab, amelyeket a felsőbb rétegek nem tudnak áthágni. Az alacsonyabb réteg meghúzza a határokat, de szabadon hagyja a lehetőségeket. Egy alacsonyabban lévő szint nem határozza meg pontosan a magasabb szinteket… egyetlen szintnek sincs hatalma a saját kerületi feltételei fölött, így nem is létezhet egy magasabb szinten. Ennek megfelelően tehát, az elektronikus kereskedelmet nem az Internet hozta létre, mégis igaz az, hogy az elektronikus kereskedelem teljes mértékben az Internetre van utalva.
Miközben tehát az ingyenes programokat támogató megmozdulások a programipar kereskedelmi övezete számára kereskedelemellenesnek tűnhetnek, ezek a kereskedelmi és a divatrétegtől teljesen függetlenül történnek. Ezek sokkal tartósabb, tisztán műveltségi, természetes dolgok. Ha megnézzük a piaci ágazatot szegélyező kerületi feltételeket, láthatjuk azt is, miért ütköznek gondokba a programkészítő cégek olyankor, amikor az iparágukat felülről lefelé próbálják meg beilleszteni a rendszerbe. A Microsoft talán sikeres volt ebben, de csak átmenetileg. Manapság mindannyian (beleértve a Microsoftot is) azt érezhetjük, hogy a program alulról felfelé kezd növekedni, és ez elsősorban az Internetnek köszönhető, illetve annak a kultúrának, amely azt létrehozta. Láthatjuk, hogyan fejlődnek az olyan protokollok, mint a SOAP vagy az XMP/RCP, amelyeket a Világhálón történő közlés elősegítésére fejlesztettek ki. Ezek a módszerek teljesen nyitottak és bárki számára hozzáférhetőek. *Dave Winer* és vállalata, az Userland, mindkét fejlesztésben részt vett. Ugyanez igaz a Microsoftra és a Developmentorra is. Amikor megkérdeztem Dave-től, hogy hogyan zajlott a fejlesztés, azt mondta, hogy a Microsofttal és a Developmentorral végzett közös munka volt a legeredményesebb azok közül, amelyekben cégével részt vettek.
A Microsoft a személyi számítástechnikából érkezett, és örökre ott is fog maradni. Egy olyan világban is boldogulnia kell azonban, ahol a számítástechnika egyre inkább a szűkebb-tágabb környezetéhez csiszolódik. A Unix már a kezdetek óta fel volt készítve erre.
Craig Burton arra hívja fel a programkészítő cégek figyelmét, hogy „az igazi kihívás az, hogy mindenütt jelen lévő iparágazatot építsünk ki, miközben folyamatosan növeljük részvényeink értékét”. A feladat egyáltalán nem könnyű, a megoldásban azonban két dolog is segíthet. Az egyik, hogy összhangba kerüljünk azokkal a fejlesztőmérnökeinkkel, akik az ágazatot készítik, a másik pedig az, hogy képesek legyünk felismerni, hogy mi válik be hosszú távon is.
A Long Now alapítvány hét alapelvet fogalmaz meg azon cégek számára, amelyek hosszú életűek és értékállók szeretnének maradni:

- Szolgálja a hosszú távú terveket.
- Vegye kézbe a felelősséget.
- Jutalmazza a türelmet.
- Figyeljen a mélyebb értelemre.
- Lépjen szövetségre a versenytársakkal.
- Ne kösse magát semelyik oldalhoz.
- Használja ki a hosszú távú lét előnyeit.

A cégek állandóan tanulnak – még azok is, amelyeket elsodort a divat. Vegyük például az Apple-t. A legjelentősebb változás, amely az Apple berkein belül történt az elmúlt évben, nagyon távol áll a divattól, talán éppen ez az oka annak, hogy keveset is hallhattunk róla. A változás a szabályozás szintjén történt. Miután hetvenezer szaki együtt látott neki az Apple BSD-alapú Darwin operációs rendszerének továbbfejlesztéséhez, az Apple válaszul a „természetes” irány felé mozdította el a forráskódra vonatkozó felhasználási szerződést. Az új szerződés természetesen még nem olyan, mint a GPL, de sokkal közelebb került ahhoz. *Chris Bourdon*, az OS X termékmenedzsere azt mondta: „Megfoghatod a Darwint, és azt tehetsz vele, amit akarsz. Bárki hozzáférhet.”
Még szép, hiszen ez is Unix.

Doc Searls
a Linux Journal szerkesztője és a Cluetrain Manifesto társszerzője.
➲ www.cluterain.com
➲ www.longnow.org

Tanácsadás



E havi fő témánk a tanácsadásról szól. Írásainkat főleg azoknak szánjuk, akik szeretnének tanácsadással foglalkozó vállalkozást indítani, netán részesei ilyennek, illetve azoknak akik éppen fontolgatják, hogy tanácsadó segítségét veszik igénybe. Egyre többen használnak Linuxot, így a linuxos gyakorlat és tapasztalatok értéke nagyon gyorsan nő, akárcsak a linuxos szakemberek iránti igény is. Rendszerfelügyelettel foglalkozó decemberi számunkban már szóba került, hogy mit is tehetünk, ha szakemberként kell együtt dolgoznunk a témában képzetlen munkatársakkal. A tanácsadónak viszont egyáltalán nincsenek munkatársai, így leginkább csak magára számíthat. Természetesen a szabadúszásnak is számos előnye van, például a függetlenség és a rugalmas munkaidő.

A tanácsadó egy kicsit a zsoldosokra hasonlít (ha szabad ezzel a hasonlattal élni), oda megy, ahová hívják, fegyvere a tapasztalat és a találékonyság. Nyilván sokakat vonz ez a romantikus kép, például a csalódott és csakis magányos farkasként dolgozó szakértőket. Nos, a linuxos tanácsadás során azzal kereshetnek pénzt, amit szeretnek. *Marty Larsen* jelenleg a VA Linux Systems tanácsadói részlegének vezetője. Cikkében öt feltételt említ, amelyeknek teljesülniük kell ahhoz, hogy jó tanácsadó válhasson belőlünk. Nála első helyen áll a foglalkozz azzal, amit szeretsz elv (*28. oldal*).

A külsős tanácsadó azért fontos, mert új nézőpontból tekint a vállalat gondjaira. Vegyünk két példát az irodalomból. Shakespeare Polóniusza (Hamlet dán királyfi tanácsadója) az országot elözönlő romlottság egyik fő jelképe a műben. Helyzetének megerősödése elvakítja őt, tanácsai pedig csupán unásig ismételt közhelyek. Ő s mindazok, kik rá hallgattak, a mű végére meghalnak. Vagy ott van a bibliai József, kit testvérei eladtak Egyiptomba. Másik országból, más kultúrából, másfajta hitből érkezett a Birodalomba. Az „üzlet” először kicsiben működött (rabtársainak segített), majd jóakarója ajánlása folytán a fáraó személyes tanácsadója, s így valójában Egyiptom uralkodója lett (ehhez azért egy kis jóstehetség is szükségeltetett).

A saját, Linux Prophet nevű tanácsadó cégét irányító *Glen Otero* szerint a Linux új szakterületként fontos szerepet játszhat a tanácsadó cégek működésében, azonban figyelmeztet, ne számítsunk arra, hogy pusztán a Linuxból meg lehet élni. Ennek oka, hogy az operációs rendszert még mindig inkább a nyílt forrású fejlesztés iránt elkötelezettek részesítik előnyben, és nem a nagyvállalatok. Tapasztalataira alapozva Glen azt állítja, hogy a Linux csupán az eszközök egyike, s a tanácsadónak alaposan ismernie kell ahhoz, hogy a leghatékonyabban alkalmazhassa (*27. oldal*).

A szintén független *Joshua Drake* az ügyfelek számára szolgál hasznos ötletekkel. Írásából megtudhatjuk, milyen szempontokat érdemes mérlegelnünk a megfelelő tanácsadó kiválasztásakor, illetve a szerző néhány olyan helyet is bemutat, ahol a keresést elkezdhetjük (*30. oldal*).

Bár nem a Vezérfonal része, de *Dennis Roman Gesker* cikke is e témához kapcsolódik: arra a kérdésre keresi a választ, hogyan válasszunk fejlesztési megoldást terveink megvalósításához (*44. oldal*).

Richard Vernon

Linux-próféta

A Linux eszköz, nem pedig kész megoldás.

A Linux Prophet (Linux-próféta) nevű tanácsadással foglalkozó céget vezetem a kaliforniai San Diegóban. Tulajdonképpen én magam is Linux-próféta vagyok. Beowulf-alapú telepeket tervezek és építek biotechnológiával és oktatással foglalkozó ügyfelek számára. A Linux Journal szerkesztőségétől kaptam a felkérést arra, hogy osszam meg az olvasókkal azokat a tapasztalataimat, melyeket a munkám során szereztem. Boldogan tettem eleget a felkérésnek. Azok az emberek, akik a Linux köré szerveznek tanácsadó céget, valószínűleg vitába szállnának azzal a ténnyel, hogy a Linux a világ többi része számára egyáltalán nem egy nyilvánvaló választás. Egyelőre még nem. Úgy értem, nem szabad azt hinni, hogy rögtön szerződéshez jutunk pusztán azért, mert a Linux egy pompás rendszer.

Igen részletesen el kell magyaráznunk az ügyfélnek, hogy miért is kell pont Linuxot használnia. A Linux és általában a nyílt kódú programok oktatása fontos része a munkámnak. A Linux-oktatás azonban gyakran nem elég a szerződés nyélbe ütéséhez.

Ugyan miért? – kérdezheti az olvasó. Azért, mert a döntéshozók nem vakbuzgó Linux-hívők, sőt elképzelhető, hogy korábban még csak nem is hallottak róla. Ha tárgyalópartnereink hallottak is a Linuxról, akkor sem törődnek majd az árakkal, a monopóliumokkal, a programok birtoklásának erkölcsi kérdéseivel és a világuralommal. Jó teljesítményt akarnak és megbízható megoldásokat.

Miért nem törődnek az árakkal? Egyrészt azért, mert már hozzászoktak ahhoz, hogy ezek a rendszerek általában igen sok pénzbe kerülnek. Másrészt a komoly játékosok tudják, hogy minden pénzbe kerül, és ami ingyenes, azzal általában valami bibi van. Nem érdemes tehát túlságosan kihangsúlyoznunk azt a tényt, hogy a Linux ingyenes, mivel ezzel elriaszthatjuk a leendő fizetőképes vásárlókat, esetleg engednünk kell majd árainkból. Ha nem a rendszer mérsékelt költségeit hangsúlyozzuk, hanem a benne rejlő szabadságot, akkor elkerülhetjük az imént említett buktatókat.

Monopóliumok? Ha jól látom, a bírósági ítéletek és az ezeket követő fellebbezések egyáltalán nem csökkentették a felhasználók körében a legelterjedtebb asztali operációs rendszer népszerűségét. Nevezhetjük a Microsoftot, aminek csak akarjuk, ez a cég azonban a legtöbb intézményben már megvetette a lábát, és nem fog hirtelen eltűnni. Erkölcs? Leendő ügyfeleink biztosan nem gondolják, hogy a felhasználási szerződés, illetve a szerzői joggal védett program a gonosz megtestesülése. Én magam sem ismerek a nyílt fejlesztés hívein kívül olyan embereket, akik az ingyenes programoknak erkölcsi jelentőséget tulajdonítanak.

Mi a lényeg? Az, hogy senki sem fogja pusztán a Linuxért igénybe venni a szolgáltatásaimat. Azért kérnek fel, mert jó munkát végzek. A megoldás elkészítéséhez a lehető legjobb eszközöket használom, függetlenül attól, hogy a forráskód nyílt vagy sem. A Beowulf-telepek esetében a legjobb eszköz a Linux.

A Linuxszal először néhány évvel ezelőtt kerültem kapcsolatba, amikor rövid, de érdekes kirándulást tettem a bioinformatika világában. Amikor úgy döntöttem, hogy otthagyom a laboratóriumot, és kizárólag a számítógépes biológiára fogok összpontosítani, ehhez megfelelő eszközökre volt szükségem. Ekkor hallottam először a nyílt forrású programfejlesztésről, és ennek kapcsán olyan eszközökről, mint a Linux és a Perl. A Perl hatalmas terheket vett le a vállamról a különféle projektek során. Rájöttem arra is, hogy az általam feltett kérdések megválaszolása igen erőforrás-igényes feladat – itt került képbe a párhuzamos programozás. Ebben az időben kezdtek elterjedni a géptelepek, és a tudományos feladatokhoz használt operációs rendszer a Linux lett. Lassan formálódott bennem egy látomás olyan linuxos telepeken futó bioinformatikai programokról, amelyek központi szerepet tölthetnek be az emberi génekmegismerésében. Ezzel a célkitűzéssel láttam munkához.

Az informatika küzdőterére tudományos háttérrel érkeztem ugyan, de sem számítástechnikai, sem mérnöki múlttal nem rendelkeztem, és ez egészen egyedi rálátást adott a Linuxra.

A linuxos szaktanácsadással kapcsolatban – véleményem szerint – a legfontosabb az, hogy ne a Linuxról szóljon. A Linux csak egy eszköz. Kétségtelen, hogy ez az eszköz egészen különleges képességekkel bír, de akkor is csak egy eszköz, nem pedig kész megoldás. A párhuzamos programozás, az emberi génekanulmányozása és a számítógépes biológia már a Linux feltűnése előtt is léteztek, szorosan kapcsolódtak egymáshoz, és ezek most sem függenek semmilyen módon a Linuxtól. A Linux csak a közelmúltban vált e kutatási területek elsődleges eszközévé. Mindezzel arra próbálok rávilágítani, hogy ha valaki önálló Linux-tanácsadóként sikeressé szeretne válni, akkor a Linux magasztalása mellett más szolgáltatásokat is nyújtania kell, és egyedi nehézségeket is meg kell oldania. Fejlesszük a szaktudásunkat műsorszórók szakértőjeként vagy rendszergazdaként – végezzünk jó munkát! Jobbat, mint azok, akik azt a bizonyos másik operációs rendszert használják. Linuxos ismereteink ily módon fokozatosan gyarapodnak, míg végül mindentudó Linux-szakértővé válhatunk. Kezdjük azzal, amit már tudunk, és a Linux talán segít abban, hogy még jobbak lehessünk. Ha nem látjuk tisztán a feladatunkat, akkor semmilyen eszköz sem segíthet rajtunk. Minden ember más, és vannak olyan emberek is, akik nem alkalmasak arra, hogy saját vállalkozást indítsanak, ezért nem tudok olyan általános megoldást kínálni, amely mindenki számára megfelelő lenne. Néhány fontos dolgot azonban szeretnék kihangsúlyozni:

- Egy egyszemélyes tanácsadói cég beindítása nem könnyű feladat. Kezdetben elég csak részmunkaidőben foglalkozni a szaktanácsadással, és ha van rá mód, akkor nem egyedül, hanem például egy megoldásszállító cégnél (ilyen például a VA Linux). Pusztán szakkönyvek olvasásával lehetetlen felkészülni arra a számtalan különböző helyzetre, amelyekkel munkánk során találkozni fogunk.
- Ha úgy döntünk, hogy egyedül vágunk bele a dologba, már a legelején el kell gondolkoznunk azon, hogy hogyan fogjuk számlázni a munkánkat. Ha napidíjat szeretnénk kérni, akkor találjunk ki egy jó kiindulópontot. Az alapösszeg kitalálásánál mind a piaci árakat, mind a saját költségeinket vegyük figyelembe. Lehet, hogy egy hónapban csak húsz órát fogunk kiszámlázni egyéni szaktanácsadóként, és ennek fedeznie kell a költségeinket! Számításba kell vennünk azt is, hogy nem minden hónapban tudunk ennyi órát számlázni, sőt lehetnek olyan hónapok, amikor egyáltalán nem is lesz szerződésünk.

Glen Otero
(gotero@linuxprophet.com) az immunológia és a mikrobiológia doktora, emellett a Linux Prophet nevű szaktanácsadó vállalkozást irányítja a kaliforniai San Diegóban. Kedvenc időtöltése a szörfözés az óceánon.

Mi a sikeres szaktanácsadó titka?

Marty Larsen ajánlásai öt pontban.

Tanácsadói munkám kezdetén boldognak és sikeresnek éreztem magam, mert független voltam és dolgozhattam, ráadásul azzal foglalkoztam, amit szeretek. Az elsődleges célom az volt, hogy informatikus mérnökként keressem a kenyerem, és ne egy katonai vállalatnak dolgozzak. Egyszerű alapelveimnek köszönhetően a gazdasági növekedés visszaesése ellenére sikeres voltam. Szerintem a tanácsadói munka **ELSŐ**, legfontosabb pontja, hogy azzal foglalkozzunk, amit szeretünk.

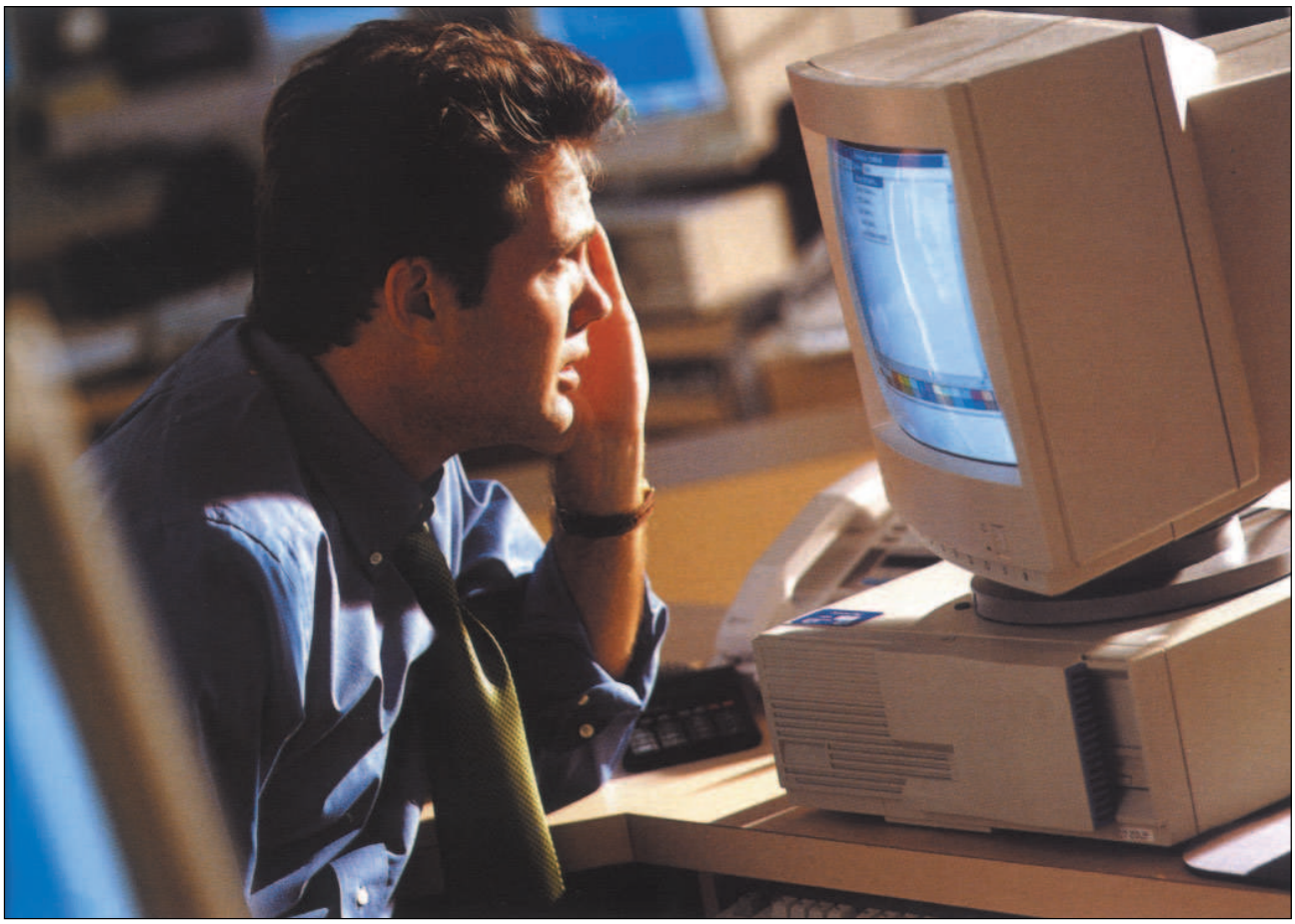
Az idő múlásával azonban egyre inkább korlátokba ütköztem. Az első időkben, „bérelhető” voltam, munkáról munkára vándoroltam, és a szabad időmben végeztem a papírmunkát. A kevésbé vonzó kötelező munkák – például a könyvelés, az adóbevallás – mindig legutoljára maradtak, amikor már feltétlenül meg kellett csinálni.

Ez az üzleti modell poénnak jó volt, de nem lehetett bővíteni. A dolgok fontossági sorrendje idővel megváltozott, és egyre inkább igényeltem a biztos, rendszeres jövedelmet. Elkezdtem azon gondolkodni, hogyan lehetne a tanácsadáshoz fejlettebb üzleti modellt alkalmazni. Amikor megvizsgáltam a régi modell működését, arra jutottam, hogy az életet akkor látom szépnek, amikor az ügyfelekkel foglalkozhatok, és akkor érezem magam nyomorúságosan, amikor a köztes feladatokkal kell megküzdeni – a számlákkal foglalkozni, hirdetni a szakértelmemet, miközben megállás nélkül keresnem kellett a sok pénzt hozó, hosszú távú, érdekes szerződéseket.

Üzleti nyelvre lefordítva, ez volt az a pont, amikor nem lehetett újabb beruházás nélkül további növekedést elérni. Két választásom volt: felvehettem segítséget vagy csatlakozhattam egy működő céghez, mely hozza a megbízásokat, cserébe kiegészítem a szolgáltatásaikat a szakértelmemmel. Az új, kiterjesztett modellem két pillérre támaszkodott.

Szerződtem egy kis, számítógépes céggel, az Advanced Engineering Labsszal, és „megoldásokat” adtam el nekik. Azzal kerestem pénzt, hogy megoldásokat dolgoztam ki az AEL ügyfelei számára, az AEL pedig gépeket adott el az ügyfeleknek megoldásaim megvalósításához. Együtt teljes és megbízható megoldást biztosítottunk az ügyfelek számára. Az együttműködés sikeres volt, és eléggé egyedinek bizonyult az asztali gépek korának hajnalán.

Az üzlet másik pillére az oktatás lett. Elektronikus áramkörök tervezését tanítottam a Heald Főiskolán. Az oktatással egyúttal fejlesztettem az előadói készségemet is, ez ugyanis nagyon fontos, ha valaki felfelé kapaszkodik a tanácsadói világ szamárlétráján. Így ismét „sikeres” tanácsadónak érezhettem magam.



Ügyfeleim egyre nagyobb cégek lettek, így a tevékenységem is egyre összetettebbé vált. A keresletben jelentkező növekedés arra kényszerített, hogy feladjam a tanítást, és minden időmet a tanácsadói munkának szenteljem.

A sikeres tanácsadói munka **MÁSODIK** fontos pontja a folyamatos továbbképzés. Akár hivatalos oktatás keretein belül, akár önművelés útján, például szakmai tanácskozások látogatásával és szaklapok rendszeres olvasásával.

Annak a tanácsadónak, aki nem képezi tovább magát folyamatosan, olyan hamar lejár a szavatossági ideje, mint egy pohár joghurtnak. Nem tudom eléggé hangsúlyozni a tanulás fontosságát. A feladataim egyre nagyobbak és összetettebbek lettek, ezzel párhuzamosan az önéletrajzom is egyre meggyőzőbbé formálódott. Az ügyfelek mindig kíváncsiak arra, hogy a tudásunk mennyire naprakész, és milyen feladatokat valósítottunk meg.

A **HARMADIK** sarkalatos pont: mindig legyen a kezünk ügyében könnyen elérhető, részleteket is tartalmazó adattár.

Amint egyre több munkát tudhattam magam mögött, felhalmoztam egy programokból, diagramokból, dokumentumokból, eszközökből és tapasztalatból álló tudásbázist. Újból és újból feltalálni ugyanazt, bűn. Annak a független tanácsadónak, aki egyedül is talpon akar maradni, mindenképpen szüksége van egy bármikor elérhető, rendszerezett adatgyűjteményre. Ahogy egyre bonyolultabb feladatokat oldunk meg, el kell sajátítani az összetett megoldások ügyfél számára is érthető továbbadását. Ebben jelentős segítséget nyújt a jól bevált adat-, eljárás-, módszer- és programtár, amivel megalapozhatjuk sikerünket és bizalmat kelthetünk ügyfeleinkben. Minden ügyfél szereti látni, amiért fizet.

A sikeres tanácsadás **NEGYEDIK** pontja: mindig készítsünk alapos felmérést. Az ügyféllel becsületesen és tárgyilagosan kell közölni a felmérés eredményét.

Ha egyszerű bérelhető munkaerőnél többet szeretnénk nyújtani, akkor egyszerre kell kereskedőnek, lélekgyógyásznak, mérnöknek, könyvelőnek és üzletembernek lennünk. Pontosan meg kell értenünk az ügyfél igényeit, a gondjait és azt, hogy mennyire sürgős megoldást igényel a helyzete. Az ügyfél megnyerésének legbiztosabb módja, ha a munka megkezdése előtti felmérést jól megoldjuk, így tisztábban láthatjuk igényeit. Másképpen fogalmazva, mielőtt beleugranánk a „megoldásnyújtásba”, meg kell győznünk az ügyfelet, hogy pontosan értjük az ő „egyedi” és különleges helyzetét. Mindig hangsúlyozni kell a felmérés fontosságát! Ez lehet egyszerű – például ha csak belenézünk a kódba –, de lehet összetett feladat is. Ilyen eset, amikor fel kell térképezni és leírni az ügyfél teljes üzletmenetét. A felmérésnek elfogulatlannak kell lennie. Nem elhanyagolható szempont az sem, hogy hasznos legyen az ügyfél számára. A sikeres tanácsadó nem megy fejjel a falnak. Nagyon rossz szolgálatot tennénk az ügyfélnek, ha nem mérnénk fel pontosan a helyzetet, nem adnánk tudtára a kutatás eredményét, azután nem együtt dolgoznánk ki a megfelelő megoldást. Ha ragaszkodunk a tárgyilagos megközelítéshez, ha alapos munkát végzünk, ebből az ügyfélnek is haszna származik, ezért természetesen nem ingyenes. Minden ügyfélnek ajánlottak már más ürgék „ingyenes” felmérést. De semmi sincs ingyen! Az ügyfél mindig azt kapja, amiért fizet, és a legtöbben közülük tudják is ezt. Ha nem, akkor érdemes meggondolni, hogy szerződést kössünk-e velük. Akik nem készítik el a házi feladatokat, gyakran nincsenek megelégedve a végeredménnyel.

A sikeres tanácsadás **ÖTÖDIK** pontja, hogy tegyük emlékezetessé a befejezést.

Szóval olyan tanácsadók vagyunk, akik azzal foglalkozunk, amit szeretünk, folyamatosan részt veszünk továbbképzéseken, szorgosan bővítjük tudásbázisunkat és értékes felméréseket készítünk. Mi kell még? Elkészíteni a munkát? Igazából maga a munka a feladat könnyebbik része, feltéve, hogy szeretjük azt, amivel foglalkozunk.

A következő legfontosabb dolog, amire gondot kell fordítanunk – a befejezés. A bokszoló akkor is győzhet, ha az edzéseket elhanyagolta, az első kilenc menetben nem küzdött túl jól, és csak az utolsó pillanatban ütötte ki az ellenfelét. Ezzel ellentétben, ha a bokszolónk keményen készül, elemzi ellenfele küzdőstílusát, és szemet gyönyörködtetően küzd az első kilenc menetben, de a tizedikben egy másodperces figyelmetlenség miatt kiütik, akkor ő veszít. Ugyanez érvényes a tanácsadókra is. A jelenlétünkkel és a munkánkkal tegyünk jó benyomást az ügyfélre, végül hagyjunk benne kellemes emlékeket. Tartsunk befejező előadást a munkánkról, foglaljuk össze, hogy mit vittünk véghez a munkánk időtartama alatt, adjuk át az ügyfélnek a végső jelentést, amihez csatoljunk egy másolatot minden olyan iratról, amiben elismerte az érdemeinket. Tegyünk meg mindent, hogy emlékezetes legyen a befejezés. Ez az utolsó és legjobb esélyünk arra, hogy az éppen lejáró szerződésünket újabbak kövessék. A befejező összefoglaló akkor is nagyon fontos, ha a szerződés feltételeit nem sikerült maradéktalanul teljesíteni: hangsúlyozza az elvégzett munkánkat, feltárja azokat a nehézségeket, amelyekkel szembe kellett néznünk és gyakran újabb lehetőségekhez vezet.

Marty Larsen
informatikai szaktanácsadással foglalkozik tizenöt éve. Jelenleg a VA Linux Systems Inc. tanácsadói részlegének elnökhelyettese. Tanácsadóként olaj- és gázipari, valamint elektronikai nagyvállalatoknak dolgozott. Az olaj- és gázipari, valamint a távközlési tanácsadással foglalkozó EDS-nél igazgató volt, az Intel Corporation IOS adatközpontjában a Global Capacity Planning osztályt vezette. Villamosmérnöki diplomáját a Colombus Egyetemen szerezte.

Rántsuk le a leplet!

Őszinte és alapos áttekintés a különféle tanácsadókról.

Általánosan elterjedt az informatikában a tévhit, hogy Linuxhoz nem létezik magas színvonalú támogatás. Nemrég beszélgettem barátommal, akinek gondjai akadtak egy Windows NT kiszolgálóval, mely fájl- és nyomtatókiszolgálóként üzemelt egy közepes méretű hálózatban. Barátom azért volt elkeseredve, mert a Windows NT nem tudja megfelelően kezelni a nagy terheléssel járó helyzeteket. Ha egy felhasználó nagyobb állományt mozgatott a hálózaton keresztül, a kiszolgáló terhelése jelentősen növekedett.

Kérdeztem tőle, miért nem használ más megoldást, például Linuxot Sambával. A válasza nagyon érdekes volt. Nem úgy hangzott: „Ugyan már, Linux? Viccelsz?” és nem is úgy: „A Linux semmit sem ér!”, hanem a következőt mondta: „Mi történik, ha valami hiba jelentkezik? Most komolyan, ki lesz a felelős, ha ilyesmi történik?” „Furán” néztem rá, hisz a cégem (➲ http://www.commandprompt.com/) egyik működési területe éppen a linuxos felügyeleti szolgáltatás.

Más ügyfelekkel folytatott beszélgetéseim alkalmával is számos hasonló választ kaptam. A sajtó óriási erőfeszítéseket tesz, hogy népszerűsítse a Linuxot, emellett ugyanakkora erőt fektet a lehúzásába is. Végül is a sajtó a Linux-ipar első számú marketingeszköze, de szerintem az egészet elszúrta. Arra gondolok, hogy magas színvonalú, kereskedelmi vállalkozások, melyek linuxos támogatást nyújtanak, igenis léteznek. Ehelyett a sajtó az internetalapú támogatási forrásokra, például a Usenetre összpontosított, esetleg még a Linux Documentation Projectre ➲ http://www.linuxdoc.org/.

Az olyan vállalatokat is meg-megemlítik, mint a LinuxCare vagy a RedHat. A baj csak az, hogy a RedHat, a LinuxCare vagy a comp.os.linux.networking nem kézzel foghatók vagy dicsérhetők, és ha úgy hozza a sors, nem lehet megragadni a grabancukat sem. Ne gondoljátok, hogy gonosz vagyok, vagy én lettem az ördög ügyvédje. A nagy cégeknek, mint a RedHat, a VA Linux és a LinuxCare, megvan a maguk helye a támogatási piacon. Valóban nagyszerű telefonos támogatást nyújtanak – feltéve, hogy minden gond megoldásáért 325 dollárt szeretnénk fizetni a RedHatnek. Van egy rossz hírem, ez a szolgáltatás még drágább, mint a Microsofté. A LinuxCare támogatása már óránként 180 dollárért igénybe vehető, és máris nyeregben vagyunk. Remek! Óránként 180 dollárt fizetek, és csak remélhetem, hogy tudnak segíteni. Na de mi történik, ha mégsem? A RedHat legalább megoldásonként számláz.

Ha közepes vagy nagy méretű vállalattól telefonálsz, a RedHat vagy a LinuxCare jó választás lehet. Olyan szolgáltatást nyújtanak, amire számos kisebb tanácsadó nem képes, és bármikor elérhetők. Általában gyorsabban előállnak a megoldással, hiszen vannak embereik, akik egész nap csak a telefonhívásokat várják. Rengeteg mozgósítható belső erőforrással bírnak. Ha az egyik mérnök nem tudja a választ, a mellette ülő másik talán igen.

Viszont nem az ár az egyetlen hátrányuk. A túloldalon ülő mérnök nem látja, hogy mi történik. Pontosan el kell mondani neki, és eközben félreértések, pontatlanságok fordul(hat)nak elő. Természetesen, ha a helyzet úgy kívánja, a mérnök számára távoli hozzáférésre is adhatunk engedélyt.

A részletes számla szintén nagy segítséget jelenthet. Amikor felhívod a telefonos támogatást, a segítség igénybevétele után részletesen megtudhatod, hogy a neked segítő személy pontosan mit csinált, mennyi ideig tartott, és pontosan mit is javított ki.

A kisebb tanácsadó cégek és a független tanácsadók számos vállalat számára kiváló megoldást jelentenek. Ha 1–100 felhasználód van, alkalmazásuk lehet a legjobb megoldás. A tanácsadó biztosan kéznél lesz, ha valami hiba történik, és emiatt elő kell venni valakit. Sokszor helybéli, és nem egy esetben azt is tudjuk, hogy hol lakik. Sok olyan tanácsadó van, aki nagyobb tudással bír, mint a RedHatnél ülő átlagos mérnökök. A tanácsadókra jellemzőbb a közvetlenül gépekkel végzett munka, a saját próbalabor és az új dolgok kipróbálására való hajlandóság.

A tanácsadók e tulajdonságának megvan a maga jó és rossz oldala is. A jó oldala az, hogy a tanácsadó több linuxos programot és megoldást ismer(het). A rossz az, hogy a tanácsadó, ha egy új és jobb megoldást talál, hajlamos rá, hogy az első megbízójával fizettetett telepítési költségbe beleszámítja a saját tanulási költségeit is.

Azt azért hangsúlyozni kell, hogy általában nem ez a helyzet. Az üzleti életben az a jellemző, hogy nem kell megfizetni azt, hogy valaki megtanulja, miként nyújthassa számunkra az adott szolgáltatást.

A gyakorlatban az is előfordul, hogy a felhasználó olyan módszerre tart igényt, amit a tanácsadó még nem ismer. A vállalat ezután vagy kifizeti a tanácsadónak a telepítés elsajátítását, vagy keres egy másik tanácsadót, aki ismeri az adott megoldást. A tanácsadó felelőssége, hogy beismerje, mennyire van tisztában az adott módszerrel.

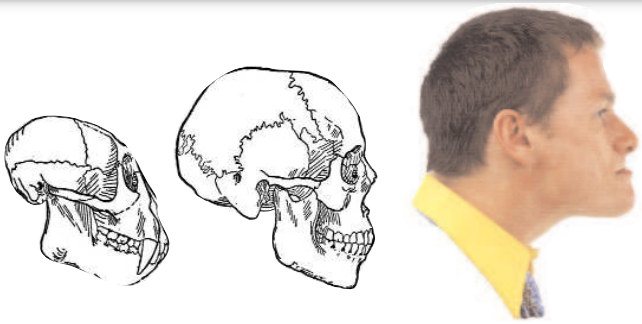
Mindezek ismeretében hogyan találhatunk tanácsadót? Ez bizony nehéz kérdés. A linuxos tanácsadók legnagyobb adatbázisa a ➲ http://www.linuxports.com címen található Linux Consultants Search and Guide. Jelenleg ezernél is több bejegyzés van benne, Kaliforniától egészen Törökországig, a jelentkezők között irányítószám, kulcsszó és egyéb adatok alapján is kereshetünk.

Egy másik lehetőség a környékbeli Linux-felhasználók közötti keresgélés. Sok helyen csoportok alakulnak (Linux User Group), ezek listáját a ➲ http://www.linux.com/lug/ címen megtalálhatjuk. Érdemes meglátogatni a Groups of Linux Users Everywheret is (GLUE) a ➲ http://www.ssc.com/ címen. A Linux felhasználói csoportok könnyen fellelhetők, és könnyen találhatunk köztük valakit, aki tud segíteni linuxos gépekkel kapcsolatban.

Ha Linux-tanácsadót keresünk, a legnehezebb az lesz, hogy jót találjunk. Számos tanácsadó rendkívül független, önálló. Tökéletesen önállók akarnak maradni, mindent a saját elgondolásuk szerint intéznek, és amíg nem beszéljük a nyelvüket, elég nehéz együttműködni velük. Nehéz lesz olyan tanácsadót találni, akivel „értelmesen” lehet beszélni. Ha sikerül ilyenre bukkanni, máris előrébb jutottunk. Megvan az oka, hogy sok nagy tanácsadó cégnek – akár Linux, akár más területen – saját fogyasztói részlege van, vagy legalábbis külön emberei a hívások fogadására.

A megvalósítást végző emberekkel sokszor nagyon nehéz szót érteni. Ha nem vagyunk járatosak a programozás és a számítógépek világában, még nehezebben értethetjük meg magunkat velük. Azt, hogy jó tanácsadót találtunk-e, nehéz eldönteni, de néhány dologra érdemes odafigyelni.

- A jó tanácsadó türelmes, mégsem lekezelő. Sokat beszélget velünk. Minden elvégzett és elvégzendő lépést teljes részletességgel elmagyaráz. Ha kell, ezt többször is hajlandó megismételni.



- Mindent a kezünkbe ad írásban is. Akár egy egészen kicsi, akár egy nagyobb tervezet esetén vázlatot készít. A tervezet vázlata lehet egészen egyszerű, egyes esetekben akár féloldalas is, de több tíz oldalt is kitehet. A lényeg az, hogy mindent rögzítsünk. A telefonos hívások alkalmával a tanácsadók a hívás során egyre részletesebb párbeszédet kívánnak. Segítenek dönteni az ügyfélnek, hogy milyen lépéseket kell megtennie. Amikor a hívás véget ért, írásos elemzést nyújtanak az elvégzett feladatokról és arról, hogyan hárították el a hibát. Az elemzést a számlázáskor is megismétlik.
- Minden elvégzett munkával elszámolnak, és azt fogják mondani: „Úgy becsülöm, két óra alatt végezni fogok ezzel a munkával." Az ügyfélnek viszont meg kell értenie, hogy a számítógépek viselkedése kiszámíthatatlan, a tanácsadók többsége pedig idő- és eszközalapon számláz. A munka a becsültnél kevesebb vagy több ideig is tarthat. A tanácsadó felelőssége, hogy ezt megértesse az ügyféllel.
- A jó tanácsadó készséggel fogja átnyújtani hivatkozási listáját, mely tudásáról és etikus viselkedéséről tanúskodik. Fontos megjegyezni: a Linux új rendszer, ennél fogva számos kezdő tanácsadó próbál megélni belőle. Mégse féljünk a kezdő tanácsadóval végzett munkától. Ha kétségeink merülnének fel tudásukkal kapcsolatban, alkudjunk az árból. Kérdezzük meg, ki tud-e jönni és felméri-e a hibát ingyenesen. Mondjuk el, hogy ha megfelelő megoldást tud nyújtani a hibára és el tudja hárítani, teljes fizetséget ajánlunk természetesen a hiba megállapításának idejére is. Ezáltal módunk nyílik modora megismerésére, felmérhetjük, vajon milyen teljesítményt fog nyújtani, és általános értelemben véve is kialakulhat bennünk a megérzés, hogy vajon együtt tudunk-e dolgozni vele.
- A jó tanácsadó nem csak Linuxszal foglalkozik. Számos olyan tanácsadó van, aki azt hangoztatja, hogy csak Linuxszal foglalkozik. Azt hiszik, hogy így legalább műszaki szempontból jobb tanácsadónak tűnnek. A baj az, hogy – hacsak nem kizárólag linuxos gépeket használunk – számos gyártó különböző termékeivel kell foglalkozni. Van olyan ügyfelem, akinek WTS (Windows NT Terminal Server) 4.0/Citrix, AIX, Linux és SCO rendszert futtató gépe is van. Mindegyikkel foglalkoznom kell, és a hozzájuk csatlakozó felhasználókkal is. Ha csak a WTS-hez értenék, nem tudnék átfogó szolgáltatást nyújtani. (Ettől függetlenül minden lehetséges alkalommal támogatom az áttérést Linuxra.) Nagyon fontos, hogy a tanácsadó megértse munkakörnyezetünket. Nem várhatjuk el tőle, hogy minden területen szakértő legyen, de munkakörnyezetünk alapvető ismerete nagyon fontos. Ugyancsak elengedhetetlen, hogy az alapvető rendszergazdai műveleteket más operációs rendszereken is el tudja végezni.

Másrészről pedig a Windows NT hálózatunkat fenntartó személytől ne várjuk el, hogy a Linuxhoz is értsen. A Linux a Windows NT-től merőben eltérő operációs rendszer. Teljesen máshogy működik, és munkára fogni is eltérő módon lehet. A Linux évekig gond nélkül fog működni, ha hozzáértő személy végzi a karbantartását, ha azonban nem találunk ilyen embert, folyamatosan csak bosszúságot fog okozni. Ha azt tervezzük, hogy áttérünk Linuxra, és szeretnénk megtartani meglévő tanácsadóinkat, győződjünk meg arról, hogy naprakész a linuxos tudásuk. Ha ezek jellemzik a jó tanácsadót, feltehetjük a kérdést: „Vajon a mi tanácsadónk miért nem ilyen?"

A jó tanácsadónak is megvannak azonban a maga hátrányai. A valóban jó tanácsadó, ha életben akar maradni és követi a leírtakat, úgy fog számlázni nekünk, mint egy ügyvéd. Ha felhívom az ügyvédemet, akármilyen egyszerű kérdéssel is, azonnal kiszámlázza a hívás díját. Ha a cégem nyereséges akar maradni, akkor ki kell számláznia az időmet. Lehet, hogy részemről csak tíz percet igényel a hívás, de akár 30-45 percig is eltarthat a rögzítése, átadása az ügyintézőnek, aki rögzíti a hívást, majd a későbbi hivatkozásokhoz iktatja a jegyzeteimmel együtt. Ha naponta tíz ilyen hívás érkezik, az azt jelenti, hogy három órát kell kiszámlázni.

Mint már említettem, a tanácsadó az ügyvédhez hasonlóan idő- és eszközalapon végzi tevékenységét. Ha a leírt szintű szolgáltatást szeretné nyújtani a felhasználó számára, kénytelen lesz az átlagosnál többet számlázni. A feljegyzések elkészítése időt igényel, ezért meg kell fizetni. Sőt lehet, hogy a megbeszéléseket, az egyeztetéseket, vagy egyszerűen csak magát a törődést is meg kell fizetni. Az ügyfélnek lehet mindig igaza, de az már nem előnyös számára, ha a tanácsadó emiatt nem tudja elvégezni munkáját. Minél több idejét vesszük el a tanácsadónak, annál többe fog kerülni. A tanácsadót nem szabad lépésről lépésre vezetni, hanem bizalmi viszonyt kell kiépíteni vele. Arról azonban meg kell győződni, hogy tartja-e magát a megbeszéltekhez. Ehhez természetesen tapintatos viselkedés szükséges.

Mindezt figyelembe véve, hogyan különböztethetjük meg a jó és rossz tanácsadókat? A jó tanácsadó sokkal több, mint egy műszaki szakember. Rossz tanácsadóból is sokféle van, és szerintem ezek közül nem mindegyik nevezhető egyértelműen rossznak. Én személy szerint nem kedvelem a „menedzsmentet". Azokra az emberekre gondolok, akik minden reggel nyakkendőben jelennek meg, mindenhez értenek és a PC Worldöt olvassák. Az agyamra mennek. Ettől én rossz tanácsadó lennék? Nem hinném, ha megfelelően tudok tárgyalni, és elég türelmes vagyok azzal a nyakkendős fickóval, aki épp most tette le a PC World legutóbbi számát.

Ismétlem, a legtöbb tanácsadó nem rossz. A nézeteltérések túlnyomó része abból származik, hogy a tanácsadók és a cégek nem tudnak megfelelő párbeszédet folytatni egymással. Ha nem rendelkezünk megfelelő kapcsolattartó képességekkel, mindig nehézségeink lesznek. Az ügyfél és tanácsadó közötti párbeszéd két formában valósul meg: az ügyfélkezelésben és a tanácsadó-kezelésben.

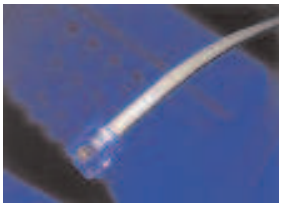
A jó tanácsadó jó ügyfélkezelési képességekkel bír. Az ügyfélkezelés az a képesség, hogy a felhasználóval annak megfélemlítése, zaklatása vagy idegesítése nélkül tudjunk párbeszédet folytatni. Az a képesség, hogy a kéréseket anélkül tudjuk megoldani, hogy közben védekező állásba merevednénk. Az a képesség, hogy egyenlő félként tudjuk kezelni az ügyfelet, ugyanakkor meg tudjuk szabni azt a határt, ameddig együtt dolgozhat velünk. A leggyakoribb hiba, amit tanácsadók elkövetnek, hogy munkaidőn kívüli hívást is rendes áron számláznak ki. Vannak, akik ezt fogyasztói támogatásnak nevezik, de a fogyasztók – természetükből és üzleti érzékükből fakadóan, tudniillik minél kevesebb pénzt próbálnak költeni – kihasználják ezt. Ha egyszer megtesszük, sajnálatos példát teremtünk.

A másik oldalról a jó ügyfél is képes a tanácsadók kezelésére. A tanácsadónak pontosan meg kell értenie, mi a szerepe, és mit vár el tőle a felhasználó. Ha a tanácsadó nem érti meg ezeket a dolgokat, folyamatosan „de én úgy gondoltam, hogy…" és „várjon csak, nekem azt mondta, hogy…" kezdetű szövegekbe fogunk ütközni. Ha ellentmondásmentes eljárásokat használunk, ha pontosan, tisztán és szabatosan fejezzük ki mondanivalónkat, a tanácsadás során nem lesznek nézeteltérések.

Joshua Drake
e-kereskedelmmel és Linux tanácsadással foglalkozik.
A Linuxot a kezdetektől fogva használja.
A Linux Documentation tervezet webmestere.

Az eTransMan segít a bajban

Vállalkozásszintű egészségügyi alkalmazások



Napjainkban a legtöbb amerikai egészségügyi cég újabb és újabb módokon próbálja emelni szolgáltatásainak színvonalát, a költségek csökkentését is szem előtt tartva. Az előálló gondok megoldását egyre többen a szakmai beruházásokban látják. Cégünk – mely 1300 dolgozót foglalkoztat az Egyesült Államok 13 államának negyven hivatalában – szintén erre az útra lépett, hiszen a jelenlegi betegellátási és gazdasági nyilvántartó rendszerünk tíz évvel ezelőtti fejlesztés eredménye, s mint ilyen, képtelen megfelelni folyamatosan bővülő szervezetünk igényeinek.

Munkacsoportunk számos tagja jelentős tapasztalattal bír a magas hozzáférésű rendszerek nagy adatáramainak kezelésében, e szakmai háttérrel felvértezve a kiépítendő rendszerrel szemben a következő kívánalmakat fogalmaztuk meg:

- Legyen méretezhető, hordozható, és megbízható mind szakmai, mind adatvédelmi szempontból.
- Teljesítse a HIPAA – az egészségbiztosítás szakmai megvalósítását szabályozó törvény az Egyesült Államokban – követelményeit.
- Tegye hozzáférhetővé minden alkalmazásunkat, valamint a cégek közötti kereskedelmi kapcsolatban is tudjuk használni.
- Elérhető legyen bármikor, bárhol és bármely készülékről.
- Legyen kifizetődő.

A vállalkozásszintű alkalmazás

Vannak vállalatok, ahol az alkalmazások igényeire a választ újabb és újabb eszközök beszerzése jelenti. Hisz ez manapság már nem is drága – mondogatják a forgalmazók. Esetünkben azonban nincs mód arra, hogy egyetlen bájtot, processzorciklust, vagy lemezblokkot is feláldozzunk a túlméretezett vagy kevéssé hatékony alkalmazások kedvéért, így rendszerünk méretezhetősége létfontosságú.

A hordozhatóság és a méretezhetőség között szoros kapcsolat áll fenn. Jó példa erre egy Intel-felületen készült alkalmazás, melyet a megfelelő teljesítményű működéshez nagygépen kell futtatnunk. Ilyenkor a programnak szükségszerűen más operációs rendszer és gépfelépítés mellett is működnie kell. A rendszerek alkatrész-összeállítása akár egy éven belül is megváltozhat, amennyiben ezt a sebességigény, a költségek csökkentése, a megbízhatóság vagy a személyi körülmények indokolják – alkalmazásainkat azonban nem cserélhetjük ilyen gyakorisággal.

A HIPAA törvénycsomagot – mely komoly irányelveket tartalmaz az egészségügyi szervezetekre nézve – az Egyesült Államok kongresszusa 2000-ben hagyta jóvá, hatálybalépése pedig 2002-ben esedékes. A jogalkotók szándéka az igények feldolgozásának szabványosítása, valamint a cégek közti adatforgalom és a betegek nyilvántartásának szabályozása volt. Rendszerünknek meg kell felelnie a törvény előírásainak, valamint követnie kell a szabályozókörnyezet változásait. Ipari elemzők szerint az egészségügyi szervezetek a HIPAA előírásainak követésére két-háromszor akkora összeget költenek, mint az Y2K kezelésére. Ugyanezen elemzők munkáiból kiderül azonban az is, hogy az egészségügy lemaradt a korszerű adatátvitel alkalmazásában. Sok cég számítógépes rendszerei felett eljárt az idő. Emellett a leendő rendszernek a régi programokkal is tartania kell a kapcsolatot. Ha ugyanis nyilvántartásba veszünk egy beteget, azonnal be kell jegyeznünk a gyógyszerszolgáltatónknál is, hogy a kapott receptet még aznap kiválthassa, bárhol tartózkodjon is az Egyesült Államok területén. Mindemellett a gyógyászati segédeszközök forgalmazójához is adatokat kell küldenünk, hogy új betegünk mihamarabb hozzájuthasson a megfelelő eszközhöz.

Szemle az alkalmazások világában

Cégünk több mint ötven jelenleg használt egészségügyi alkalmazást vett tüzetes vizsgálat alá – azonban egyikük sem bizonyult megfelelőnek. Sok vállalat dollármilliókat költött csak arra, hogy DOS-ról 32 bites windowsos kiszolgálóalapú rendszerre térjen át. Mások megmaradtak a DOS-alapú alkalmazásoknál, és meglehetősen furán néztek ránk, mikor nagy nehezen elmagyaráztuk, mit várnánk el a rendszertől. Végül is, a kész alkalmazások között nem találtunk használhatót, ezért úgy döntöttünk, hogy összeállítunk egy fejlesztői csoportot, melynek tagjai jelentős tapasztalatokkal bírnak a nagyméretű tranzakciók kezelése terén. Ők készítik el az alaprendszert, majd az erre épülő alkalmazások sorát.

A rendszer felépítése

Első feladatunk a rendszer felépítésének körülírása volt. Többrétegű felépítést szerettünk volna megvalósítani, azonban emellett a lehetséges megoldások számát sem kívántuk korlátozni. A piacon található számos tranzakciómotor és alkalmazáskiszolgáló elemzése nem vezetett kedvező eredményre – igényeinknek egyikük sem felelt meg. Némelyikük túlzottan drága volt, az olcsóbbak pedig vagy túl rugalmatlannak, vagy túl megbízhatatlannak bizonyultak.

Mit értünk mi megbízhatóságon? Nos, egy egyszerű elektronikus kereskedés esetében egyezreléknyi szolgáltatáskiesés – ez évente nagyjából kilenc órát tesz ki – legfeljebb néhány megrendelés elvesztését jelentheti. Az egészségügyben azonban, egyszerűen elengedhetetlen, hogy ha egy nővér a betege adatait akarja letölteni, elérhesse a szükséges adatokat.

Megoldás – az eTransMan

Nos, minthogy más lehetőségünk nem maradt, magunk kezdtünk bele a szteroidok tranzakcióit lebonyolító motor építésébe – az új jövevény az eTransMan nevet kapta a keresztségben (teljes nevén eCommerce Transaction Manager, vagyis elektronikus kereskedelmi tranzakciókezelő).

Az eTransMan egyszerűen használható környezetet teremt az egészségügyben használatos átfogó, többrétegű programok fejlesztéséhez és fenntartásához. Így a fejlesztők programjaikat modulonként készíthetik el bármilyen ismert nyelven, legyen az C, C++, Java, COBOL, Perl, Visual Basic, vagy akár assembly. Ezeket a modulokat azután az eTransMan egy nagyon jól

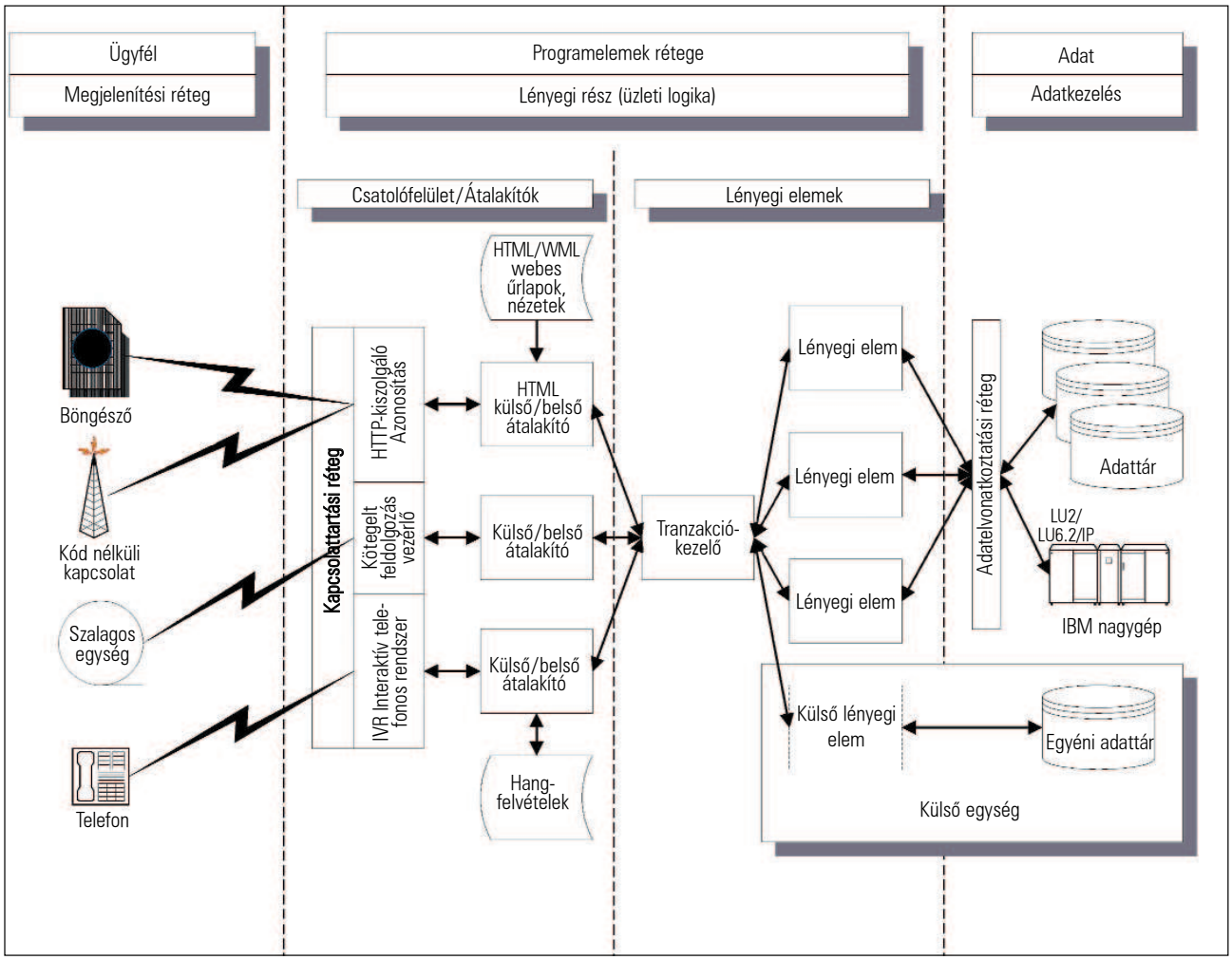
méretezhető, hordozható, nagyteljesítményű és biztonságos környezetben futtatja.
Az elemek fejlesztése az eTransMan felépítéséből következően egyszerű (lásd *ábra*). Fejlesztőink a lényegi elemeket C++-ban és Javában írják, másutt azonban – ahol nem áll rendelkezésre ilyen képzett csapat – előfordulhat, hogy csak a COBOL, a Visual Basic, vagy a PowerBuilder alkalmazható. Számukra az eTransMan már azért is nagy segítséget jelent, mert így az új rendszer megismerése mellett nem kell újabb programnyelv használatát is elsajátítaniuk.

Az eTransMan felépítése

Az eTransMan szerkezetének gerince a Transaction Manager (tranzakciókezelő), mely a hibatűrést és a teljesítménykiegyensúlyozást, valamint a feladatok ütemezését, a biztonsági szolgáltatásokat és a lényegi (az üzleti logikát megvalósító) elemek folyamatos nyilvántartását nyújtja.
Az alkalmazás felépítése követi a hagyományos többrétegű modellt, tartalmaz azonban néhány kiegészítést is. Szerkezetében különválnak a megjelenítés, a lényegi kód és az adatelérés szintjei. Jelen felépítés választását a következő szempontok indokolták:

- a tranzakciókezelőtől nagy átviteli teljesítményt vártunk viszonylag szerény gépteljesítmény mellett;
- olyan moduláris rendszert kívántunk létrehozni, mely hatékony, megbízható és könnyen kezelhető a mindennapi munka során;
- lehetővé kívántuk tenni az elemek fejlesztői számára, hogy a felépítés felületes ismeretével is könnyen és gyorsan írhassanak programokat e környezet számára.

A tranzakciókezelő linuxos rendszeren C/C++ nyelven készült, fordításához pedig a nyílt forrású gcc fordítót használtuk. Így a program könnyen átvihetővé vált bármely rendszerre, mely támogatja a gcc-t a beágyazott gépektől egészen az IBM 390-es nagygépig (ez utóbbi esetben a kitűnő felépítésnek köszönhetően alkalmazásunk teljesítménye meghaladta a várt szintet). Jelenleg az eTransMant két, kétprocesszoros Pentium III 700-as kiszolgálón futtatjuk.
Negyven helyen 450 felhasználó elérését teszi lehetővé, a működtetéshez szükséges alkatrészek összköltsége pedig nem érte el a 30 000 dollárt. A program csak szükség esetén foglalja le az erőforrásokat, így lehetővé teszi a gyors választ szerényebb rendszerek esetében is. A futtatás során az átlagos terhelés nem haladta meg a két százalékot. Mindezek mellett az eTransMan lefordított program, így nincs szüksége értelmezőre a futtatáshoz.
Számos lehetőséggel támogatja a magas hozzáférhetőséget, így képes az elérési és időtúllépési listák feldolgozására, valamint az elemek önműködő újraindítására, és az eredeti állapotok – önműködő vagy felhasználói közreműködéssel zajló – visszaállítására. A lényegi elemek démonként működnek, így lehetővé teszik az Apache webkiszolgáló gyors válaszát. A program azonban a főalkalmazás adatáramlásának kezelése mellett az elemek gördülékeny és hatékony működését is megvalósítja.
A hatékony adatátvitelhez az eTransMan több azonos elem között osztja el a terhe-



Az eTransMan szerkezete

lést, és folyamatosan számon tartja, mely elem érhető el adott tranzakció végrehajtásához. A növekvő átviteli igény kezelésére dinamikusan létrehozott elempéldányokat állít munkába, melyeket feladatuk végrehajtása után megsemmisít. Mindezek lehetővé teszik az erőforrások legjobb kihasználását, kielégítve ezzel a napjainkban felmerülő igényeket. A linuxos nyílt kód segítségével az eTransMan képes a kérelmeket dinamikusan hozzárendelni az elemekhez, így az

A fájlkiszolgálók

alkalmazások folyamatainak kezelése igen rugalmassá válik.

Az ügyfelek és az alkalmazások közti kapcsolatok hatékony, kétirányú kezelésével az eTransMan megteremti a kritikus alkalmazások működésének feltételeit, például a nagymennyiségű központilag vagy szétszórtan tárolt adatok kezelésének lehetőségét, a nagy igényeknek megfelelő adatátvitelt, az adatok épségét és szavatolja a biztonságot, akár folyamatos szolgáltatás mellett is.

Az erőforrások felhasználását a beállítási fájlok szabályozzák, előbbiek igénybevételére azonban csak égető szükség esetén kerül sor, így a rendszer gyengébb gépen is gyors válaszra képes. A beállítási fájlok kezelésére egy webes felületet is létrehoztunk.

Az eTransMan felépítése a többrétegű modellt követi. Az adatáramlás a következő módon zajlik:

- Értelmes adatügyfelek begyűjtik a feldolgozáshoz és megjelenítéshez szükséges adatokat. Ezek az ügyfelek lehetnek Linux, Windows 2000/98/95, vagy Mac munkaállomások, drótnélküli készülékek, interaktív hangfelismerő rendszerek, a Unix csaknem minden kereskedelmi változata, kötegelt bevitelek, számítógépalapú pénztárgépek, video-adatfolyamok vagy más, a kívánt adatok bevitelére alkalmas eszközök.
- Az ügyfelenként különböző átalakítók előkészítik az adatokat az eTransMan számára, majd elküldik az adatügyfél kérelmét a programnak. Így az új ügyfelek beépítése igen egyszerű, hiszen csak egy olyan fordítót kell készítenünk, mely képes az adatokat átalakítani az eTransMan közös formátumára és viszont. A programok lényegi részét tehát nem kell átírnunk.
- Az eTransMan ellenőrzi a kérelmet, majd az adatokkal együtt továbbküldi a megfelelő elemnek.
- Az elemek adatbázis-elérésére számos módszer ismeretes – csoportunk a költségek csökkentésére és a hatékonyság növelésére saját adatbázis-elérő elemet fejlesztett ki, mely egy Linux alatt futó Oracle 8i adatbázissal áll kapcsolatban. Külső adatbázisokkal saját könyvtáraik segítségével, az ODBC, a JBDC, vagy nagygépen futó HLLAPI elemmel tudunk érintkezni. A lényegi programelemek visszatérnek a válasszal és a kapott adatokkal az eTransManhez, mely továbbküldi azokat az értelmezőhöz – ez a továbbiakban saját szabályai szerint alakítja át az adat formátumát.
- Az értelmező megkeresi a válaszhoz és az adatokhoz a megfelelő sablont, majd az adatokat egyesítve a sablonnal visszaküldi az ügyfélnek az ott megkövetelt formátumban. A sablonok képesek a különálló és a táblázatos adatokat egyazon válaszban kezelni.

A megjelenítési réteg

Ahogy a tranzakciókezelő választ kap az elemektől, továbbítja a kérelmet leadó kapcsolati réteghez. Ez olyan formátumra hozza az eredményt, melyet az ügyfél képes megérteni. Így az ügyfél az adatokat a saját formátumában kapja meg.

A megjelenítési réteg tervezésekor mindenekelőtt arra kellett odafigyelnünk, hogy a felületet minél egyszerűbben kezelhetővé tegyük. Erőfeszítésünk első igazi megmérettetése az volt, amikor meglévő PC-alapú böngészőnket vezeték nélküli Palm Pilot gépekre és mobiltelefonokra írtuk át.

Azt szerettük volna elérni, hogy a használat során a felhasználók bármilyen egyszerű eszköz – esetünkben mobiltelefon – segítségével hozzáférhessenek az adatbázisok tartalmához. A rendszert a gyakorlatban elsőként egy beteg hazavezetésével próbáltuk ki. Böngészőt futtató mobiltelefonon, illetve vezeték nélküli Palm Pilot gépen kapta kézhez az útmutatásokat. Fordítási módszerünk segítségével lehetővé vált, hogy webes alkalmazásokat (jelen esetben egy 150 oszlopos jelentést) egy új sablon segítségével a vezeték nélküli készülékek világában is felhasználjunk. Az átírás

Vezetési útmutatás vezeték nélküli rendszeren

WML felületre kevesebb mint négy órát vett igénybe, és nem volt szükség a lényegi és az adatbáziselemek átírására. Hasonlóképpen jártunk el a Palm VIIx esetében is. Ez szintén négy órát vett igénybe, főleg azért mert nem ismertük a használni kívánt WML rendszert. A Yankee Group elemzői szerint az egészségügyben a vezeték nélküli rendszerek 12 millió leendő felhasználóra számíthatnak, ami – jóllehet manapság még a világháló korszakát éljük – a vezeték nélküli készülékek uralmát vetíti előre. Mindezek miatt rövidlátó terv lenne olyan rendszert alkotni, mely a vezeték nélküli eszközök szélesebb körű elterjedése esetén átírásra szorul.

A kapcsolati réteg

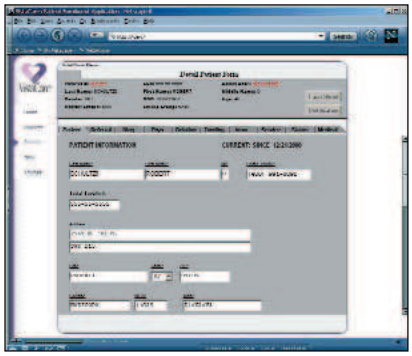
E réteget a felületek és a fordítók alkotják, melyek az adatokat a lényegi elemek felé közvetítik, emellett ezek formázzák az eredeti ügyfeleknek szánt adatcsomagokat is. Ez utóbbi történhet sablonok segítségével, így a HTML adhatja például a hagyományos részletes jelentés formátumát.

A felületek és a fordítók lehetővé teszik, hogy a fejlesztők csak a lényegi programokkal foglalkozzanak, nem kell törődniük azzal, hogy milyen formátumban történik az adatok továbbítása. A fordítók így a lényegi elemektől közös formátumban kapják az adatokat, és az eredményt szinte bárhová továbbíthatják – a világhálóra, interaktív hangfelismerő rendszerekre, vezeték nélküli eszközökre, vagy akár nyomtatókra is.

Ennél is fontosabb azonban az, hogy az elemek így újra felhasználhatók maradnak akkor is, ha az adatügyfél megváltozik. Következésképpen, mivel így sikerült a megjelenítési és a kapcsolati elemeket elválasztani a lényegi kódtól, a fejlesztés is gyorsabbá válik. Másrészről ez a rendszer lehetővé teszi azt is, hogy a fejlesztőknek – és itt most leginkább a webtervezőkre és a grafikusokra gondolunk – a megjelenítési réteg megvalósításánál ne kelljen foglalkozniuk azzal, mi történik a lényegi elemekben vagy az adatbázisok területén.

Az eTransMan lényegi elemeinek modellje

Az előzőekben már többször említettük, hogy az eTransMan felépítése programelemeken alapul. Az adatügyfelek és mások kérelmei lényegi elemeket hívnak életre, a fejlesztők pedig ezek programozását végzik, illetve az itt zajló folyamatoknak megfelelően módosítják saját elemeiket. Ez a rendszer lehetővé teszi az elemek gyors fejlesztését, és ehhez a programozóknak nem kell feleslegesen sokat tudniuk a teljes rendszerről.



Egy beteg adatlapja

etransman_manage

Az általunk készített alkalmazásban szükség volt arra, hogy az egyes lényegi elemek elérhetőségét a többitől függetlenül módosíthassuk. Ezzel a gonddal is magunknak kellett megbirkóznunk, ugyanis az általunk vizsgált kereskedelmi és nyílt forrású tranzakciós felületek közül soknál nem találtunk ilyen lehetőséget.
Az egészségügyi alkalmazások területén különleges éberséget kell tanúsítanunk a rendszerfelügyelet terén, a nehézségeket még az is tetézi, hogy a teljesítmény- és biztonsági felügyelet, a hiba- és riasztáskezelés, valamint számos hasonló szervezési gond fokozott nehézséggel jelentkezik szétszórt rendszerek esetében. Mindezen feladatok megoldása mellett webes felügyeleti lehetőségeket is beépítettünk, hogy a rendszerrel elektronikus kereskedelmi alkalmazásokat is lehessen fejleszteni. E lehetőségek megvalósításának legkézenfekvőbb módjaként az kínálkozott, hogy a felületi és fordítóréteg működését a Weben keresztül oldjuk meg – így végül az eTransMant alkalmazhattuk az eTransMan felügyeletére.

A adatelvonatkoztatási réteg

Ez a réteg adja az adatbázis-kezelés közös felületét – segítségével a fejlesztők úgy létesíthetnek kapcsolatot különböző adatbázisokkal, hogy ehhez nem kell módosítaniuk a lényegi elemeket.
Fejlesztőinknek tehát nem kell tudomást szerezniük arról, ha a mélyben valamilyen adatszerkezeti váltás történt (például áttértünk egy IBM nagygépes DB2-ről egy Linux alatt futó Oracle-re), és kódjukat nem kell különböző adatbázisok egyedi tulajdonságaihoz igazítaniuk (így egy elemen belül nincs szükségünk SQL, vagy HLLAPI hivatkozások használatára). Mindezt az teszi lehetővé, hogy egy erre a célra készült elem az általános kérelmeket lefordítja a kívánt adatforrás nyelvére.
Más felületektől eltérően, melyek a JBDC, vagy az ODBC segítségével érintkeznek a megfelelő adatbázisokkal, az eTransMan képes kihasználni az adatbázisok saját könyvtárait. Ezzel a módszerrel lehetőség nyílik a gyors adatbázis-elérésre csekély fenntartási költségek mellett.
Az elemektől az adatbázisok eljárásaihoz érkező hívásokat egy metaadat-réteg vezérli, melyet az elempéldányosításkor hoz létre. Így végül elértük, hogy minden réteg azt tegye, amihez a „legjobban ért”, vagyis az adatbázisok írják és olvassák az adatokat, a főprogram pedig a lényegi elemeket futtatja. Az előbbiek mellett ez a réteges szerkezet azt is lehetővé tette számunkra, hogy az egyik adatkezelő rendszer alatt írt programot egy másikra gond nélkül átvigyünk.

Adatbázis-kapcsolatok készletezése

Ez a módszer a lehető legkevesebbre csökkenti az adatbázissal létesített kapcsolatok számát. A kapcsolatok létesítése ugyanis fokozottan igénybe veszik az adatbázisokat és a rendszer erőforrásait, ráadásul rendszeres felépítésük és lebontásuk általános teljesítménycsökkenéshez vezet. A nagymennyiségű tranzakciót lebonyolító környezetekben a kapcsolatkészletezés biztosítja az erőforrások hatékony kihasználását, emellett csökkenti a válaszidőt és növeli az adatátvitel sebességét.
Napjaink számos adatbázis fejlesztője az egyszerre kiszolgálható felhasználók, illetve egyszerre fenntartható kapcsolatok száma, vagy az adatbázist működtető processzorok összteljesítménye alapján sorolja be termékét. Az eTransMan tranzakciós felület nem ró további jelentős költségeket a fenntartóra, hiszen a rendszer meglévő erőforrásaival gazdálkodva képes az elérhető legjobb teljesítményre.
Ha átlagos napi terhelésünk alapján kétszáz egymás melletti kapcsolatra számítunk, ezek kezelésére hagyományos körülmények között meglehetősen gyors gépre lenne szükség, jelentős mennyiségű memóriával. Kétszáz felhasználónkat azonban könnyedén kiszolgálhatjuk öt készletezett adatbázis-kapcsolattal, lehetővé téve, hogy az eTransMan szükség esetén újakat is kialakítson.

Összefoglalás

Rendszerünk elkészült, és jelenleg sokat dolgozik. A siker titka pedig az, hogy a rendszer minden alkotórésze apró, újraindítható és több példányban létező elemek között oszlik el, így adott folyamat lebonyolításához általában több járható út is elérhető. A lényegi elemek – amennyiben több példány használata egy helyben nem megfelelő megoldás – több különböző kiszolgálón is futhatnak, így biztosítva a folyamatos működést akár valamely kiszolgáló teljes kiiktatása esetén is.
Az eTransMan ilyenkor a működésképtelen kiszolgálón található elemeket elérhetetlennek nyilvánítja, és a folyamatokat másfelé tereli. A webkiszolgálók esetében ez a módszer nem újdonság, azonban mostantól az alkalmazások is élvezhetik az előnyeit.
Az alkalmazásokkal szemben támasztott követelmények gyorsan változnak, ahogy a cégek életében fordulatok állnak be – semmit sem tekinthetünk állandónak. Sőt, biztos, hogy a jövőben más lényegi és adatbáziselemeket fogunk használni, mint ma. A megfelelő rendszerfelépítéssel lényegi elemeinket könnyen más környezetbe helyezhetjük – így tettünk akkor is, amikor a webes környezetről a vezeték nélküli világba léptünk át.
Áttekintésünk a szakmai részleteken túl körvonalaz egy fontos tanácsot – és ez a cikk valódi tanulsága –, ne kötődjünk túl szorosan egyetlen gyártóhoz, módszerhez, felülethez, webkiszolgálóhoz vagy adatbázishoz sem!

Garry Bennett
(linuxrules@vista-care.com) számítástechnikai osztályvezető a VistaCare nevű egészségügyi szervezetnél. Több mint 17 év programfejlesztési tapasztalata van orvosi, közszolgálati és katonai területen. Szabadidejében a domborzati térképek rabja, és gyakorta tervez túrákat Arizonában és környékén.

A Transmeta viharos indulása

A webpadpiacok és hasonló területek feltárása lehet a Transmeta útja, ha többet szeretne, mint egyenesbe jönni.

A tavalyi, nagy nyilvánosság előtti bemutatkozás óta eltelt időszak igencsak zsúfolt volt a Transmeta számára. Megkezdték a cég első termékének forgalmazását, mely termék elnyert néhány jelentős tervezési díjat és teljesíti a piacra dobás feltételeit (IPO). Annak ellenére, hogy részvényeinek értéke azóta felére esett vissza, a Transmeta hárommilliárd dolláros értékével még mindig igen tekintélyes piaci erőt képvisel. A cég a vártnál keményebb harcra kényszerült az Intellel a PC piacon, és csak lassan tudott előre törni a webpadok területén is. Nemrégiben megnyitotta harmadik arcvonalát is, és megtette az első lépéseket a kiszolgálók piaca felé.

Indulásakor – mely arról is híres, hogy az alkalmazottak sorában megtalálhatjuk *Linus Torvaldsot* is – elkészítették a Crusoe-lapkát, mely Intel-megfelelő, viszont az Intel-processzoroknál sokkal kevesebbet fogyaszt. Ezt a mutatványt különleges kódátalakító (code-morphing) módszer segítségével képes véghezvinni, amely utánozza az Intel szerkezetét az egyszerűbb VLIW gépen. Kezdetben a Transmeta hitt abban, hogy az Intel nem lesz képes megközelíteni a Crusoe kis fogyasztását saját kódátalakító kifejlesztése nélkül. Arra számítottak, hogy ez éveket vesz majd igénybe, de csalódniuk kellett, mert az Intel hamar visszavágott.

Az Intel a kódátalakítás ismerete híján, lapkái energiafelhasználásának csökkentéséhez inkább a félelmetes gyártási módszereit állította csatasorba. Tavaly júniusban az Intel új processzorokat dobott piacra, melyek alacsonyabb feszültséggel működtek, mint elődei és a Transmeta IBM által készített lapkái. Az IBM birtokol ugyan az Inteléhez hasonló megoldásokat, de fenntartja belső termékeihez. Annak ellenére, hogy néhány tanulmány szerint a Crusoe még mindig az élen jár az energiatakarékosság terén, az Intel legújabb lapkáival szemben már kicsi az előnye.

A Transmeta használható teljesítményadatot sem tudott szolgáltatni a processzorához. Természetesen az igazsághoz az is hozzátartozik, hogy a kódátalakítás nem teszi éppen egyszerűvé a teljesítménymérést. A Crusoe-lapka ugyanis gyorsabbá válik, ha a programot többször futtatjuk. Az első Crusoe rendszerek teljesítménye nagyjából egy egyszerű 300 MHz Pentium III képességeivel egyezett meg. Azonban ezeket az ellenőrzőprogramokat csak egyszer futtatták le. Azt nem lehet tudni, hogy többszöri futtatás során milyen gyorsan működne a lapka, mivel a Transmeta hallgatásba burkolódzik.

A lapka teljesítményéről szóló rövid híradásokkal és a támadások kereszttüzében álló kis fogyasztásával, az elmúlt pár év során a Transmeta korlátozott mértékben tudott csak előretörni a hordozható gépek piacán. Ennek ellenére néhány nagy cég is beépítette a Crusoe-lapkát új, ultrakönnyű gépeibe. Ilyenek például a Sony, a Fujitsu, a Hitachi és a NEC. Ezek a rendszerek főként a japán piacot célozták, ahol a méret és az energiatakarékosság igen fontos tényező. Az amerikai PC-piac vezetői – az IBM, a Toshiba, a Compaq és a Dell – még nem hozakodtak elő semmilyen Crusoe-alapú rendszerrel, bár valamennyien számolnak a Transmeta termékeivel. Fentiek következtében a Transmeta részesedése az ultrakönnyű PC-piacon eddig megközelítőleg tíz százalék. Mivel ez csak igen kis része a teljes PC-piacnak, a Transmeta jóval kevesebb, mint egy százalékát birtokolja a teljes PC-processzorpiacnak. Így a cég nemigen jelent veszélyt az Intel bevételeire. Természetesen a Transmetának nincs is szüksége az Intel részesedésének túl nagy százalékára, ugyanis az ultrakönnyű PC-piacnak már megközelítőleg a fele már egyenesbe hozná a Transmetát.

A befektetők az egyenesbe kerülésnél azért többet szeretnének, hiszen a Transmeta több mint kétszázmillió dollárt veszített amióta megalakult és valószínűleg további ötvenmilliót fog veszíteni még akkor is, ha év végére eléri hőn áhított célját, a nyereségességet.

A befektetők azonban szerény megtérülésben is reménykednek. Ez az, ahol az új piacok érdekessé válnak. A Transmeta második célpiaca a webpadok területe. Ezek a hordozható eszközök vezeték nélküli kapcsolaton keresztül érik el a hálózatot, és lapos képernyővel vannak felszerelve.

A webpadok többnyire PC-khez használt processzorokat tartalmaznak, ezáltal lehetővé teszik a Weben található számtalan bővítmény futtatását. Ehhez azonban nem feltétlenül kell Windowst igénybe venni, sokan inkább Linuxot használnak. A jó hír az, hogy az egyedülálló PC-megfelelőségének, az energiatakarékosságának és a kedvező árának köszönhetően, gyakorlatilag minden webpad, ami eddig megjelent, Crusoe-t használ. Olyan cégek, mint a Gateway, a Phillips, a Hitachi és a Frontpath (korábban S3) választották webpadjaikhoz a Crusoe-t.

A rossz hír: a webpadok forgalma mind a mai napig csekély, mivel drágák, és ez erősen lassítja terjedésüket. A webpadok legdrágább része a lapos megjelenítő, amely egymagában is százezer forint körüli összegbe kerül. Mindamellett a PC-eladás jelenlegi lelassulása csökkentette a lapos megjelenítők iránti keresletet is, így az árak valószínűleg mérséklődni fognak. A gyártókapacitás folyamatos növekedésével együtt, ezek az árak jelentősen csökkenhetnek, így a webpadok 2001-ben egyre népszerűbbek lehetnek. A Transmeta idei jövedelmének kevesebb, mint húsz százalékát várja a webpadok területéről.

A Transmeta mostanában jelent meg harmadik célpiacán, a webkiszolgálóknál. Bár a Crusoe nem olyan erős, mint az Intel vagy más cégek kiszolgálóprocesszorai, a kis fogyasztású lapka nem igényel hatalmas hűtőrendszert. Ez lehetőséget ad a szállítóknak arra, hogy több Crusoe-lapkát építsenek be egyetlen energiaéhes Intel-lapka helyére. Mivel a honlapok kiszolgálásának feladatai könnyen elosztható több processzor között, a Crusoe a mennyiségen hozza be, amit az egyedi teljesítményen veszítene.

Ezidáig mindössze egyetlen kis cég támogatta a Transmeta kiszolgálókkal kapcsolatos elképzeléseit, így a Transmeta 2001-ben csekély bevételekre számíthat erről a területről. Azonban ez a gyorsan növekvő piac újabb lehetőséget teremt a Crusoe számára, még akkor is ha az Intel meg tudja állítani a Transmetát a PC-piacon. A Transmeta azt reméli, hogy a bevételek kitartanak addig, míg ezek az új piacok végre jövedelmezővé válnak. 2002-ben talán végre működés közben is megcsodálhatjuk a Transmeta teljes üzleti tervét.

Linley Gwennap (linleyg@linleygroup.com) a kaliforniai Mt. View-i székhelyű elemző cég, a The Linley Group ➲ http://www.linleygroup.com/, alapítója és vezető elemzője.

Többrendszeres Debian telepítés

Építsünk multimédiás munkaállomást, amely egyaránt képes Linux, Win98, WinME, WinNT vagy Win2K indítására a LILO segítségével.

Lépésről lépésre végigmegyünk egy olyan Debian Potato-telepítésen, mely után egymás mellett használhatjuk a Linuxot Windows 98/NT/2K operációs rendszerekkel.

A gépünk asztali rendszerként fog szolgálni egy programfejlesztő számára, akinek feladatai átmenetet képeznek a csak windowsos alkalmazások készítése és a Linux támogatása közt. Mielőtt végeznénk, látni fogjuk, hogy néhány dolog kissé félrevezető lehet a leírásban, ezért nem ajánlatos vakon követni a Debian utasításait vagy a Weben található linuxos HOGYAN-okat.

Gépem, mely előzőleg windowsos programfejlesztő rendszerként üzemelt, Abit BH6 alaplappal rendelkezik, Intel Celeron 300 MHz-es PC, 128 MB memóriával, Yamaha CD-íróval, SoundBlaster Live! hangkártyával, Haupaugge WinTV tévékártyával, 8 MB-os Hercules Terminator 2x/i videokártyával, valamint húszgigás merevlemezzel. Lehet, hogy nem látszik túl fényes gépnek, viszont az igényeimnek megfelel. Cégem szellemiségéhez az illik, hogy kerüljük a túlzott előretörést a gépek fejlesztési görbéjén. Olyan gépeken szeretnénk programokat fejleszteni, amelyek nem különböznek túlságosan a felhasználóink gépeitől. Így elkerülhetjük azt a kellemetlen meglepetést, hogy figyelmetlenségből olyan programot készítünk, ami csak és kizárólag a legújabb és legjobb PC-ken fut.

Nem akartam kockáztatni a lemezen található adatokat, és mivel amúgy is helyszűkében voltam, beépítettem a rendszerbe egy új 20 GB-os Maxtor DiamondMax Plus EIDE Ultra ATA/66 merevlemezt is. Ez a 7200 rpm-es Maxtor merevlemez lényegesen gyorsabb, mint a régi, 5400 rpm sebességű húsz gigabájtos merevlemezem. Szükségem is lesz erre a sebességre, például a videós munkáimhoz, ugyanis itt mára szűk keresztmetszetté vált az adatátvitel. Még az a szerencse, hogy lényegesen olcsóbbá váltak a nagy kapacitású gyors merevlemezek. Az új egység beépítéséhez le kellett szerelnem a tápot és ki kellett húznom az AGP kártyát a foglalatából.

A meghajtó Caldera DR-DOS telepítőlemezzel együtt érkezett, amely leírást, lemezfelosztó programot, meghajtótükröző programot tartalmazott, ezeket azonban nem használtam. Átraktam az IDE-kábelt a meglévő meghajtóról (ezzel egyúttal ideiglenesen kikapcsoltam a régit) az új merevlemezre, a BIOS-ban felismertettem a lemezt, meghagytam az alapértelmezett LBA beállítást, és már tovább is léptem. Minden eshetőségre felkészülve a telepítést ezen a teljesen új rendszeren végeztem. A régi meghajtót szándékosan húztam ki – a biztonság kedvéért.

A telepítést a Windowszal kezdtem. Ennek a változatai ugyanis nem túl barátságosak más, már a számítógépen található operációs rendszerekkel. Elkerülhetjük az összeütközést, ha elsőként a Windowst telepítjük és csak később a Linuxot. A Windows 98 SE ára körülbelül 27 000 forint, a Win2K-é pedig megközelítőleg 45 000 forint. Mivel a számítógépemet alkatrészenként állítottam össze, nem kaptam hozzá Windows-rendszert. A Windows 98-nak három nagyobb változata van: az eredeti, a Second Edition és a Millennium Edition. Én az SE és ME változatokat részesítem előnyben, mivel támogatják a többképernyős üzemmódot, és néhány egyéb ötletes eszközmegoldást is tartalmaznak. A Win98 SE-t CD-ről (kétszer) indítva telepítettem, és a FAT 16-os lemeztámogatást választottam (így legfeljebb 2 GB-os lemezterületet tudunk kezelni), valamint megadtam az int3 és 0x300 értékeket a nem-PNP ISA hálózati kártyámhoz, majd közöltem a tűzfal mögötti IP-címemet (192.168.1.2) és a hálózati maszkot (255.255.255.0). Ugyanazt a felhasználónevet és jelszót használtam a Win98 SE-ben és a Win2K-ban (nem a rendszergazdait), mivel így kevesebb nehézségbe ütköztem, amikor hálózati meghajtókat nyitottam meg Windows alatt.

A hálózati átjáró egy WinNT gépen futó WinGate, a 192.168.1.1-es címen. Ez teljesen fejletlennek tűnhet, hiszen a Linux elismerten jobb tűzfal, mint a WinNT, de a rendszer ilyen, hiszen kizárólag Windowst használó boltként kezdtük. Ez pont a fordítottja annak az általánosan elterjedt nézetnek, miszerint a Linuxnak kiszolgálónak, a Windowsnak pedig asztali gépnek kell lennie, s pontosan ez az, ami az átlagos Windows-felhasználót visszatartja attól, hogy a Linuxot asztali gépként is kipróbálja. Sikeresen elindítottam a Win98 SE-t a merevlemezről, majd a hálózaton keresztül rákapcsolódtam a letöltött Win98 meghajtók

A közismerten legtisztább Linux-változat, a Debian telepítése első ízben – mondjuk ki nyíltan – a rendszergazdák büntetése.

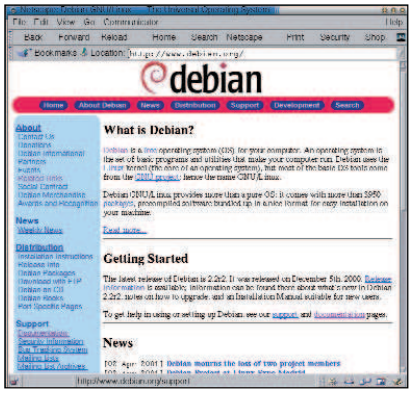
kincsestárára. Ha egy másik PC is elérhető a telepítés közben, az jelentősen felgyorsíthatja a folyamatot. Az egyik első dolog, amivel foglalkoztam, a helyes videomeghajtó telepítése volt, hogy kikerüljek a 640x480-as VGA felbontásból. Az első gonddal is itt szembesültem. A Win98 SE-nek újratervezett videoalrendszere van, amely megfojtotta Hercules Terminator 2x/i videokártyám Win98-as meghajtóját. A képernyő elsötétült. Újra kellett indítani a rendszert csökkentett módban. A feladat azonban nem is volt olyan egyszerű, mivel a csökkentett módot indító mágikus billentyű Win98 SE alatt a Ctrl és nem az F8, ahogy az a Win98-ban volt. Nem is bírtam elsőre kitalálni. A Windows nem jeleníti meg, milyen billentyűkombinációk használhatók rendszerindításkor. A tápellátás megszakítása azonban kikényszerítette a csökkentett módú indítást (ez azért nem követendő példa! – a szerk.). Feltelepítettem egy választható meghajtót a ➲ WinDrivers.com-ról. A Win98 további telepítése zökkenőmentesen zajlott.

A Windows 98-at telepítsük elsőként, mivel ez a legkevésbé elnéző más operációs rendszerekkel szemben. Hajlamos arra, hogy felülírja az elsődleges indítóterületet, az MBR-t. Következőként a Windows 2000 Professional rendszert telepítettem. A pro kiadás asztali változat, a Windows NT Workstation utóda. A név kicsit félrevezető, amatőr változata ugyanis nem létezik, másik választási lehetőségként a Server-változatok szerepelnek. A Win2K-t CD-ről indítottam és 2 GB-os NTFS lemezterületet készítettem neki. A Win2K telepítése hibátlanul végbement. Most egyaránt tudtam Win98 SE-t vagy Win2K-t indítani az ntldr windowsos rendszerindító program segítségével. Elővigyázatosságból a Win98 SE és Win2K rendszerhez is készítettem biztonsági indítólemezeket. Szerencsére nem volt szükség rájuk.

A Win98 SE lemezterületet azért készítettem FAT 16-os rendszerűre, mert a Linux és az MS-DOS is tudja kezelni. A FAT 32 rendszernek kevés előnye van, amennyiben a lemez kisebb, mint 2 GB. Emellett azért érdemes megfontolni, hogy FAT 32-t használjunk, ha nem kell régebbi Linuxokat vagy valamilyen DOS-t használnunk.

A Win2K-hoz több okból is az NTFS-t választottam, de a legfőbb indítékom az volt, hogy ha a rendszer összeomlik, elkerüljem az időrabló windowsos scandisket. NTFS alatt rendes körülmények között egy áramszünet sem okoz bajt. Előszeretettel telepítettem az operációs rendszereket külön 2 GB-os lemezterületekre, mivel úgy találtam, hogy ez megnöveli a megbízhatóságot Windows alatt.

Akkor most lássunk hozzá a Debian Linux telepítéséhez! A Debiant nem egyetlen ember vagy kis, zárt csoport fejlesztette. A Linux rendszermag hagyományait követi: azok fejlesztik, akik használni fogják, és ez jobb minőséghez, dinamikusabb, és valóban moduláris rendszerhez vezet (lásd a Linuxvilág 1999. novemberi számának 6. oldalán).



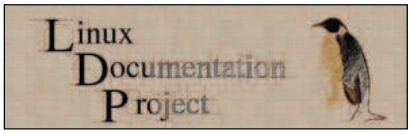
➲ http://www.debian.org

A Debian honlap ➲ http://www.debian.org/ szerint a Debian GNU/Linux több mint 3950 előre lefordított program csomagját tartalmazza a könnyű telepítéshez összeállítva. Meg kell jegyeznem, a Debian közismerten a legtisztább változat a szabad programok között és a nyílt forráskód szellemiségében. A szabad programok támogatóinak vélhetően tetszeni fog a Debian social contract című írás ➲ http://www.debian.org/social_contract/. A Debian esetében azonban nem beszélhetünk olyan háttércégről, mint például a RedHat.

Mondanom sem kell, hogy nem kaphatunk üzleti támogatást az olyan cégektől, mint például a LinuxCare. Ott van viszont a közösség támogatása, hiszen a Debian főként linuxos szakemberek és programozók tudásán alapul. A Debian felhasználók listáján elérhető támogatás valóban csodálatos. Azt tapasztaltam, hogy általában tíz percen belül helyes választ adnak a kérdéseimre. Ráadásul a csoport hangulata is üdítő. Létezik magyar nyelvű lista is, ezt a ➲ www.linux.hu alatt kereshetjük meg.

A Debian-kiadásoknak nevük van: a jelenlegi, 2.2-es kiadást Potatóra (krumpli) keresztelték, a következő kiadás pedig a Woody (fafej) névre hallgat. Eredetileg FTP-vel, hálózaton keresztül akartam letölteni a Potatót egy tükörkiszolgálóról, de nem volt kedvem éjszakára otthagyni a telepítést, inkább lemezekre vágytam. Beszereztem a teljes, hat CD-s összeállítást a Linux System Labstól (➲ http://www.lsl.com/) kilenc dollárért. Hihetetlenül olcsó a Windowshoz képest. Egyébként a megrendelésnél – ha akarjuk – öt dollárral támogathatjuk a Debian fejlesztőit. (Magyar olvasóknak a Linuxvilág első két számát ajánlom, bár a két szám csak az alap telepítőkészletet ismertette.)

A Debian telepítése első ízben – mondjuk ki nyíltan – a rendszergazdák büntetése. A leírás gyakran igen messzire vezet attól, amire az egyszerű halandóknak szükségük lenne a telepítéshez. Jó és rossz tanácsok, ütköző



➲ http://www.linuxdoc.org

nézetek és túlhaladott utasítások erdejét találhatjuk benne – egy leírásnak talán nem kellene ilyen keservesen nehéznek lennie. Jobban megismerve, talán nem is olyan nehéz, de a kezdők számára mindenképpen zavarbaejtő. Amikor még számítástechnikát tanítottam a monterey-i Naval Postgraduate Schoolban, tanultam egy szakkifejezést, amit sosem hallottam azelőtt (és azóta sem). A haditengerészeti pilótatanulók a gouge szó jelentését kérdezték. Ez olyan leírást jelent, ami pontosan megad minden fontos adatot, de csakis az odatartozó adatokat. A gouge-ok tehetséges újoncok számára készültek, és nem törekedtek a részletességre. A Debian is használhatna néhány gouge-ot. Most megpróbálkozom kiutat mutatni ebből a rengetegből.

Az első debianos nehézségem az volt, hogy az LSL Potato CD nem volt hajlandó a rendszerindításra. Valamilyen okból kifolyólag nem rendszerindító lemezként készítették el. Megpróbáltam Windows alól elindítani az E:\install\boot.bat paranccsal, de a leírás rámutatott, hogy ez a parancs csak MS-DOS módban működik, emulált DOS ablakban nem. Win98 SE-ben kiválasztottam a *Restart in MS-DOS mode* menüpontot a rendszerleállító menüből. Ez hibátlanul újraindította a rendszert MS-DOS módban, csakhogy előfordulhat, hogy szükségünk van a valós módban futó DOS meghajtókra, hogy olvasni tudjuk a CD-t. Álljunk csak meg egy pillanatra és gondoljuk végig.

Az online Debian GNU/Linux: Guide to Installation and Usage leírja, hogyan kell rendszerindító lemezt készíteni.

Az utasítások rosszak, mivel a fájlnevek, és a CD-kiosztás Potato alatt már megváltozott, de az alapelgondolás most is ugyanaz. Nem törődtem a Weben található figyelmeztetésekkel, melyek szerint a rendszerindító lemezek ördögiek. Nekem nem volt semmi gondom vele.

Íme, így kell két lemezről elindítani a Potatót, ha a CD nem érhető el DOS-ból:

1. Win98 SE-ben fel kell másolni a Debian CD E:/install könyvtárát a C: merevlemezre.

2. A rendszert indítsuk újra MS-DOS módban. A következő lépés az indítólemezkészítés (ugyanis a rawrite program nem működik a Windows DOS ablakában).
3. DOS-ban indítsuk el a C:\install\rawrite2.exe parancsot. Gépeld a rescue.bin és a: szavakat a megjelenő parancsjelhez és helyezd be az első hajlékonylemezt.
4. Ismételd meg ezt a root.bin és a: szavakkal, valamint a második lemezzel.
5. Indítsd újra a rendszert a frissen elkészült biztonsági lemezekkel, és íme, máris van egy egyszerű Linux-rendszered, amely látja a CD-t.

A merevlemez felosztása kissé elrettentő lehet. Van valami abban, hogy az összes adatod visszavonhatatlan elvesztését kockáztatod. A Debian telepítője a Linux cfdisk-be visz, ami bár szépen dolgozik, elsőre kicsit fenyegetően néz ki. Általános érvényű szabály: a hibák elkerüléséhez mindig ahhoz az operációs rendszerhez szánt programmal kell felosztani a merevlemezt, amit telepíteni szeretnénk.

Ha a lemezfelosztásról beszélünk, mindig szem előtt kell tartani, hogy a lemezfelosztási táblában összesen négy fő bejegyzés található. Az elsődleges és a kiterjesztett (extended) lemezterületek egy-egy ilyen bejegyzést használnak fel. A kiterjesztett lemezterületen belül létrehozott logikai területek azonban lényegesen többen lehetnek (az 5. sorszámtól fölfelé). A Win98 SE elsődleges lemezterületet hoz létre. A WinNT vagy a Win2K alapértelmezés szerint logikai lemezterületre kerül. Minden operációs rendszer esetében – ami lehetővé teszi, hogy logikai lemezterületet használjunk –, vigyázzunk, hogy ki ne fussunk a lemezfelosztási tábla bejegyzéseiből. Ha ugyanis kifutunk belőlük, nem tudunk több lemezterületet készíteni, még akkor sem, ha egyébként lenne még üres helyünk a merevlemezen.

A Linuxnak két lemezterületre van szüksége: a rendszerindító (boot) és a csereterületre (swap). Úgy döntöttem, a két linuxos lemezterület méretét együttesen 2 GB-ban határozom meg, 250 MB-ot tartva fenn a csereterület számára. Tehettem volna logikai lemezterületre a Linuxot, de mivel volt helyem, elsődleges lemezterületre raktam (bár ezzel szükségtelenül elpazaroltam egy bejegyzést). A Linux rendszerindító programja, a LILO a másik dolog, ami a hideglelést hozza az újdonsült felhasználókra. A félelmek ellenére, telepítése könnyű volt. Örültem, hogy nem követtem azt a széles körben elterjedt tanácsot, mely szerint az ntldr-t kell használni elsődleges rendszerbetöltő programként (ahogy azt a ➲ linuxdoc.org Win95 + WinNT + Linux Multiboot Using LILO Mini-HOWTO ajánlásában olvashatjuk). Sokkal választékosabb a LILO-val indítani mindent, ráadásul a telepítése is kevesebb gonddal jár. Amikor választanod kell, telepítsd a LILO-t az MBR-be (egyébként is ez az alapértelmezett választás). Így ideiglenesen működésképtelenné teszed a windowsos rendszerindítást. A következő lépés az ntldr elérésének visszaállítása a LILO-ban.

A /etc/lilo.conf fájlt kell módosítanod úgy, hogy a Windowsra mint másik rendszerre mutasson, a DOS indításhoz tartozó leírás formai követelményeit követve. Egyébként, ha még nem tudod, miképpen kell kezelni a unixos szövegszerkesztőket (például a vi-t), akkor mindenképpen meg kell tanulnod. (De megismerkedhetsz az `ae`, az `mcedit`, vagy a `joe` szerkesztőkkel, lásd még a 64. oldalon.) Nem szabad elfelejtened a /boot/bootmess.txt fájl módosítását és a /sbin/lilo parancs futtatását, hogy a változtatások érvénybe lépjenek. Indítsd újra a rendszert, és – ha arra utasítod – a Linux az ntldr-be (vagy a Windows 98 rendszerindítójába) visz, ahol megpillanthatod a szokásos windowsos indítási képernyőt. A másik operációs rendszer ezután már könnyedén indítható.

A régi meghajtómat szolgalemeznek kapcsoltam át, ugyanarra az IDE-kábelre kötöttem, amin az új meghajtó található, majd elindítottam a rendszert, és felismertettem a meghajtókat a BIOS-szal. Mivel egy másik ntldrm is volt a (most már) /dev/hdb eszközön, egy kicsit varázsolnom kellett a lilo.conf-ban található egyszerű map kapcsolóval, hogy engedélyezzem az indítómeghajtók cseréjét. Most egyaránt indíthattam Linuxot, az új Win98 SE rendszert, valamint a régi Win98 és WinNT rendszereimet is:

```
other=/dev/hda1
        label=Win2k
        alias=2
        table=/dev/hda

other=/dev/hdb1
        label=WinNT
        alias=4
        table=/dev/hdb
        map-drive=0x80
        to = 0x81
        map-drive=0x81
        to = 0x80
```

AWin98 SE megfelelően elindult, de amikor a Win2K-t próbáltam indítani, a *ntoskernel missing* hibaüzenetet kaptam. A megoldás a Windows lemezterület-számának növelése volt a c:/boot.ini fájlban. Ha ugyanis utólag iktattam be elsődleges lemezterületet (ahogy azt a Linux esetében szükségtelenül bár, de megtettem), az növelheti a logikai lemezterületek számát. Egyszerűen csak hozzá kellett adni egyet a lemezterületek számához. A logikai lemezterületek sorban kapják számaikat egyazon kiterjesztett lemezterületen belül.

```
multi(0)disk(0)rdisk(0)partitio
       n(3)\WINNT="Microsoft
Windows2000 Professional"
       /fastdetect
```

Beállítottam a Linux alatt a hálózati csatolók értékeit is:

```
# /etc/network/interfaces -- az
# ifup(8), ifdown(8)
# beállításfájljai

iface lo inet loopback

iface eth0 inet static
       address 192.168.1.2
       netmask 255.255.255.0
       gateway 192.168.1.1
```

és megadtam a tűzfalam IP-címét a /etc/resolv.conf-ban

```
nameserver 192.168.1.1
search 192.168.1.1
```

Összefoglalva az elmondottakat, elsőként a Windows operációs rendszereket telepítettem, majd végül a Linuxot. A sikertől mámorosan, immár könnyedén indíthattam Linuxot, Win2K, Win98 SE, WinNT vagy akár Win98 rendszert a LILO segítségével! Következő írásomban az Xfree86 beállítását követem végig, hogy elindíthassam az X Windows rendszert, illetve megfoltozzam a 2.2.17-es rendszermagot, hogy telepíthessem a Video4Linuxot. Így televíziót is nézhetek Linux alatt.

Robin Rowe (robin.rowe@movieeditor.com) 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 van többek közt ügyfél/kiszolgáló felépítésű 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.

Változó tartalom a Weben

A hagyományos programnyelvek, mint amilyen a C, vagy a Fortran, segítségünkre lehetnek a számításigényes webalkalmazások elkészítésekor.

Kezdetben, amikor az első webkiszolgálók megjelentek, elsődleges feladatuk az volt, hogy bizonyos állományokat távolról is el lehessen érni arról a gépről, amelyen futottak. Az ötlet magva, hogy a fájlok tartalmát egyszerűen át kell vinni HTTP segítségével a TCP-hálózaton keresztül. Hamar rájöttek azonban e megoldás ama korlátjára, hogy segítségével nem lehet változó tartalmat szolgáltatni. Erre kínált megoldást a webkiszolgálókhoz hozzáadott CGI-felület.

A Common Gateway Interface (Közös Átjárófelület) használata lehetőséget ad a webkiszolgálónak arra, hogy a kérelem beérkezésekor egy folyamatot elindítson. Ennek végeredményét a futtatott kód határozza meg, és ezt úgy küldi el az ügyfél böngészőprogramjának, mintha az egy állandó tartalmú fájl lenne. Azóta sokféle parancsfájlmotor és CGI-rendszer fejlődött ki, melyek egyszerűsítették a programozók feladatát és hatékonyabban kezelték a több külső szál futtatását (Perl, Python, PHP stb.). Ezeknek a nyelveknek azonban megvan az a hátrányuk, hogy futási időben értelmezettek. Nem hagyhatjuk figyelmen kívül azt sem, hogy rengeteg kódot írtak már C és Fortran (Emlékszünk még?) nyelveken, ezek bonyolult, számításigényes feladatokat oldottak meg. Nagyon hasznos egy lekérdezés alapján előálló változó tartalmat értelmezett nyelvekkel létrehozni, de nem lenne jó ötlet ezekkel megoldani képfeldolgozási vagy Fourier-átalakítási feladatokat, mert a megvalósításukhoz szükséges idő jóval meghaladná azt a válaszidőt, amit a felhasználó elvár. Két okunk is van tehát arra, hogy C vagy Fortran nyelvű CGI-alkalmazásokat írjunk. Ezeket azután természetesen gépi kódra fordíthatjuk. Egyrészt számos előre megírt forráskód létezik, másrészt bizonyos feladatok igénylik a C és a Fortran által lehetővé tett számítási sebességet.

Hogyan adja át a webkiszolgáló az adatokat a programunknak?

A HTTP-protokollal két módszer használható a böngészőprogramból a webkiszolgáló felé irányuló adatátvitelre. Az egyik a GET módszer, itt az adat valójában az URL része, általában a „?” utáni rész. A másik pedig a POST módszer, ez a böngészőprogram által visszaküldött űrlap név–érték párjaiból áll. A GET megértéséhez tekintsük a következő címet: ➲ http://www.mydomain.com/pages/external.cgi?additional-data.

A GET adatrész: additional-data

A POST működése egy csöppet bonyolultabb, vegyük szemügyre a kérést létrehozó oldal forrását, keressük meg az elküldendő űrlapot:

```
<FORM>

<INPUT TYPE="text" NAME="fld01" VALUE="val01">

<INPUT TYPE="hidden" NAME="fld02" VALUE="val02">

<INPUT TYPE="checkbox" NAME="fld03" CHECKED>

<SELECT NAME="fld04">

<OPTION VALUE="val03"> Val-03
```

➲ www.ichus.net/CGI/City/

```
<OPTION SELECTED VALUE="val04"> Val-04

<OPTION VALUE="val05"> Val-05

</SELECT>

</FORM>
```

A POST adatok

```
fld01=val01&fld02=val02&fld03=on&fld04=val04
```

Láthatjuk, hogy a POST adatok egy folytonos karakterláncban érkeznek, ahol a név és értéke között egyenlőségjel ("="), az egyes név–érték párok között „és” jel (&) található.

Valójában a POST adatok átviteli formátuma tovább bonyolódik, mert bizonyos karakterektől „meg kell védeni” a webkiszolgálót. Ilyenek például a vezérlőkarakterek és az elválasztókarakterek, ezek ugyanis megzavarnák a webkiszolgálót. Erre az a megoldás született, hogy pluszjelekkel ("+") kell lecserélnünk a szóközöket és a nem nyomtatható karaktereket, az és jelet, valamint a plusz- és az egyenlőségjelet, és a „%[0-9,A-F][0-9,A-F]” formában kódoljuk. Igen, a szóközt nemcsak a "+" jellel, hanem a "%20" kódsorozattal is leírhatjuk. Az Apache, valamint az IIS általam ismert változatai mindkettőt elfogadják.

A webböngésző másféle módon adja át a GET és a POST adatokat a külső szálnak. A GET adatok egy környezeti változóba kerülnek, ez az adott szál helyi környezetében érhető el. A környezeti változó neve "QUERY_STRING". A GET adatok megszerzése C/C++ programból rendkívül egyszerű:

```
char *pszGetData = getenv("QUERY_STRING");
```

Ez minden Unix és Microsoft fejlesztői környezetben működik. A POST adatok a külső szál szabványos bemenetére kerülnek.

Akiknek nem ismerős a szabványos bemenet fogalma, gondoljanak arra, hogy ez olyan adatforrás, mint a billentyűzet, tehát ugyanolyan módon kell a POST adatokat feldolgozni, mint a billentyűzeten keresztül bevitt adatokat. A szabványos bemenet folyamtípusú, és a folyam természetéből fakad, hogy nem lehet tudni előre, mennyi adat vár feldolgozásra. Kézenfekvő megoldás lehet addig olvasni bájtról bájtra a folyamot, amíg véget nem ér, és ennek megfelelően menet közben növelni a tárolóhelyet. Szerencsére, a webböngészők átadnak egy további adatot, amely

➲ cgi.resourceindex.com

megmenti a fejlesztőket attól, hogy a növekvő tárolókkal bajlódjanak és lehetővé teszi, hogy lekezeljék a kivételeket, amikor egy futási időben kért memóriafoglalás nem jár sikerrel.
Amikor a webböngésző átadja a POST adatokat a külső szál szabványos bemenetére, az egész POST adathalmazt egyszerre helyezi oda, ugyanis azután, hogy elkezdtük kiolvasni az adatokat a folyamból nem fogadható további adat.
A webböngésző azt is elárulja a folyamatnak, hogy mennyi adatot helyezett a szabványos bemenetére. A szál helyi környezetében elérhető a „CONTENT_LENGTH” nevű környezeti változó, amelyben ASCII szövegként van megadva a szabványos bemeneten beolvasásra váró bájtok száma. Tehát a POST adatok megszerzése C/C++-ban egy háromlépcsős folyamat, amely minden Unix és Microsoft környezetben működik:

```
long    iContentLength = \
        atol(getenv("CONTENT_LENGTH"));

char   *szFormData = (char *) \
        malloc(iContentLength * sizeof(char));

bzero(szFormData, iContentLength * sizeof(char));

fread(szFormData, (iContentLength - 1) * \
        sizeof(char), 1, stdin);
```

Egy adott weboldal tartalmazhat egyidejűleg GET és POST adatokat is, a két módszer tehát egyszerre használható. A <FORM> tagban adható meg, hogy az űrlap tartalma GET vagy POST módszerrel kerüljön továbbításra.

Hogyan adunk vissza adatokat a webkiszolgálónak?

A weboldalról megszerzett adatokat programunk feldolgozza, és megmondja a webkiszolgálónak, hogy mit kell válaszolnia. A válasz

1. lista Számolás tízig

```
printf("HTTP/1.0 200 Okay\n"

"Content-;Type: multipart/x-;mixed-;replace;"

"boundary=SoMeRaNdOmTeXt\n"

"\n--SoMeRaNdOmTeXt\n");

for (Count = 1; Count <= 10; Count++) {

printf("Content-type:
        ➥text/html\n\n<HTML><HEAD><HEAD>"

       "<BODY><H3>%d.
        ➥frissítés</H3></BODY></HTML>\n"

       "\n--SoMeRaNdOmTeXt\n", Count);

fflush(stdout);

sleep(1);

} /* end for */

printf("Content-type:
        ➥text/html\n\n<HTML><HEAD></HEAD>"

       "<BODY><H3>Ennyi volt,
        ➥srácok!</H3></BODY></HTML>\n");

fflush(stdout);
```

lehet sima szöveg, HTML-dokumentum (ez a leggyakoribb), kép (rendszerint GIF vagy JPEG) vagy bármi más összetett adattípus. Ezeket az adattípusokat MIME-típusoknak nevezik, és szabványos részhalmazukat majdnem minden jelenleg használatos böngésző-program ismeri. Az adatátvitel a programunktól a webkiszolgálóig – amennyiben az ügyfél böngészőprogramjának szánjuk – úgy zajlik, hogy a szál szabványos kimeneti folyamába írunk, pontosan akként, ahogyan a képernyőre írnánk kedvenc programnyelvünkön. A használandó adatformátum egyszerű:

```
Content-type:[szóköz][MIME-típus];
       [CR][CR][Dokumentumadatok]
```

Első lépésként a programunknak meg kell adni a MIME-típust. Ezek a következők lehetnek:

- text/plain – egyszerű szöveg, ezt írógépbetűkkel jeleníti meg pontosan úgy, ahogyan átküldjük, nem változtatja a szóközöket és a sorvégejeleket.
- text/html – szabványos HTML-dokumentum.
- image/gif – a Compuserve GIF előírásainak megfelelően kódolt kép. Itt jegyzendő meg, hogy a formátum a Lempel-Ziv (LZ77) tömörítő eljárást használja. Bővebb tájékoztatásért érdemes megnézni a ➲ http://lpf.ai.mit.edu/Patents/Gif/Gif.html oldalt és a ➲ www.unisys.com/unisys/lzw/, illetve ➲ www.unisys.com/unisys/lzw/lzw-license.asp oldalakat.
- image/jpeg – a JPEG képszabvány használatával kódolt kép.

Második lépésként a programunknak el kell küldenie egy pontosvesszőt (";"), amelyet két „kocsi vissza” karakternek kell követnie ("\n"). Harmadikként pedig el kell készítenie az átvitelre kerülő dokumentum törzsét. Ez lehet sima szöveg, HTML vagy olyan nyers adat, amely egy GIF vagy JPEG képet alkot.
A webkiszolgálót tehát nagyon egyszerű rávenni a válaszadásra, csak ezt kell írnunk:

```
printf("Content-type: text/html\n\n<HTML><HEAD>
        ➥</HEAD>"

"<BODY><H3>Tesztoldal</H3></BODY></HTML>\n");
```

Alapesetben mindössze ennyit kell tennünk, hogy a webkiszolgáló az ügyfélprogram kérésére válaszoljon. Természetesen az alapeset néhány trükkös kiegészítésével befolyásolhatjuk az átküldött dokumentum megjelenését. Az egyik ilyen kiegészítés a "charset=" módosító kifejezés a MIME-típus után, rögtön a „kocsi vissza” karakter előtt, ez utasítja a böngészőt a megfelelő karakterkészlet használatára (például „ISO-;9660-;1”, „ISO-;8859-;2”, „KOI-;8”, „WIN-;1250” stb.). Nézzük meg, hogyan küldhetünk át egy orosz nyelvű ellenőrzőoldalt:

```
printf("Content-type: text/html;
        ➥charset=KOI-;8\n\n"
       "<HTML><HEAD></HEAD><BODY><H3>"
       "<BODY><H3>Maya malinkayaproba
        ➥</H3></BODY></HTML>\n");
```

Folyamatos frissítések küldése a böngészőnek

Gyakran előfordul, hogy egy weboldalt olyan hosszan tartó folyamatok megfigyelésére használnak, melyek tovább tartanak annál, mint amit a webböngésző képes kivárni. Ezt a helyzetet is jól lehet kezelni a hagyományos programnyelvek segítségével. Az alapötlet az, hogy utasítjuk a webkiszolgálót a böngészővel való TCP-kapcsolat fenntartására, és bizonyos időközönként – ezt programunkban adjuk meg – új dokumentumot küldünk át.
Az itt megadott leírás az Apache webkiszolgálóra érvényes. Azért ezt választottuk, mert manapság ez a legnépszerűbb HTTP-démon a linuxos világban. Elképzelhető, hogy más HTTP-démonnal is működik a dolog, mindenkit bátorítok, hogy próbálja ki a saját kiszolgálójával, és számoljon be az eredményről. Ezek a lépések szükségesek hozzá:

1. Nevezzük át programunkat úgy, hogy az "nph-;" karakterekkel kezdődjön. Azaz ha a program neve "update.cgi" volt, változtassuk meg "nph-;update.cgi"-re.
2. Küldjük át a HTTP-fejlécet. Ezt szokásos esetben a webkiszolgáló küldené el a böngészőnek:
   `printf("HTTP/1.0 200 Okay\n");`
3. A dokumentum MIME-típusaként adjuk meg ezt: „multipart/x-mixed-replace”:

   ```
   printf("Content-;Type:multipart/x-;
           ➥mixed-;replace;"
          "boundary=SoMeRaNdOmTeXt\n");
   ```

4. Kezdeményezzük az első dokumentum átvitelét a „boundary”-ben megadott kulcsszó átadásával:

   ```
   printf("\n--SoMeRaNdOmTeXt\n");
   ```

5. Küldjük el a frissített dokumentumot. Ez egyszerűen egy dokumentum, amely addig marad a böngészőablakban, amíg ugyanazon a megnyitott kapcsolaton keresztül érkező másik frissítés fel nem váltja valamikor a jövőben. A frissítést ismét a „boundary”-ben megadott kulcsszó követi:

   ```
   printf("Content-type: text/html\n\n<HTML><HEAD>
           ➥</HEAD>"
   "<BODY><H3>%d. frissítés</H3></BODY></HTML>\n"
   "\n--SoMeRaNdOmTeXt\n", Count++);
   ```

6. Kényszerítsük ki a szabványos kimenet tárolójának kiírását:

   ```
   fflush(stdout);
   ```

7. Ismételjük meg az ötödiktől a hetedikig tartó lépéseket mindaddig, amíg az összes frissítést át nem küldtük. Az utolsó frissítéskor ne küldjük át a határoló kulcsszót, csak egyszerűen kényszerítsük ki a szabványos kimenet kiírását és lépjünk ki. Ezáltal az utolsó frissítés eredménye a böngészőablakban marad a kilépés után is.

A kiszolgálóoldali kényszerített frissítést mutatja be az *1. listán* olvasható egyszerű példa, amely egytől tízig számol a böngészőablakban. Minden frissítés között egy másodperc telik el.
Ennek magyarázatához meg kell érteni a kiszolgáló háttérben végzett tevékenységét. Eddig a programunk kimenetét leellenőrizte a kiszolgáló, például a megfelelő MIME-típus, a megfelelő elválasztójel stb. tekintetében, majd mielőtt elküldte volna a böngészőprogramnak, átküldött egy HTTP-fejlécet. Azért, hogy jobban befolyásolhassuk a webkiszolgáló és a böngészőprogram párbeszédét, meg kell kérnünk a webkiszolgálót, hogy ne végezze el az ellenőrzéseket, és ne küldjön HTTP-fejlécet. Ezért adtuk az „nph-”-t a program fájlnevéhez, mert ez azt jelenti: No Parsed Headers, azaz nem dolgozza fel a fejlécet. Ha a programunk neve „nph-”-val kezdődik, a webkiszolgáló úgy veszi, hogy a program maga gondoskodik a szükséges ellenőrzések elvégzéséről és a fejléc átküldéséről, ami pedig szokásos esetben a webkiszolgáló feladata lenne. A webkiszolgáló csak a böngészővel felépített TCP-kapcsolat fenntartásáért felelős, és a programunk szabványos kimeneti folyamát egy az egyben átküldi böngészőnek ezen a TCP-csatornán keresztül. Most már világos, miért kellett a második lépésben elküldeni a HTTP-fejlécet.
Következő lépésben közölnünk kell a böngészővel, hogy folyamatos frissítésekre számítson, ne csak egyszeri adatcsomagra, ezért nem szabad lezárni a TCP-csatornát az első dokumentum átvitele után. Ezt a dokumentum MIME-típusának megfelelő beállításával (`multipart/x-;mixed-;replace`) érhetjük el. Ezenkívül a böngészővel tudatni kell, hogyan különítheti el az egyes dokumentumokat a sok dokumentumból álló folyamban. Ezt a MIME-típus megadását követő `boundary=SoMeRaNdOmTeXt` módosító kifejezéssel oldhatjuk meg. Ezzel azt mondjuk a webböngészőnek, ha bármikor találkozik a `--SoMeRaNdOmTeXt` bájtsorozattal a bemeneti folyamban, meg kell állnia, és feltételezni, hogy az ezt követő adatok egy új dokumentum részei, amelyek az előzőt váltják fel a böngészőablakban.
A dokumentum végét és a következő dokumentum elejét elválasztó karakterláncot határoló kulcsszónak (vagy röviden határszónak) nevezik, és általában sokkal bonyolultabb és hosszabb, mint az itt megadott példa. Többnyire 50–60 bájt hosszú számokat és betűket tartalmazó véletlen karakterláncot szoktak választani erre a célra. A karakterláncnak elég hosszúnak és elég véletlenszerűnek kell lennie ahhoz, hogy elhanyagolható legyen annak az esélye, hogy a dokumentum törzsében valahol véletlenül előfordul.
Végül, miután a dokumentumot kiírtuk a szabványos kimenetre, és a határszót is kiírtuk utána, ki kell kényszeríteni a szabványos kime-

4. lista CGI keretrendszer hagyományos programnyelven

```
void GenerateHtmlDocument () {

char   *TextField     = GetFormStringValue
                        ("TextField");

int     NumericField = GetFormIntegerValue
                        ("IntegerField");

float   FloatField    = GetFormFloatValue
                        ("FloatField");

printf("<HTML><HEAD><HEAD><BODY><H3><CENTER>"

       "TextField     = [%s] <BR>"

       "NumericField = [%d] <BR>"

       "FloatField    = [%f] <BR>"

       "<CENTER><H3><BODY><HTML>\n",

       TextField, NumericField, FloatField);

} /* end function GenerateHtmlDocument() */

int main (int argc, char **argv) {

       printf("Content-Type: text/html;\n\n");

       GenerateHtmlDocument();

       ReleaseFormData();

} /* end of main() function */
```

net tárolójának tényleges kiírását, mert csak így érhetik el az adatok a böngészőprogramot. Ha ezt nem tennénk, a démon az adatokat nem küldené el mindaddig, amíg a folyam átmeneti tárolója – ennek mérete függ a használt operációs rendszertől – be nem telik, és a kiürítést el nem végzi az operációs rendszer.

Biztonsági kérdések és a webkiszolgáló bemenetének értelmezése

A legtöbben a biztonsági kérdések miatt rettennek vissza webes programok hagyományos programnyelveken történő megírásától. Valószínűleg a biztonság megteremtése volt a Perl és a PHP nyelvek fejlesztésének egyik legfontosabb célja. Nehéz olyan alkalmazást írni, amely elég ügyes ahhoz, hogy az összes lehetséges támadást kivédje, miközben az adatokat csak környezeti változókban és a szabványos bemeneti folyamban adhatja át a böngésző a programnak.
Az első megoldásra váró nehéz feladat az adatok egyszerű és memóriatakarékos feldolgozása, tehát hogy egyszerűen kiválaszthassuk a keresett mezőt, és az adatot egy lépésben megszerezhessük. Ezenkívül gondolnunk kell bizonyos biztonsági gondokra, például az adattúlcsordulásra, amit a rosszindulatú böngészőprogram okoz, felülírva az alkalmazás memóriáját a túlcsordult adattal vagy akár meg is bénítva a szolgáltatást.
Bemutatunk egy függvénycsaládot, amely pontosan az említett egylépéses adatelérést valósítja meg a POST adatok feldolgozására és a biztonságra egyaránt ügyelve. A példa C nyelvű, de könnyen átvihető Fortranra vagy C++-ra:

```
char   *TextField     = GetFormStringValue
                        ("TextField");

int     NumericField = GetFormIntegerValue
                        ("IntegerField");
float   FloatField    = GetFormFloatValue
                        ("FloatField");
```

A függvények forrását a 2. lista mutatja be, az ezeket támogató függvényeket pedig a 3. lista tartalmazza. Terjedelmi okokból ezeket a listákat nem közöljük az újságban, de letölthetők az ➲ ftp://ftp.ssc.com/pub/lj/listings/issue82 címről. Az összes megadott függvényt kipróbáltuk Unix és Windows környezetben is, jól működtek, és ellenállóak a tárolók alulcsordulásával vagy túlcsordulásával szemben. Minden függvény az első meghívásakor lefoglal egy memóriaterületet, és itt elhelyezi a feldolgozott POST adatokat. Ez megmarad a memóriában, és a következő függvényhívás alkalmával egyszerű lineáris kereséssel végigmegyünk ezen a memóriaterületen a kívánt mezők után kutatva. A memóriafoglalásra csak egyszer kerül sor, és a különleges karakterek átalakítása ezen a memóriaterületen zajlik sorfolytonosan, ugyanis erre a célra nem használunk más ideiglenes tárterületet.
Mivel a bemutatott példa C nyelvű, melyben nincs a C++-ból ismert automatikus megszüntető (destructor), a program kilépése előtt egy takarítófüggvényt szükséges meghívni. Ez a ReleaseFormData(), mely nélkülözhetetlen a futási időben lefoglalt tárterület felszabadításához. Ha ezeket a függvényeket C++ osztályokká alakítjuk, egyszerűen csak meg kell hívni ezt a függvényt a POST adatok elérését megvalósító osztály megszüntetőjében. A *4. listán* bemutatunk egy egyszerű keretrendszert egy hagyományos programnyelven megírt CGI-programhoz.

Miről lesz szó a jövőben?

Természetesen csak felületesen érintettük, mi történik akkor, ha a gyors és hatékony nyelveket (pl.: C/C++) használjuk webes alkalmazások fejlesztésére, és megszabadulunk attól a tehertételtől, amit az értelmezett nyelvek feldolgozása jelent. Könnyen belátható, miért kell még az alábbiakról is szót ejtenünk:

- A helyi fájlrendszer használata a CGI-programjaink állapotának tárolására.
- Miért tárolható az állapot a helyi fájlrendszeren Linux alatt lemeztúlterhelés veszélye nélkül, ami más operációs rendszerek használata esetén felléphet.
- Sütik létrehozása, módosítása és megsemmisítése az ügyfél böngészőprogramjában CGI-programok segítségével.
- Biztonsági beállítások, hogy csak a mi CGI-programunk érhesse el az állapotadatokat a helyi fájlrendszeren, és senki másé.
- Pehelysúlyú szálak és a gyors CGI.

Dan Teodor tanácsadóként dolgozik a PriceWaterhouseCoopersnél a texasi Houstonban. Megszállott Linux-rajongó középiskolás évei és a Slackware 0.x rendszermag-kiadások óta. Nagyratörő terveket szövöget a webalkalmazások elterjesztéséről és a sajátos üzleti modellekről, miközben arról ábrándozik, hogy Új-Mexikóban síel, és minden évben alapít egy csődbe menő dot-com vállalatot.

Dinamikus alkalmazásfejlesztés

Az eszközválasztás szempontjai.

Nemrég abban a megtiszteltetésben részesültem, hogy egy nagy, philadelphiai orvosi kiadó cég internetes részlegénél dolgozhattam Pennsylvaniában. A részlegünk a cég számos marketingosztályának nyújtott segítséget különféle Internetre szánt anyagok kezelésében, illetve irányította és ütemezte a tartalom elhelyezését a honlapján. A részlegünk gyakorlatilag összekötőként szolgált a kiadó különféle marketing- és termékfelügyeleti osztályai között Philadelphiában, New Yorkban, Chicagóban és a Baltimore-i IT részlegben, ahol a webkiszolgálók üzemeltek.

A tevékenységgel együtt járt, hogy hatalmas mennyiségű anyag került a legkülönfélébb utakon a részlegünköz. A legtöbb adat elektronikus levélben érkezett, egyes fájlok lemezen jöttek az irodák közötti postával, míg továbbiakat közvetlenül a Baltimore-i IT részlegen található ftp-kiszolgálókra töltöttek fel. A kavarodást csak tovább növelte, hogy az anyagokból sokszor többféle változat is létezett, mivel a részlegek még az utolsó pillanatban is módosítottak rajtuk. A feladatom az volt, hogy a különböző tartalmakat központi alkalmazással szervezve ideiglenes megoldást dolgozzak ki a különféle anyagok érkezése miatt keletkező kavarodás mérséklésére. Ezt egy kisebb webes alkalmazással igyekeztem elérni, mely az egyes tervezetek címét, kezelőjét, részlegét, részlegének kódját és költségelszámolási kódját volt hivatott követni.

Az alkalmazást gyorsan és olcsón kellett elkészíteni, mivel a vásárlások visszafogása és bizonyos beszerzési folyamatok miatt a következő pénzügyi évig nem volt mód jelentősebb fejlesztésre. Teljesen hétköznapi PC-s felszerelést kaptam, a részleg már meglévő eszközeiből. Az alkalmazásnak kicsinek, gyorsnak, könnyen használhatónak kellett lennie, további követelmény volt, hogy a részleg személyzete is is képes legyen karban tartani. Ők viszont inkább üzleti, és nem szakmai szempontból kezelték a világhálót. Annak ellenére, hogy ez volt az internetes részleg, tevékenysége inkább a tartalom felügyeletére korlátozódott. Az IT részleg kezelte a nagyobb terhelésű webkiszolgálókat és az irodán kívüli távközlési eszközöket.

Célkitűzés

Írásom egyszerűbb dinamikus internetes alkalmazás készítéséhez szükséges és elérhető eszközök tanulmányozása során szerzett tapasztalataimra épül. Az összefoglalást a kereskedelmi és ingyenes programokhoz egyaránt elkészítettem. Néhány meghatározással fogjuk kezdeni, illetve a dinamikus internetes alkalmazások hátterével, majd áttekintünk néhány elérhető módszert. Szándékom szerint a cikk kiindulópontként szolgál majd a jelenlegi és leendő fejlesztők számára, ha internetes alkalmazásaik fejlesztéséhez eszközöket keresnek.

Milyen egy dinamikus honlap?

A haladó honlaptervezők a HTML és a CSS mellett gyakran egy JavaScriptnek nevezett parancsnyelvet, illetve a honlap összetevőinek elnevezési rendszerét – a Dokumentum Objektum Modellt, avagy a DOM-ot – használják dinamikus tartalom készítésére. Az eredményt gyakran dinamikus HTML-nek, vagy DHTML-nek is nevezzük.

A megvalósítás során készített parancsfájlok a JavaScripttel ellentétben a kiszolgálón, és nem az ügyfélnél futnak. Az itt ismertetett eszközök és módszerek lehetővé teszik, hogy a későbbiek folyamán további interaktív vagy stílusbeli elemekkel bővítsük az alkalmazást.

Mi az adatbázis?

Az adatot tények, számok, betűk és jelek összességeként határozzuk meg, amit számítógép vagy távközlési rendszer dolgoz fel, hogy értelmezhető adatot állítson elő belőle. Számítógépes rendszerekben ezeket az elemeket jellemzően állományokban tárolják. Az egymással összefüggő állományok adatbázist alkotnak. A fájlokon belül az elemeket sorokba és oszlopokba rendezzük. Ebben az esetben azonban valami kifinomultabb rendszerre van szükség, mint állományok halmazára.

A tervezethez egy relációs adatbázis-kezelő rendszer (RDBMS) szükséges. Az RDBMS olyan programcsomag, mely az adatokat sorokból és oszlopokból álló táblákban tárolja. A különböző táblák közt relációk hozhatók létre, így adható válasz a felhasználó által feltett kérdésekre. Ezeket a kérdéseket lekérdezéseknek nevezzük.

Az RDBMS elgondolást először *Codd* vezette be 1970-ben egyetemi tanulmányában, kereskedelmi forgalomban azonban a hetvenes évek közepéig nem létezett ilyen rendszer. Az RDBMS választ ad minden lekérdezésre, függetlenül attól, hogy az adatokat lekérdezzük, frissítjük vagy töröljük a táblából.

A honlapot kiszolgáló RDBMS-t gyakran háttérrendszernek is nevezzük. A honlapokat, melyeket a felhasználó lát, megjelenő szolgáltatásnak hívjuk. A háttérrendszernek az SQL (Structured Query Language – rendezett lekérdezőnyelv) használatával tehetők fel a kérdések. Röviden összefoglalva, ha adatokat kell feldolgozni, valamilyen DBMS-re mindenképpen szükség lesz. A legnépszerűbb rendszerek napjainkban a relációs DBMS-ek, melyek SQL lekérdezésekre és parancsokra adnak választ.

Az adatbázis kiválasztása

Oracle

Az Oracle olyan termék, melynek neve az adatbázis szó rokon értelmű kifejezésévé vált. Az Oracle (➲ http://www.oracle.com/) jelentős szerepet játszott abban, hogy jelenleg igen elterjedtek a relációs adatbázisok. Az évek során adatbázis-kiszolgálója jelentős elismerést vívott ki, mint teljes szolgáltatáskészletet nyújtó, gyors és megbízható rendszer. Az Oracle a Linuxot is támogatja, és a jelek szerint a linuxos rendszerek elkötelezettje.

Ennek ellenére van két jelentős ok, amiért ennél a fejlesztésnél nem használtam Oracle-t. Először is a gépre vonatkozó követelményei bőven meghaladták az alkalmazás fejlesztéséhez és kiszolgálásához rendelkezésre álló gépét. Mindjárt kezdetként 800 MB helyre van szükség a merevlemezen, és 256 MB memóriára. Másodszor az Oracle túlságosan drága választás lett volna ekkora „átmeneti” alkalmazáshoz. Még a legkevesebb szerződési költséget véve is, öt át nem ruházható felhasználói joggal egyetlen kiszolgálóra is 800 dollárt kellett volna fizetni a program használatáért, azaz felhasználónként 160 dollárt.

MySQL

A felhasználók számos nyílt forráskódú alkalmazást érhetnek el. Figyelembe véve a működő honlapokat, a MySQL ➲ http://www.mysql.org/ rendkívül népszerű választásnak tűnik a Linux-közösségen belül.

A MySQL nagyon gyors, többszálú, többfelhasználós és erőteljes SQL adatbázis-kiszolgáló. A MySQL jelenleg szintén nyílt forráskódú, és nemrég szövetségre lépett a VA Linux Systems céggel, mely linuxos

Kapcsolódó címek

Axmark, 2000 MySQL Manual – a kézikönyvet *David Axmark*, *Michael (Monty) Widenius* és *Paul DuBois* készítette és gondozza
➲ http://www.mysql.com/documentation/mysql/commented/
Allaire ➲ http://www.allaire.com/
Gilmore, 2000
➲ http://www.oreillynet.com/pub/a/ network/2000/06/16/magazine/php_mysql.html
MySQL Org ➲ http://www.mysql.org/

Oracle, 2000, Press Release
➲ http://www.oracle.com/corporate/press/index.html?134934.html
Oracle FAQ, 2000, Oracle 8i on Red Hat 6.1 Installation FAQ
➲ http://platforms.oracle.com/linux/
Oracle árak, 2000 ➲ http://oraclestore.oracle.com/

Perl Org ➲ http://www.perl.org/
PHP ➲ http://www.php.net/

PostgreSQL ➲ http://www.postgresql.org/
PostgreSQL Admin Guide, melyet Thomas Lockhart szerkeszt
➲ http://www.postgresql.org/doxlist.html
PostgreSQL FAQ, 2000
➲ http://www.postgresql.org/docs/faq-english.html

gépeket árusít és támogat. A MySQL első nyilvános kiadása 1996 novemberében jelent meg, és kezdettől fogva a forráskóddal együtt volt elérhető. A MySQL bizonyítottan villámgyors és megbízható adatbázis-megoldás, melyet egyre több cég használ. Többek közt az SGI, a ValueClick, a Nortel/Insight, a Tucows.com, a Cisco és sok más cég bizalmát is elnyerte.
Úgy tűnt, hogy a MySQL több mint megfelelő megoldás a kezdetben leírt egyszerű kis alkalmazáshoz. Nagy méretű rekordhalmazokkal is rendkívül gyorsnak bizonyul. A MySQL kézikönyve ötvenmilliónál több rekordot tartalmazó rendszerekről is beszámol. Ezenkívül egyre szorosabb együttműködés jelei mutatkoznak a MySQL és a PHP fejlesztői csapatok között. A dinamikus páros növekvő népszerűsége, társulva mindkét oldal fő fejlesztőinek határtalan lelkesedésével idén az izraeli találkozóban csúcsosodott ki. Az eredmény a PHP 4.0 csomaggal együtt közreadott MySQL könyvtár lett, illetve egy megegyezés, miszerint a felek kölcsönösen segítik egymást a termékek együttműködésének és minőségének javításában.
Ennek ellenére a MySQL használatának is van néhány hátránya. Ezek egyike a tranzakciók területén jelentkezik. A tranzakció (néhány nyelvjárásban: ügylet) egymással összefüggő változtatások halmaza egy adatbázisban. Az SQL szabvány előírja, hogy változtatások egész sora adható ki az adatbázisnak, majd oszthatatlan egységként hajtható végre vagy vonható vissza. Így lehetővé válik, hogy például az adatbázis különböző tábláiban található számlaszámok között pénzt mozgassunk át úgy, hogy az egyik számlához hozzáadjuk az összeget, majd megpróbáljuk levonni a másikból. Ha a másik változtatás sikertelen, az egész műveletsort egyetlen lépéssel visszavonhatjuk.
A tranzakciók támogatásának hiánya nem jelentett volna ugyan feltétlenül gondot az alkalmazás készítésekor, a jövőre azonban még ilyen kis alkalmazásnál is gondolni kell, így más rendszer után néztem, mely a tranzakciókat is kezeli.
Meg kell említeni, hogy a MySQL az idegen kulcsokat is csak korlátozottan támogatja. Az idegen kulcs a relációs adatmodell fontos elemét képezi. Azt a módot jelenti, ahogy a viszonyokat ábrázoljuk, amolyan ragasztóként is felfoghatjuk, mely a táblákat együtt tartja, hogy azok egy relációs adatbázist formálhassanak.
Egy másik területén is gyengének találtam a MySQL-t, a részlekérdezéseket ugyanis nem támogatja. Ezzel a szolgáltatással a fejlesztő egy SQL kifejezést egy másikba ágyaz bele. Ismétlem, hogy alkalmazásom kicsi ugyan, és valószínűleg nem okoztak volna túl nagy gondot a MySQL említett hiányosságai, mégis úgy döntöttem, másik RDBM-et keresek.

PostgreSQL

A MySQL kapcsán felmerült hiányosságokat a jelek szerint pótolta a PostgreSQL (➲ http://www.postgresql.org/) fejlesztői csapata. A PostgreSQL 6.5.3 változata a RedHat 6.2-es kiadásában is megtalálható. Ez a változat már támogatja a tranzakciókat, a részlekérdezéseket, bár az idegen kulcsok terén csak korlátozottak a képességei.
A PostgreSQL leírása szerint a PostgreSQL a nagy, kereskedelmi DBMS-ekhez hasonlóan már a tranzakciók, a részlekérdezések, a triggerek, a nézetek és a kifinomult zárolások használatának lehetőségét is biztosítja.
Van néhány olyan szolgáltatása is, amelyeket más szabad adatbázis-rendszerek nem nyújtanak. Ilyenek például a felhasználó által meghatározott típusok, az öröklés, a szabályok és a többváltozatú versenyhelyzet-kezelés a zárolások miatt jelentkező torlódások csökkentésére. Ez a bővebb szolgáltatáskészlet sajnos nem biztosítható a MySQL-ével megegyező sebességgel. Összehasonlítva a MySQL-lel vagy a tanulásra képes adatbázis-rendszerekkel, a PostgreSQL lassabb a frissítések és beillesztések végrehajtásánál, mivel a tranzakciók miatt jelentkező többletterhelést is kezelnie kell. Hasonlóan a MySQL-hez és éles ellentétben az Oracle-lel a PostgreSQL legkisebb rendszerkövetelményei szerények. A PostgreSQL rendszerfelügyeleti kézikönyve szerint, bár a PostgreSQL futtatásához akár 8 MB memória is elegendő lehet, észrevehető sebességnövekedés tapasztalható, ha 96 megabájt vagy annál is több memóriát használ. Az alapszabály az, hogy memóriából sosem lehet túl sok. Azt is ellenőrizni kell, hogy elegendő terület áll-e rendelkezésre a merevlemezen. A fordítás alatt körülbelül harminc megabájt szükséges a forrásokhoz, és nagyjából öt megabájt a telepítési könyvtárhoz. Egy üres adatbázis körülbelül egy megabájt helyet foglal, egyéb esetekben hozzávetőlegesen ötször akkora, mint amekkora a benne tárolt adatok mérete egyszerű, szöveges állományban lenne. Ha az időutazásos ellenőrzést is futtatjuk, átmenetileg további húsz megabájt helyre is szükség lesz.
Az általam használt számítógép bőven megfelelt ezeknek a követelményeknek, így a sebesség csökkenésétől nem kellett tartanom. Elégedett voltam azzal, ahogy a PostgreSQL megoldja a MySQL- és Oracle-termékek kapcsán felmerült gondjaimat, tehát úgy döntöttem, hogy tervem megvalósításához a PostgreSQL-t használom.

Parancsnyelv választása

Az adatbázisokhoz hasonlóan a parancsfájlok készítésére alkalmas programnyelvekből is rengeteg elérhető a Linuxhoz. Ezek közül több is használható kiszolgálóoldali programok készítésére dinamikus honlapokhoz.

ColdFusion

A kereskedelmi termékek közül az egyik lehetséges választás az Allaire cég (➲ http://www.allaire.com) ColdFusion rendszere volt. Az Allaire termékét széles körben használják, a környezet kiszolgálóoldali része teljes támogatást élvez és egyaránt elérhető Linux és Windows NT-rendszereken. Az Allaire a ColdFusion Studiónak nevezett csomag részeként nagyon csinos fejlesztőkörnyezetet kínál, melynek futtatásához Windows 9x vagy NT operációs rendszer szükséges. Mindkét összetevő letölthető kipróbálásra, illetve megvásárolható az Allaire honlapján.

A ColdFusion csomag összetevőit a ColdFusion Markup Language (CFML) támogatására készítették. A CFML formai követelménye nagyon hasonló a HTML-éhez, nyitó és záró jeleket használ, melyek azonban a rendes HTML-énél sokkal bővebb lehetőségeket teremtenek. Ezek a különleges jelek keveredve vagy beágyazva jelennek meg a HTML alkalmazásban, illetve oldalon.

A ColdFusion kiszolgáló együttműködik a webkiszolgálóval, elfogja ezeket a különleges jeleket, így teszi lehetővé az alkalmazás háttérszolgáltatásait biztosító adatbázissal vagy kiszolgálóval folytatott párbeszédet.

A termékről és a környezetről szerzett benyomásaim rendkívül kedvezőek. Hallatlanul jó felépítésű alaprendszernek tűnik, melyet könnyen lehet telepíteni és használni. A CFML és a HTML közötti hasonlóságok szintén segítik a kezdőket, hogy hamar munkára foghassák alkalmazásukat. Emellett a ColdFusion együttműködik az Apache kiszolgálóval, ami újabb jó pontot jelent. Vannak azonban hátrányai is, például az, hogy a ColdFusion kiszolgáló linuxos változatának rendszerkövetelménye 512 megabájt memória. Ez messze meghaladta az általam használt gép adottságait. A 2000. július 17-én az Allaire honlapjára feltett árlista alapján a ColdFusion Server 4.5 Professional for Linux termék ára 1295 dollár volt. A Windows-alapú fejlesztési környezet további 495 dollárba került. Ha alkalmazásunk költségvetése elviseli a ColdFusion megvásárlásának terhét, megfontolandó választás.

Perl

Azon túl, hogy a Perlt internetes parancsfájlkészítő nyelvként használják, kevés olyan dolog van, amit a Perl nem tud kezelni. A nyílt forrású Perlt *Larry Wall* készítette. Először 1987-ben jelent meg, az 5.6-os számot viselő változata pedig 2000. márciusában. Letölthető a ➲ http://www.perl.org/ honlapról.

A Perl rendkívül jól átgondolt és életképes választás internetes programok készítésére. Fejlesztői közössége hatalmas, és remek támogatás érhető el hozzá. Képes szorosan együttműködni az Apache kiszolgálóval, és elérhető majdnem az összes Linux-változattal. Csakhogy én azt reméltem, hogy valamilyen egyszerűbb csomaggal is meg tudom oldani weblapjaim adatbázishoz kapcsolását. Az, hogy a Perl a legkézenfekvőbb megoldásnak tűnt, még nem jelentette azt, hogy egyben a legjobb is. Az egyszerűség követelménye jelentős szerepet játszott a programnyelvek vizsgálatakor, és ez eltérített a Perl választásától. Alkalmazásomat minél hamarabb, a lehető legrövidebb tanulási időszak után el akartam készíteni.

PHP

A PHP (➲ http://www.php.net/) nyílt forrású, HTML-be ágyazott parancsnyelv. A Perllel ellentétben – amit a rendszerfelügyelet segítésére készítettek –, a PHP-t kezdetektől fogva honlapokkal végzett munkához tervezték. A PHP számos régebbi programozási nyelvből kölcsönöz elemeket, így a Perlből és a C-ből is. A PHP leírása a PHP és a Perl közötti különbségeket is tárgyalja.

A PHP legnagyobb előnye a Perllel szemben éppen az, hogy a PHP-t internetes oldalak parancsfájljainak készítésére tervezték, míg a Perlt számos egyéb feladatra is, ezért meglehetősen bonyolult lett. A Perl rugalmassága és összetettsége folytán használatával könnyen készíthetők olyan kódok, melyeket másik fejlesztő vagy olvasó csak nagyon nehezen ért meg. A PHP megőrzi a rugalmasságot, de formátuma kevésbé zavaró és kötött. Könnyebben együttműködik a meglévő HTML oldalakkal, mint a Perl. A PHP a Perl szinte összes kellemes tulajdonságát magában hordozza, a szerkezetek és a formai követelmények terén egyaránt, ám eközben nem válik olyan bonyolulttá, mint amilyen a Perl lehet.

Hasonlítsuk össze a PHP-t és a ColdFusion-t! A PHP leírása remek összefoglalást ad, és hivatkozik a két csomag *Mike Sheldon* által készített összehasonlítására is. Mike elemzése (➲ http://marc.theaimsgroup.com/) alapos munkának látszik. Kiemeli a ColdFusion néhány olyan erősségét és szolgáltatását, melyeket én nem tárgyaltam. A PHP két, számomra meglepő területen gyengélkedik: a hibák és a dátumok kezelésében.

Az alkalmazás egyszerűsége miatt úgy gondoltam, ezek a hiányosságok számomra nem jelenthetnek különösebb hátrányt. Mindössze egyetlen adatbázist akartam használni, és az alkalmazás esetében minden adatbázishoz meghatározható egy rendeltetéshalmaz.

A ColdFusion által nyújtott adatelvonatkoztatási réteg előnyeinek ez esetben nem láttam hasznát. A végső választásom négy összetevőt tartalmazott: GNU/Linux operációs rendszert, PostgreSQL relációs adatbázis-kezelő rendszert, PHP kiszolgálóoldali internetes parancsnyelvet és az Apache webkiszolgálót.

Összegzés

A Linuxhoz milliónyi programmegoldás érhető el. Azt ajánlom, hogy akik hasonló programcsomag telepítését fontolgatják, alaposan gondolják át, vessék össze a programkínálat alkotóelemeinek erősségeit és fogyatékosságait, melyek abból fakadnak, hogy valamilyen különleges igényt elégítenek ki. A fejlesztők és tanácsadók értéke csak részben származik abból, hogy egy-egy megoldást szinte tökéletesen ismernek. Azoknak a folyamatoknak a megértése, melyek várhatóan hatással vannak a megoldás elkészítésére, elsődleges fontosságú lehet. Mivel rengeteg a kiváló minőségű lehetőség egy-egy számítástechnikai kihívás megválaszolására, a szakértő feladata ezeknek a megoldásoknak a kezelése, ami kulcsfontosságú lehet a teljes tervezet megvalósításának sikere vagy sikertelensége szempontjából.

Saját tervemnél a Linux – Apache– PostgreSQL – PHP összeállítás tűnt megfelelőnek. Ugyanez az összeállítás más tervekhez nem feltétlenül megfelelő. Ennek ellenére remélem, hogy saját választásom szempontjait ismertetve sikerült segítséget nyújtanom néhány olvasó következő tervének megvalósításához és a különféle megoldások vizsgálatához.

Irodalomjegyzék

Castro, E. (2000). HTML for the World Wide Web: Visual Quickstart Guide, fourth edition, Berkley, CA: Peachpit Press.

Kientzle, T.(2000). „Database Engines: MySQL versus Oracle”. Dr. Dobb’s Journal, 25, 7, 98–104.

Loukides, M. and Oram, A.(1997).Programming in GNU Software. Cambridge: O’Reilly. Watson, R. (1999). Data Management: Databases and Organizations, second edition, New York: John Wiley & Sons, Inc.

Dennis Roman Gesker
(drgst30@katz.pitt.edu) jelenleg a Pittsburghi Egyetemen tanul. Az oklevél megszerzése után a montanai Kalispell Alamon Telco cég vezetőségéhez csatlakozik majd.

A Mix-in osztálytípus

Bemutatjuk a Pythonban alkalmazható bekeveredő osztályokat.

A bekeverő programozás olyan programozási stílus, amellyel a szolgáltatásokat osztályokban készítik el, majd pedig más osztályokba belekeverik azokat. Első hallásra talán egyszerű öröklésnek hangzik, de a Mix-inek néhány dologban eltérnek a hagyományos osztályalapú szerkezettől. Gyakran megesik, hogy a Mix-in egyetlen adott osztálynak sem „elsődleges” őse, nem számít, milyen osztályokat használtak fel, ugyanis több, az osztályrangsorban elszórtan elhelyezkedő osztályból épül fel és dinamikusan, futásidőben lesz bevezetve.
Több érv is szól a Mix-inek használata mellett. A létező osztályokat új szolgáltatásokkal bővítik, anélkül, hogy szerkeszteni, karbantartani vagy egybeolvasztani kellene a forráskódjukat. Segítenek a projekt összetevőit (mint a tartomány-keretrendszer, a felület-keretrendszer) elkülönítetten tárolni. Megkönnyítik az új osztályok létrehozását, mert szolgáltatások tárházát nyújtják, melyeket aztán igény szerint lehet összeválogatni. Túllépnek az öröklődés egy korlátján, mivel egy osztály megváltoztatása után is nyugodtan lehet használni az eredeti osztályt a program más részeiben.
A Mix-inek használata nem különleges szakmai lehetősége a Pythonnak, de előnyei kedvéért megéri, hogy vessünk rá egy pillantást.
A Python eszményi nyelv a bekeverő programozáshoz, mivel támogatja a többszörös öröklést, a teljes körű dinamikus kötést (binding) és lehetővé teszi az osztályok dinamikus módosítását.
Mielőtt azonban fejest ugranánk a Pythonba, hadd említsem meg, hogy a Mix-inek nem éppen új keletű dolgok. Ezen a néven elsőként az egykori Talingent projekt megismerésekor láttam, mely a Pink operációs rendszerről és a CommonPoint alkalmazáskörnyezetről volt ismert.
Mivel a C++ nem támogatja a második nyelvi szolgáltatást (teljesen dinamikus kötés), valamint a harmadikat sem (futásidejű dinamikus változtatásokat), nem lennék meglepve, ha ez a megközelítés nem válna be annyira, mint ahogy azt megalkotói eredetileg remélték.

1. lista Mix-in dinamikus telepítése

```
[gt]python
Python 2.0 (#1, Oct 16 2000, 18:10:03)
[GCC 2.95.2 19991024 (release)] on linux2
>>>
>>> class Friendly:
...     def hello(self):
...             print ´Hello´
...
>>> class Person:
...     pass
...
>>> p = Person()
>>> p.hello()
Traceback (most recent call last):
  File "lt;stdin[gt]", line 1, in ?
AttributeError: 'Person' instance has no
➥attribute ´hello´
```

Ismerek egy másik példát is a Mix-in stílusú programozásra, igaz, eltérő név alatt. Az Objective-C rendelkezik egy „kategória” nevű, ötletes nyelvi elemmel, amely lehetővé teszi, hogy tagfüggvényeket adjunk hozzá vagy cseréljünk le meglévő osztályokban akár anélkül, hogy forráskódjukhoz hozzányúlnánk.
Ez nagyszerű lehetőség a létező rendszerosztályok javítására és képességeik bővítésére. Ráadásul a könyvtárak dinamikus betöltésének képességével együtt, a kategóriák igen hatásosak lehetnek az alkalmazások kódszerkezetének fejlesztésében és a kód méretének csökkentésében.
A grapevine arról tájékoztat, hogy a Symbolics’ objektumközpontú Flavors rendszere valószínűleg a Mix-inek legkorábbi komoly megjelenése. A tervezőket egy híres fagylaltozó (Steve’s Ice Cream Parlor) ihlette meg, ahol a vásárlók valamilyen alapjégkrém-változattal kezdtek (vanília, csokoládé stb.), majd ehhez rakhatták hozzá a kiegészítőket ízlés szerint (mogyorót, csokoládéreszeléket, tejszínhabot stb.). A Symbolic rendszerében a nagy önálló osztályok voltak az ízek, míg a kisebb, más osztályok kiegészítésére tervezett osztályok voltak a Mix-inek. További írások olvashatók hálózaton
➲ http://www.kirkrader.com/examples/cpp/mixin.htm oldalon.

A Python képességei

Miközben tiszteletünket fejezzük ki a néhai Taligent, a tetszhalott Objective-C és legendás Symbolics iránt, kezdjünk el kutatni azok közt a képességek közt, melyek a Mix-in programozás terén oly nagyszerű nyelvvé emelik a Pythont.
Először is, a Python támogatja a többszörös öröklést. Ez a Pythonban azt jelenti, hogy egy osztály akár több osztálytól is örökölhet:

```
class Server(Object, Configurable):
    pass
```

Ezenkívül a Python támogatja a teljesen dinamikus kötést. Amikor `obj.load(filename)` típusú üzenetet küldünk egy objektumnak, a Python a megadott üzenet neve és obj osztály öröklési szabályai alapján futásidőben határozza meg, milyen tagfüggvényt kell meghívni. Ez a tulajdonság pontosan úgy működik, ahogy elvárná az ember, és megjegyezni is könnyű. Akkor is működik, ha az osztály öröklési rendje vagy a tagfüggvény megváltozik a futás során.
A többszörös öröklés alkalmazása során a keresési sorrendet mindig szem előtt kell tartani. A keresési sorrend az alaposztályok közt balról jobbra értelmezett, egy adott alaposztályon belül pedig végigvezet a szülőosztályok során. Ha Mix-inekkel dolgozunk, mindig figyeljünk az esetleg előforduló névütközésekre. Ha jól elkülönített Mix-in osztályokat készítünk, és átgondoltan elnevezett tagfüggvényeket használunk, általában elkerülhetjük a meglepetéseket. Végül a Python támogatja az osztályrendszer dinamikus változtatását is.
A legtöbb Python-elem, legyen szó listáról, szótárról, osztályról vagy példányról (instance), rendelkezik elérhető tulajdonságkészlettel.
A Python osztályoknak van egy __bases__ nevű tulajdonsága, amely az adott osztály alaposztályának tárhelye (tuple). A Python tervezésével összhangban, ezt az értéket futásidőben megváltoztathatjuk.
Az első listában látható Python interaktív parancsértelmezőben készí-

2. lista Tetszőleges változó elérésére

```
class NamedValueError(KeyError):
   pass


class _NoDefault:
   pass


class NamedValueAccessible:

   def valueForKey(self, key,
default=_NoDefault):

      ''' A feltételezett kulcs a 'foo'.
          Ez az eljárás az alábbi értékekkel
          tér vissza a következő sorrendben:
             1. Eljárások a nem-eljárások
                előtt
             2. Nyilvános tulajdonságok,
                a nem nyilvánosak előtt

          Még pontosabban, ez az eljárás a
          következők egyikével tér vissza:
             * self.foo()
             * self._foo()
             * self.foo
             * self._foo

          ...vagy az alapértelmezéssel, ha van
          ilyen, egyébként kivétel.
      '''
      assert key
      klass    = self.__class__
      underKey = '_' + key
      attr     = None
      method   = getattr(klass, key, None)
      if not method:
         method = getattr(klass, underKey,
                  None)
         if not method:
            attr = getattr(self, key, None)
            if not attr:
                attr = getattr(self,
                       underKey, None)
                if not attr:
                    if default!=_NoDefault:
                       return default
                    else:
                       raise
                       NamedValueError, key
      if method:
         return method(self)
      if attr:
         return attr
```

tett példában két osztályt készítünk, majd megváltoztatjuk az öröklési viszonyokat. Az *1. listában* látható People (személy) nem túl barátságos, változtassuk hát meg. Sőt, változtassunk meg minden személyt, egyszer és mindenkorra megoldva a gondot:

```
>>> Person.__bases__ += (Friendly,)
>>> p.hello()
Hello
```

A fenti első utasítás a Person alaposztályát változtatja meg. A += használatával (az egyszerű „=” használata helyett) elkerülhetjük, hogy véletlenül töröljünk létező alaposztályokat, különösképpen, ha a kód későbbi változatában a Person más osztályoktól is örököl. A mókás kinézetű `(Friendly,)` kifejezés tuple-t határoz meg, amit általában zárójelek közé kell tenni. Igen ám, de míg a Python az (x,y) jelölést önműködően két elem tuple-jének veszi, az (x) jelölést zárójeles kifejezésként ismeri fel. A vessző hozzáadása a tuple-ként való felismerést kényszeríti ki.

MySQLdb sormutató Mix-in

A Mix-inek készítésének magától értetődő módja, ha a tervezés közben, modul készítésekor adjuk meg azokat. Az egyik legkedveltebb külső fejlesztésű Python modul, a MySQLdb, pontosan ezt teszi. A Python az adatbázis-elérésekhez egy DB API nevű szabványos csatolófelületet ➲ http://www.python.org/topics/database/) határoz meg. *Andy Dustman* MySQLdb modulja ezt a csatolófelületet használja, így ezen keresztül a Python-programozók kapcsolatokat hozhatnak létre, illetve lekérdezéseket küldhetnek a MySQL kiszolgálónak. Megtalálható a ➲ http://dustman.net/andy/python/MySQLdb/ címen. A MySQLdb három fontosabb szolgáltatást nyújt az általa készített sormutató objektumhoz. Ha szükséges, figyelmeztetéseket küld, igény szerint tárolja az eredménykészletet az ügyféloldalon, vagy használja őket a kiszolgálóoldalon, végül az eredményt tuple-ként (például rögzített listákként) vagy szótárként adja vissza. Ahelyett, hogy mindezt egyetlen hatalmas osztályba tömörítené, a MySQLdb mindegyikükhöz egy-egy Mix-in osztályt rendel:

```
class CursorWarningMixIn:
class CursorStoreResultMixIn:
class CursorUseResultMixIn
class CursorTupleRowsMixIn:
class CursorDictRowsMixIn(CursorTupleRowsMixIn):
```

Ne feledjük, a Mix-inek osztályok, és mint ilyenek, kihasználhatják az öröklődés minden előnyét, amint azt a CursorTupleRowsMixIn-től öröklő CursorDictRowsMixIn esetében meg is figyelhetjük. A fenti Mix-inek egyike sem állhat önmagában: a BaseCursor osztály bármely típusú sormutatóhoz szükséges alapszolgáltatásokat biztosítja. A fenti Mix-inek és a BaseCursor összekapcsolásával a figyelmeztetések (warnings), a tároló- és eredménytípusok (result types) minden elképzelhető változatát előállíthatjuk (összesen nyolcfélét). Amikor létrehozzuk az adatbázis-kapcsolatot, a kívánt sormutatótípust adhatjuk át:

```
conn = MySQLdb.connection
➥(cursorclass=MySQLdb.DictCursor)
```

A Mix-inek nemcsak a MySQLdb program megalkotásakor segítettek, hanem könnyebben bővíthetővé is teszik, mert a saját testreszabott sormutató osztályainkhoz egyszerűen kiválasztható képességeket fűzhetünk. Figyeljük meg, hogy az osztálynevek végéhez a Mix-in szócskát szoktuk illeszteni, ami az adott osztály természetéről árulkodik. Másik általános szokás, hogy az „-able” vagy „-ible” végződést fűzzük a névhez, amint az a Configurable vagy NamedValueAccessible esetén is megfigyelhető.

NamedValueAccessible

Használjuk fel példánkban ez utóbbit. A NamedValueAccessible Mix-In a valueForKey( ) tagfüggvény képességével ruház fel minden

4. lista Mix-in függvényünk végső változata

```
import types

def MixIn(pyClass, mixInClass, makeAncestor=0):
    if makeAncestor:
      if mixInClass not in pyClass.__bases__:
         pyClass.__bases__ = (mixInClass,)
➥+ pyClass.__bases__
    else:
      #Rekurzívan letiltja a mix-in elődosztályt,
      #hogy támogassa az öröklődést
      baseClasses = list(mixInClass.__bases__)
      baseClasses.reverse()
      for baseClass in baseClasses:
         MixIn(pyClass, baseClass)

      # beillesztjük a mix-in eljárást az
      # osztályba
      for name in dir(mixInClass):
            if not name.startswith('__'):
      #átugorjuk a titkos tagokat
            member = getattr(mixInClass, name)
            if type(member) is
types.MethodType:
                member = member.im_func
            setattr(pyClass, name, member)
```

osztályt, amelybe bekeverjük. Az obj.valueForKey(name) hívás esetén a tagfüggvény a következők egyikét fogja visszaadni:

- obj.name( )
- obj._name( )
- obj.name
- obj._name

Másként fogalmazva: a valueForKey( ) tagfüggvényt vagy tulajdonságot keres – legyen az belső vagy nyilvános –, hogy az adott kulcshoz tartozó értéket visszaadhassa. E tagfüggvény tervezése azt tükrözi, hogy a Python-objektumok esetében gyakran tulajdonságokon és tagfüggvényeken keresztül is elérhető ugyanaz az adat. A megvalósítást lásd a *2. listában*.
Egyik hasznos tulajdonsága e Mix-innek, hogy általános kódot tartalmaz a naplózáshoz (lásd a *3. listát*). Egyszerűen csak új kulcsokat kell a logColumns( ) tagfüggvényhez adni, és a napló, illetve az azt készítő kód (amely a logEntry() részben található) megváltoztatása nélkül kibővíthető. Talán még érdekesebb, hogy a logColumns( ) akár egy egyszerű beállításfájlból is kiolvashatja a lista mezőit.
A rugalmas valueForKey( ) tagfüggvény jóvoltából, a tranzakcióobjektum kötetlenül képes a kért értékeket tagfüggvényeken vagy tulajdonságokon keresztül szolgáltatni. A Mix-inek rugalmas készítése nagymértékben növeli használhatóságukat, és akár művészetnek is nevezhetjük megalkotásukat.

Utólagos keverés

Eddig példákat láthattunk arra, miképpen lehet a Mix-ineket az osztályok készítése közben használni. Csakhogy a Python azt is lehetővé teszi számunkra, hogy a szolgáltatásokat futásidőben keverjük be! Legegyszerűbb megoldás, ha megváltoztatjuk az adott osztály alaposztályait, mint azt korábban már leírtuk. Egy függvény felhasználásával lehetőség nyílik arra, hogy ezt a műveletet átlátszóan végezzük, és később – amennyiben szükséges – továbbfejlesszük:

```
def MixIn(pyClass, mixInClass):
    pyClass.__bases__ += mixInClass
```

Nézzünk egy olyan helyzetet, amely nyilvánvalóvá teszi a MixIn( ) módszer létjogosultságát. Egy internetes alkalmazás készítésekor, a tartományosztályok és a felületosztályok szétválasztása általában jó ötlet. A tartományosztályok képviselik az adott alkalmazás fogalmait, adatait és műveleteit, valamint függetlenek az operációs rendszertől, a felhasználói felülettől, az adatbázistól stb. Néhány szerző üzleti objektumként (business object), modellobjektumként vagy lényegi objektumként hivatkozik a tartományobjektumokra.
A tartomány és a felület elválasztása több okból is hasznos lehet. A célkitűzés általában két kulcsterületre bontható, amelyek nagyjából függetlenek egymástól. Mi a nehézség tárgya? Illetve, hogyan jelenítsük ezt meg? Új felületeket készíthetünk a tartományobjektumok megváltoztatása vagy újraírása nélkül is. Tulajdonképpen akár több felületet is készíthetünk.
Egy regénykiadó rendszer tartomány objektumai lehetnének például a Regény, a Szerző és a Székhely. Ezek az osztályok létfontosságú tulajdonságokat (például cím, szöveg, név, e-mail stb.) és sok műveletet hordoznak (elment, betölt, kiadás stb).
Egy ilyen rendszerhez az egyik felület lehet egy honlap, ami lehetővé teszi a felhasználók számára, hogy regényeket készítsenek, szerkesszenek, töröljenek és kiadjanak. Mikor ilyen honlapot fejlesztünk, hasznos lenne, ha a tartományosztályaink, mint például a Regény, rendelkeznének olyan tagfüggvényekkel, mint a renderView( ) vagy a renderForm( ), amelyek megjelenítik, vagy szerkesztésre felkínálják a regényt HTML formában.
Mix-inek használatával ezeket a szolgáltatásokat is a tartományosztályokon kívül fejleszthetjük.

```
StoryInterface osztály:
    def renderView(self):
        # A regény kiírása HTML formátumban
        pass
    def renderForm(self):
        # A regény megjelenítése szerkesztésre
        pass
```

A honlapot működtető kódba pedig keverjük be őket:

```
from MixIn import MixIn
from Domain.Story import Story
MixIn(Story, StoryInterface)
```

Ha úgy döntünk, hogy grafikus felületet készítünk a kiadórendszerhez, nem kell a teljes HTML szerkezetet magunkkal cipelnünk (vagy fordítva). A tartományosztályok a szükséges adatokra és műveletekre összpontosítanak, hogy a GUI fejlesztésekor pontosan azt kapjuk, amire szükségünk van.
Mondhatná valaki, hogy miért ne készíthetnénk egy új osztályt, hogy összehozzuk a kettőt:

```
class StoryInterface:
    ...
from Domain.Story import Story
class Story(Story, StoryInterface): pass
```

Vagy mondhatná azt, hogy a StoryInterface miért ne lehetne a Regény alosztálya, melynek megszerezi a képességeit. Vizsgáljuk

meg azt az esetet, ahol a Regény osztálynak már eleve voltak más tartományalosztályai:

```
class Story: ...
class Editorial(Story): ...
class Feature(Story): ...
class Column(Story): ...
```

A Regény létező alosztályaira semmilyen módon nincsen hatással az új Regény osztály vagy alosztály készítése. A dinamikus Mix-in készítése azonban egyaránt hat a további osztályokra is. Ez az, ami miatt az állandó megoldás sokszor nem működik a gyakorlatban, és ami a dinamikus megoldást nemcsak egyszerűen ügyes, hanem egyenesen szükséges megoldássá teszi.
Továbbá, vizsgáljuk meg azt az esetet, amikor a Regény objektumok a kód olyan részében készülnek, ahol a Regény közvetlenül kódolva van. Habár gyenge megoldás, mégis meglehetősen általános gyakorlat. Ebben az esetben a Regény alosztályok készítésének semmiféle hatása nem lenne azon a kódon, ami figyelmen kívül hagyja őket.
Egy figyelmeztetés a dinamikus Mix-inek használata kapcsán: megváltoztathatják a létező objektumok viselkedését (mivel megváltoztatják az objektumok osztályát is). Ez pedig váratlan eredményekre vezethet, mivel a legtöbb osztályt nem az ilyen típusú változások szem előtt tartásával tervezik meg. A dinamikus Mix-inek használatának biztonságos módja, hogy még az objektumok készítése előtt, az alkalmazás indításakor telepítjük őket.

A MixIn( ) továbbfejlesztett változatai

Az első fejlesztés, amit a MixIn( ) függvényhez adhatunk, hogy ellenőrizzük, ne keverjük be kétszer ugyanazt az osztályt:

```
def MixIn(pyClass, mixInClass):
    if mixInClass not in pyClass.__bases__
        pyClass.__bases__ += (mixInClass,)
```

A gyakorlatban többnyire arra van szükségünk, hogy a Mix-in tagfüggvények kapják az elsőbbséget, s akár helyettesítsék is az örökölt tagfüggvényeket amennyiben szükséges. A függvény következő változata a keverék osztályt az alaposztályok sorozatának elejére helyezi, de lehetőséget nyújt arra is, hogy egy tetszés szerint megadható változóval megváltoztassuk ezt a viselkedést:

```
def MixIn(pyClass, mixInClass, makeLast=0):
  if mixInClass not in pyClass.__bases__
    if makeLast:
      pyClass.__bases__ += (mixInClass,)
    else:
      pyClass.__bases__ = (mixInClass,) +
      ➥pyClass.__bases__
```

A Python hívások olvashatóságáért a jelzőkhöz ajánlott a nevük használata:

```
# nem túl olvasható:
MixIn(Story, StoryInterface, 1)
# Lényegesen jobb:
MixIn(Story, StoryInterface, makeLast=1)
```

Ez az új változat még mindig nem teszi lehetővé, hogy az éppen használt osztály tagfüggvényeit a Mix-in tagfüggvényei felülírják, ezért a Mix-in tagfüggvényeket tulajdonképpen az osztályba kell telepíteni. Szerencsére a Python elég dinamikus ahhoz, hogy ezt megtehessük. A *4. lista* bemutatja MixIn( ) függvényünk végső változatának forráskódját. Alapértelmezés szerint a Mix-in tagfüggvényeit közvetlenül a célosztályba telepíti, miközben figyel a Mix-in alaposztályainak átvitelére is. A hívás kinézete:

MixIn (Story, StoryInterface)

A kiegészítő `makeAncestor=1` érték segítségével olyan MixIn( ) függvényt kapunk, ami az eredeti elvet követve működik (azaz a Mix-int a célosztály szülőjévé teszi). Azt a képességet, ami a Mix-int az alaposztályok végére helyezi, eltávolítottuk, mivel a gyakorlatban szinte soha nem volt rá szükségünk.
A függvény még kifinomultabb változata visszaadhatná (esetleg választhatóan) a két osztály között ütköző tagfüggvények listáját, vagy ha átfedést talál, meghívhatna egy kivételt ezzel a listával.

Önműködő Mix-in telepítés

Az utólagos keverés komoly használata esetén a MixIn( ) függvény hívása többszörözötté válhat. Például egy GUI alkalmazásnak minden létező tartományosztályhoz lehet keveréke, így valami ilyesmi hívás szükséges mindegyikhez:

```
from Domain.User import User
MixIn(User, UserMixIn)
```

Az egyik megoldás a Mix-inek és a célosztály név alapján történő összekapcsolása és telepítése az alkalmazás indításakor. Például az összes Mix-int elnevezhetjük közvetlenül az után az osztály után, amit módosít és betehetjük a MixIns/ könyvtárba.

Egyéb lehetőségek

Bár mókás dolog egyre kifinomultabb MixIn( ) változatokat fejleszteni, a legfontosabb dolog mégis az, hogy a programfejlesztésben fel tudjuk őket használni. Ösztönzésképpen álljon itt néhány ötlet:

- Az osztály kibővítheti saját magát egy Mix-innel, miután olvas a beállításfájlból. Például egy webkiszolgáló osztály bekeverhet egy Threading vagy Forking osztályt, a beállítástól függően, miként lett beállítva.
- A program bővítményeket is használhat: olyan programcsomagokat, melyeket futásidőben keres meg és tölt be a program, hogy kiterjessze a képességeit. Azok, akik bővítményeket készítenek, kihasználhatják a MixIn( ) függvényt, ezáltal gyarapítva az eredeti programosztályokat.

Összefoglalás

A bekeveredők nagyszerű eszközök a részekre bontott fejlesztéskor és a már létező osztályok kiegészítésére anélkül, hogy azok forráskódjához hozzá kellene nyúlni. Ez elősegíti az olyan programozói elvek betartását is, mint a tartomány és a felület szétválasztása, dinamikus beállítás és a bővítmények használata. Ha Pythont használunk, mindig gondoljuk át, milyen Mix-inek tehetnék még hatékonyabbá a programunkat.

Chuck Esterbrook
(echuck@mindspring.com) tanácsadó, író és vállalkozó, lelkes Python- és Webwarefelhasználó.

Kapcsolódó címek

- ➲ http://www.python.org/
- ➲ http://webware.sourceforge.net/

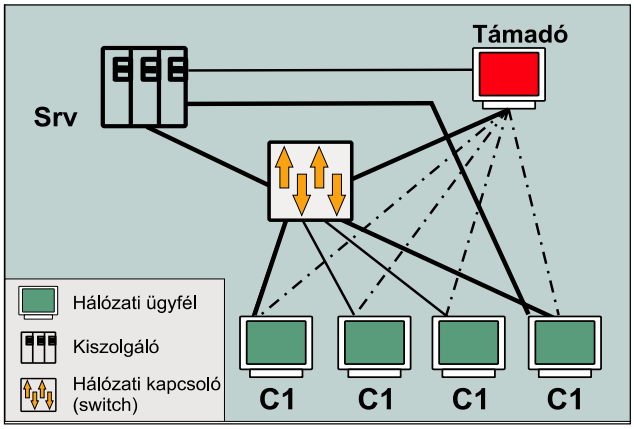
Könnyű álom (5. rész)

Behálózva

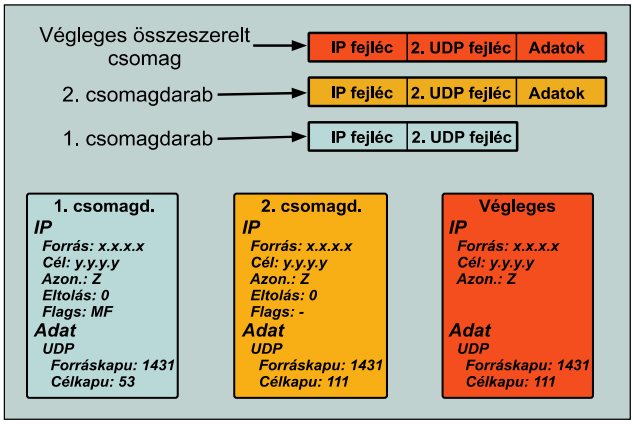
Ecikkben a hálózatba kötött gépeinket fenyegető veszélyeket és olyan megoldásokat javaslunk, amivel ezek kockázata csökkenthető. A különböző támadásoknak különböző előfeltételeik vannak (például a támadó a célgéppel egy hálózati szakaszon helyezkedik el). Ennek függvényében kell meghatároznunk, hogy milyen támadásokkal szemben védjük gépünket.
A helyi hálózatról gépeinket számos támadás érheti. A támadások kockázata nagymértékben eltér attól függően, hogy a rendszerünk egy otthoni hálózat, kis cég irodája vagy banki rendszer. Míg az otthoni rendszereknél az ilyen támadás esélye roppant kicsi – kivéve, ha az egyik gépünkön egy trójai csücsül –, addig egy banki rendszer esetében fel kell készülnünk minden lehetséges támadásra.

Lehallgatás

A legegyszerűbb támadás a hallgatózás (snooping). A támadást az teszi lehetővé, hogy a napjainkban elterjedt ethernethálózat esetén egy gép adását a hálózat többi gépe is megkaphatja. A veszély mértékét nagyban befolyásolja a hálózat felépítése. A továbblépéshez először tisztáznunk kell, hogy milyen eszközöket használhatunk a hálózatunk felépítéséhez. A legegyszerűbb eszközök az úgynevezett jelelosztók (hub) és jelismétlők (repeater), melyek egyszerűen csak összekötik a gépeket, forgalomirányítást nem tesznek lehetővé. Számunkra mindkét eszközfajta ugyanazt a szolgáltatást nyújtja, így a későbbiekben bármelyikre hivatkozunk, az vonatkozik a másikra is. Sokan ezeket részesítik előnyben, hiszen áruk igen alacsony.
A jelelosztóknál magasabb szintű szolgáltatást nyújtanak a hidak (bridge), valamint a kapcsolók (switch). Ezek az OSI modell [1.] második szintjén elhelyezkedő eszközök. Itt már kettes szintű (OSI modell adatkapcsolati rétegében zajló) forgalomirányításra nyílik lehetőség. Ha például a kapcsoló egyik csatlakozóján helyezkedik el A és B gép, valamint egy másik csatlakozóján C gép, akkor az eszköz C felé általában nem továbbítja A és B beszélgetését. Az irányítás ebben az esetben az OSI kettes szintű címzés (ethernet esetén például Ethernet-cím) alapján történik (a cikkben végig ethernet hálózatot tételezünk fel, így ha az egyszerűség kedvéért ethernet-címről beszélünk, erre gondoltunk). A kapcsolókat a hidak különleges eseténekt tekinthetjük, így szintén bármelyiket említjük is, vonatkozik mindkét eszközre.
A harmadik osztályt az útválasztó eszközök (router) képezik. Ezen eszközök az OSI modell 3. szintjén helyezkednek el, az ennél alacsonyabb szinten zajló forgalmat közvetlenül nem engedik át (a cikkben feltételezzük, hogy csak IP-alhálózataink vannak).
Amennyiben az egész hálózatot pusztán jelelosztók kapcsolják össze, akkor az egész hálózati forgalom lehallgatható. A lehallgatás útválasztókon át nem lehetséges. A hálózatban elhelyezkedő kapcsolók alapesetben a lehallgatást megakadályozzák, bár bizonyos támadási módszerek felhasználásával a kapcsolók által nyújtott védelem is kijátszható. Hogyan lehetséges ez? Két elterjedt módszer is létezik: az első módszerrel magukat a kapcsolókat támadhatják, a második támadástípus segítségével az egyes gépek IP protokollrétegei ellen indítható támadás. A kapcsolók támadása esetén a behatoló két dologra törekedhet: vagy felügyelői jogokat akar szerezni az eszközön vagy annak megvalósítási, illetve beállítási hibáit próbálja meg kihasználni. Amennyiben egy kapcsolót a telepítés után nem állítottunk be megfelelően, úgy lehetséges, hogy megmaradtak a gyári jelszavak. A különböző kapcsolókban elhelyezett hátsó ajtók (backdoors) ugyancsak veszélyt jelenthetnek. Ezeket a rövidlátó gyártók karbantartási célokra építgetik eszközeikbe a rendszergazda emlékezetkiesésének esetére. Szintén gondot jelenthet a gyárilag engedélyezett SNMP protokolltámogatás, amit elsősorban az eszköz távoli felügyeletéhez használnak. Ennek segítségével az eszköz beállítása letölthető, bizonyos esetekben akár módosítható is. Egyes eszközök tartalmazhatnak olyan biztonsági hibákat is, hogy a gyárilag engedélyezett, csak olvasási jogot biztosító hozzáférés esetén is hozzáférhetők a rendszergazdai jelszavak vagy a módosítást is lehetővé tevő SNMP-azonosítók. Emiatt soha ne felejtsük el a hálózatba kötött kapcsolók jelszavait lecserélni, az SNMP-hozzáférést pedig letiltani, vagy a szükségesre korlátozni. Az SNMP-hozzáférést engedélyező azonosítókat mindig cseréljük le! Látogassuk rendszeresen a gyártó weboldalait, és a vezérlőprogram (firmware) ajánlott javításait (különösen a biztonsági javításokat) telepítsük fel az eszközeinkre. Ez a lépés rendkívül gyakran elmarad, hiszen a kapcsolók a hálózatra kapcsolás után azonnal működnek, látszólag nem igényelnek különösebb beállítást.
A kapcsolók elleni támadás másik módja, hogy megvalósítási vagy felépítésbeli hibáikat próbálják meg kihasználni. Ehhez meg kell ismerni az alapvető működésüket (jelen leírás elég erősen leegyszerűsített). A kapcsoló veszi a csatlakozóin beérkező kereteket (frames), majd megvizsgálja azok ethernet forráscímét. Amennyiben az adott forrás nem, vagy másik csatlakozón szerepel a belső irányító táblázataiban, akkor a táblázatát módosítja. Ezután a célcím vizsgálata következik. Ha a célcím csoportcím vagy üzenetszórásos cím, azokat kiküldi az összes egyéb csatlakozóján. Egyéb esetben a célcímet megkeresi a belső irányítási tábláiban. Sikeres keresés esetén a megadott csatlakozóra, egyéb esetben az összes egyéb csatlakozóra továbbítja. Ezt a működési módot hívjuk a kapcsoló áttetsző (transparent), vagy más néven öntanuló módjának. Ebből is látható, hogy ha a kapcsolót hamisított ethernet forráscímű keretekkel bombázzák, akkor hozzáférhetnek más gépek kereteihez. Előfordulhat, hogy megpróbálják a kapcsoló belső irányítási tábláit megtölteni. Ha a tábla betelik, a kapcsoló bejegyzéseket dobál ki belőle. Ezzel elérhetik, hogy a tábla esetleg hamis címekkel lesz tele, a valódi hálózati forgalmat így kénytelen minden csatornán kiküldeni. Ugyancsak lehetséges, hogy a kapcsoló a processzorának túlterhelése esetén a kereteket nem szűri, hanem minden csatlakozójára továbbítja. Ezek támadási lehetőségek elég erősen gyártó- és megvalósításfüggőek, de elméleti lehetőségként célszerű tisztában lennünk vele. Amennyiben az eszközünk képes rá, a jó nevű gyártók eszközei általában tudják ezt, célszerű lehet az öntanuló módot részben – legalább a kényes gépekre – vagy egészen kikapcsolni. Célszerű korlátozni az engedélyezett forráscímek körét. Így csökkenthetjük annak a lehetőségét, a támadó más gép címével vagy hamis forráscímmel kereteket juttathasson a hálózatba. Amennyiben lehetőség van rá, célszerű a kapcsoló naplózását bekapcsolni, a naplóbejegyzéseket pedig valamelyik kiszolgálón gyűjteni és feldolgozni. A jobb minőségű kapcsolók (általában a nevesebbek) támogatnak egy újabb szolgáltatást is, ez a VLAN (Virtual Local Area Network). A VLAN-ok célja, hogy



1. ábra APR címhamisítás



2. ábra IP-darabolási/összeszerelési támadás

a hálózatot virtuálisan nem érintkező részekre oszthassuk. A szolgáltatás létrehozásának a célja az volt, hogy költséghatékony módon kapcsolhassuk össze számítógépeinket, valamint csökkenthessük az ütközési zónákat (collision domains). A keretek nem hagyhatják el a VLAN-t, így például a csoportcímzett keretek is a VLAN-on belül maradnak. A VLAN létrehozása természetesen nem csak egy kapcsolón belül lehetséges, több kapcsoló közösen is kifeszíthet egy VLAN-t. Ilyenkor van a VLAN-oknak igazán értelme – például a cég ugyanazon szervezeti egységei több épületben vannak –, hiszen fizikailag egy vezetéken vihetjük át több különálló hálózat forgalmát. A kapcsolók ennek megvalósításához különleges keretekkel tartanak kapcsolatot, ahol a keret tartalmazza a VLAN azonosítóját. Amennyiben a kapcsolóinkban nem szabályozzuk e keretek elfogadását, akkor a támadó képes a VLAN-ok közötti határ átlépésére. Ha VLAN-okat használunk, mindenképpen tegyünk meg minden lehetséges óvintézkedést. Semmiképpen ne feledjük, hogy a VLAN-ok biztonsági felhasználásával (például: két VLAN között tűzfalazunk) további kockázatot vállalunk, hiszen a kapcsoló feltörése vagy a vezérlőprogram hibája miatt a VLAN-ok átjárhatóvá válhatnak. Az előbbiek figyelembevételével a kapcsolók beállíthatók úgy, hogy a kifejezetten ellenük indított támadások ne járhassanak eredménnyel. Fontos azonban megjegyeznünk, hogy a kapcsolók nem biztonsági eszközök. Tervezéskor a fő cél a teljesítmény növelése és nem a biztonság fokozása volt. A rendszer biztonságát pusztán a kapcsolókra alapozni botorság (például: teljesen kapcsolt hálón Telnet protokoll használata, mondván úgysem tudják lehallgatni).

A másik lehetséges támadási módszernél a támadó úgy kísérel meg csomagokhoz jutni, hogy a feladó gépek IP protokoll rétegét próbálja megtéveszteni. Ehhez az ARP (Address Resolution Protocol) [4.] támadhatóságát használja ki, így a részletes magyarázat előtt nézzük át ezt. Cikksorozatunk harmadik részében [2., 3.] már megismertük az IP protokoll alapvető működését, valamint az általa használt csomagtípusokat és azok szerkezetét. Akkor azonban nem szóltunk arról, hogy az IP-címek miként is azonosítják a számítógépeket. Ethernet-hálózaton a címzés az úgynevezett ethernetcímen alapul. Minden hálózati kártyának saját ethernetcíme van. Az IP-cím alapján az ethernetcím meghatározására szolgál az ARP. Amennyiben az Alfa nevű gép csomagot kíván küldeni Béta gép számára, és Béta ethernetcímét nem ismeri, úgy a hálózatra egy üzenetszórt (broadcast) ARP kérést küld ki, amelyre Béta válaszol, vagy egy harmadik gép (Ubul), amely ismeri Béta címét (proxy arp). A választ Alfa elhelyezi a saját ARP-gyorstárában (ARP cache). A protokoll lehetőséget ad arra is, hogy egy gép bejelenthesse az ethernetcím változását. Ezt a lehetőséget *Gratuitous ARP*-nak hívják [4., 2.].

Most térjünk vissza az eredeti kérdéshez, azaz mit tehet egy támadó az ARP segítségével. A hálózat egy kiszolgálóból (Srv) és több ügyfélből (C*x*) áll. A támadó elkezdi bombázni ARP-válaszokkal és *Gratuitous ARP* csomagokkal az ügyfeleket, amelyben saját ethernetcíméhez Srv IP-címét adja meg. Amennyiben az ügyfelek ARP-gyorstárát sikerül így módosítania, a C4 ügyfél az Srv felé menő csomagját valójában a támadónak küldi. Tehát a támadó képes hozzájutni az ügyfelektől a gazdagép felé menő teljes forgalomhoz. Ugyanezt a trükköt be lehet vetni a gazdagép ellen is. Így az ellenirányú forgalom is lehallgatható. A példa az 1. ábrán látható. Ezek ellen úgy védekezhetünk, ha a fontosabb gépekre az ARP-gyorstárban állandó bejegyzéseket helyezünk el, vagy a teljes ARP-t letiltjuk gépeinken. Ez azonban nem minden operációs rendszernél lehetséges, hiszen az ARP-gyorstár mérete általában korlátozott. Miért ilyen fontos a forgalom lehallgatása? Egyrészt a támadó adatokat nyer a rendszerből (kiszolgálók, ügyfelek, útválasztók), valamint érzékeny adatokat is megszerezhet (pl.: jelszavak). Mint a későbbiekben látni fogjuk, néhány támadásnál fontos szerep jut annak, hogy a támadó képes-e figyelni az ügyfél vagy a gazdagép forgalmát.

IP-cím hamisítása, kapcsolat eltérítése

Az eddigi támadásoknál áttekintettük, hogy mit érhet el egy támadó, illetve mi ellen kell védekeznünk legfeljebb kapcsolókkal összekapcsolt hálózaton. Ezen támadások ellen védelmet jelentenek az útválasztók, hiszen sem az adatkapcsolati réteg szintű támadások sem az ARP-támadások nem jutnak át az útválasztókon, hiszen azok az OSI modell 3. szintjén helyezkednek el. Azonban itt is követhetünk el beállítási hibákat, amelyek újabb támadásokat tesznek lehetővé. Ismeretek birtokában hozzáláthatunk az újabb lehetséges támadási módszerek megismeréséhez: ezek az IP-címhamisítás (spoofing) és IP-eltérítés (hijacking). Azt a támadást nevezzük IP-címhamisításnak, amikor a támadó nem a saját címét, hanem tetszőleges más címet használ. A használt cím lehet létező gép címe, vagy egy eddig használaton kívüli. A célja lehet a támadóra utaló IP-cím elrejtése, vagy jogosulatlan előnyhöz jutás, például a kiszemelt célgépre csak a 10.6.75.43 címről lehet belépni. Ezeknél a támadásoknál a kalóz gépe nem kap közvetlen választ a hálózatról, hiszen a válasz a hamisított címre érkezne. Feltételezzük, hogy a támadó a válaszhoz mégis hozzájuthat ARP címhamisítás vagy a hálózati forgalom lehallgatása folytán. Amennyiben a támadó a válaszhoz nem fér hozzá, akkor a módszert vak hamisításnak (blind spoofing), vagy vak kapcsolat-eltérítésnek (blind session hijacking) nevezzük. Ennek kivitelezése lényegesen bonyolultabb, erről a későbbiekben ejtünk szót. UDP esetén az IP-címhamisítás roppant egyszerű: a támadó hamisított forráscímű csomagokat juttat a hálózatba. Bizonyos protokollok támadásához több lépcső is szükséges lehet, ekkor a válaszra is

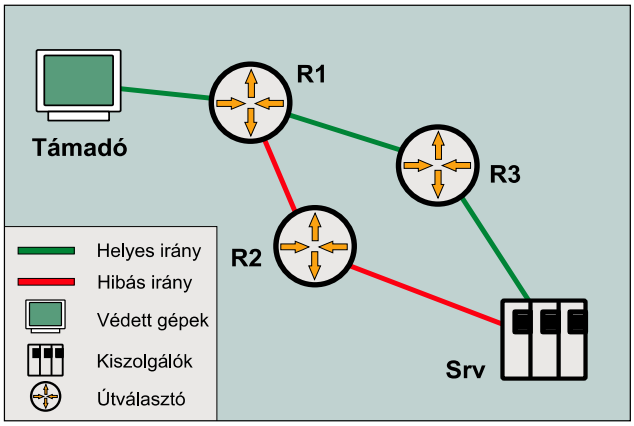
szükség van. A TCP-kapcsolatoknál általában szükség van a visszajövő forgalomra is, az ellenkező esettel a vak hamisításnál foglalkozunk. Mint azt harmadik cikkünkben már leírtuk, a TCP-kapcsolat felépítése háromlépcsős. A kezdeményező gép egy SYN-es csomagot küld a kiszolgálónak, amely erre egy SYN+ACK csomaggal válaszol. A csomag vétele után a kezdeményező egy ACK csomagot juttat a kiszolgálónak, amelyben a SYN+ACK sorozatszámánál eggyel nagyobb nyugtasorszámot használ. A kapcsolat sikeres felépítéséhez mindenképpen a jó nyugtát kell használni. Amennyiben a kiszolgáló gép kezdősorszám-előállítója (ISN – Initial Sequence Number) megfelelő, a SYN+ACK csomagra mindenképpen szükség lesz. A kapcsolateltérítés az IP-címhamisítás egy részesete. Célja, hogy a támadó egy már nyitott kapcsolatba avatkozzon be. Álljon itt erre egy egyszerű példa. A rendszergazda telnet protokollon bejelentkezett a távoli rendszerre. Tételezzük fel, hogy egyszer használatos jelszavakat használ (One Time Password – OTP), például SKEY rendszert. Ebben az esetben a támadó nem ér semmit a forgalom lehallgatásával. Ekkor a támadó megkísérelheti eltéríteni (elrabolni) a rendszergazda kapcsolatát, így az ő nevében tevékenykedhet. Ehhez figyelni kell a hálózati forgalmat, hiszen a kapcsolat eltérítéséhez ismerni kell mind az időszerű csomagsorszámot, mind a nyugta sorszámát. Ezután a következő csomagot már a támadó küldheti el. Amennyiben sikerül, az eredeti rendszergazda kiesik a szinkronból. A támadónak még egy feladata van: lehetetlenné kell tennie, hogy az eltérített munkaállomás érzékelje ezt a hibát és lezárja a kapcsolatot. Ez a feladat az IP-címhamisításnál is fennáll, amennyiben a támadó működő gép címét veszi át. Ezen támadások megvalósítására az Interneten többféle eszköz is rendelkezésre áll, a támadónak pusztán csak le kell töltenie, lefordítania és máris kezdheti „áldásos” tevékenységét.

Vak támadásnak nevezzük azt, ha a támadó nem jut hozzá a számára oly fontos válaszcsomagokhoz. Ezt a támadási formát jóval nehezebb sikeresen kivitelezni és általában könnyebb is védekezni ellene. UDP esetén a vak támadás egyszerű lehet, ha a válaszcsomagok nem szükségesek. TCP esetén a helyzet jóval bonyolultabb, mivel ilyenkor ismerni kell a megfelelő nyugtasorszámot. Ez a feladat megfelelő minőségű IP-alrendszerek esetében szinte lehetetlen. Néhány operációs rendszernél különböző megvalósítási hibák miatt ez a támadás sajnos kivitelezhető. Ilyen hiba volt például a Linux rendszermag 2.0.35-nél korábbi változataiban is [9.]. Itt bizonyos esetekben a rendszermag nem ellenőrizte a nyugtasorszámot, így a vak IP-címhamisítás kivitelezése egyszerű volt. A vak IP-címhamisítás kivitelezhetőségét leggyakrabban a rossz minőségű TCP kezdősorszám-előállító teszi lehetővé. Ez hagyományosan nem biztonsági célokat szolgált, így egyszerű eljárásokkal jól meg lehetett határozni a használandó értéket (eredetileg a hálózaton bolyongó vagy többszöröződött kapcsolatfelépítési kérések kiszűrése volt csak a célja). Elég sok kereskedelmi rendszernek nincs megfelelő minőségű TCP kezdősorszám-előállítója. Akit a téma részletesebben érdekel, kezdésként olvassa el az nmap írójának cikkét [11.]. Egy gyakorlatban is végrehajtott támadás részletes elemzése olvasható *Tsutomu Shimomura* cikkében [8.].

Ugyancsak gondot jelenthet, ha a támadó valamilyen eljárással képes behatárolni a kapcsolat által jelenleg használt sorszám tartományát. A Linux rendszermagban ilyenre is volt példa, ez a 2.0.37-es változatot is érintette [10.]. Ebben az esetben az IP-alrendszer a valódi (várt) sorszám és a hamisított sorszám távolságától függően, eltérően viselkedett. Mivel az IP-azonosítókat a rendszer sorrendben osztotta, egy gyengén terhelt gépen jól be lehetett határolni a szükséges sorszámot.

Egyéb IP-szintű támadások

Az IP-alapú protokollok nem csak a fenti módokon támadhatók. Az IP összetettségének következtében elég sok megvalósítási hiba volt a legkülönbözőbb operációs rendszerekben. Ki ne emlékezne a *Ping of death* hibára, mikor gépeink könnyen válhattak egy igen hatásos szolgáltatásmegtagadás (DoS, Denial of Service) típusú támadás áldozatává. Az ilyen támadások általában az IP darabolási képességét, valamint a darabok helytelen kezelését használták ki. Jelenleg ezen támadások mérséklődtek, hiszen a mai IP-alrendszerek már védettek ellene. A fent említett *Ping of death* lényege az volt, hogy túlméretes ICMP ECHO_REQUEST csomagra a válasz hosszabb lett, mint 64 k. Mivel a rendszermag feltételezte, hogy egy csomag nem lehet hosszabb, mint 64 k, az eredmény végzetes volt: a rendszermag belső adatszerkezetei károsodtak, az eredménye magpánik lett). A csomagszűrő tűzfalak általában feltételezik, hogy az UDP- vagy TCP-fejléc az első darabban megérkezik. A forrás- és a célkapunak mindkét esetben meg kell érkeznie, hiszen ezek az IP szempontjából az első 8 bájton belül helyezkednek el (darabolás csak 8 bájtos határon lehetséges). A csomagszűrő kiveszi az első darabból a kapuadatokat, és meghozza döntését. Az utána következő darabokat továbbítja. Ha azonban a megcímzett gép helytelenül szereli össze a darabokat, érdekes helyzet állhat elő. Képzeljük el, hogy a támadó küld egy csomagot, amelyben az MF (More Fragment) bit igaz, így a címzett újabb darabra fog várni. A második és egyben utolsó darab viszont ismét eltolási értékkel érkezik, így összeszerelés közben a kapu címe felülíródik. Ezt szemlélteti a 2. ábra. Ebben az esetben a vett csomag nem annak a kapunak adódik át, amelyikre a döntést a csomagszűrő hozta. Ez a hiba már szintén nem létezik a korszerű IP-alrendszerrel rendelkező gépeken, és a csomagszűrők is tartalmaznak alapszintű védelmet ezzel a támadással szemben. Ugyancsak nehézségeket okoz az úgynevezett Source Routing használata. Ez két IP-lehetőség (IP option): (Strict source routing, illetve Loose source routing), amelyek segítségével a feladó a csomag haladási irányát tudja befolyásolni. Segítségével



3. ábra A Source Routing veszélyei

egy támadó elérheti, hogy csomagokat küldjön olyan gépeknek, amelyekkel nem tarthatna kapcsolatot. Célszerű ezt a lehetőséget minden útválasztón és számítógépen letiltani. A Source Routingra mutat példát a *3. ábra*.

Általános védekezés IP-szintű támadásokkal szemben

Most nézzük meg, hogy mit is tehetünk e támadások ellen. Az egyszerű IP-hamisítás és eltérítés ellen azonos módon kell védekeznünk, mint a lehallgatás ellen. Az egyetlen különbség az, hogy pusztán a rejtjelezés nem oldja meg a gondot, hiszen a rejtjelezés általában csak az adatokat érinti. Természetesen megoldás lehet valamiféle VPN használata, de ez intraneten belül ritkán használt. Ökölszabályként érdemes használni, hogy az eltérő előjogokkal bíró felhasználókat „fizikailag” is válasszuk el, azaz gépeik különválasztott hálózaton legyenek, amelyeket legalább csomagszűrővel ellátott útválasztók válasszanak el. A kiszolgálókkal egy hálózatra ne helyezzünk ügyfélgépeket. Kisebb cégeknél, ahol csak egy kiszolgáló van néhány munkaállomással, de a munkaállomásokban nem bízunk meg (például az Internetről különböző trójai falovakat hozhatnak be), egyszerű megoldás lehet például több hálózati kártyát helyezni a kiszolgálóba és így elkülöníteni a hálózatot (ez teljesítménynövekedéssel is jár). Amennyiben a hálózatunk felépítése megfelelő, a vak támadások ellen könnyű védekezni. Egyszerűen az útválasztóinkon valósítsuk meg az INGRESS/EGRESS szűréseket [5.]. A megoldás: alhálózatainkba ne engedjünk be olyan csomagokat, amelyek forráscíme belső gépre utal, és ne engedjünk ki olyan csomagokat, amelyek forráscíme nem benti gépé. Az Internettel összekötő tűzfalon ne felejtsük el eldobálni a csak belső hálózati forrás vagy célcímekkel bíró csomagokat (például LINKLOCAL tartomány: 169.254.0.0/16). Ezek után elérhetjük, hogy az IP-címek alapján a kiszolgálókon behatárolható lesz a csomag feladási tartománya. Ennek jelentőségét ne becsüljük le, hiszen a kiszolgálók és a tűzfalak gyakran hoznak cím alapján döntéseket. A címhamisítás elleni védelem kialakítására mutat példát a (*4. ábra*). Hálózatunkat az Internettől célszerű alkalmazásszintű tűzfallal elválasztani. Ezzel azonnal védettséget kapunk a különböző IP-szintű támadások ellen. Az átmenő adatokat a tűzfal saját protokoll alrendszere összemajd újra szétszereli. Ennek köszönhetően – mivel nincs csomagszintű kapcsolat – a csomagszintű hibák, – például gyenge IP-azonosító vagy TCP-sorszám előállító –, által okozottak, vagy például a darabösszeszerelési hibák nem juthatnak át. E lépések megtétele után a sikeres belső támadás esélye jóval kisebb, és többé-kevésbé hitelessé tudtuk tenni a belső hálózaton közlekedő csomagok IP-címeit. A jó védelemhez elengedhetetlen, hogy erős titkosítást alkalmazó protokollokat használjunk. Mint az előbbiekből látható, a rejtjelezés önmagában nem elegendő. Emellett a JRJ (a Jó, a Rossz és a Jó) támadások (Man In The Middle – MITM) kizárásához a kapcsolattartásban részt vevő feleknek tudniuk kell kölcsönösen azonosítani egymást, de legalább az ügyfélnek tudnia kell azonosítani a kiszolgálót.

Linux rendszerek beállításai

Ebben a szakaszban részletesen kitérünk arra, hogy az előző pontban említett védelmeket Linux-rendszereken miként lehet üzembe helyezni. Amennyiben a gépünk útválasztóként vagy tűzfalként működik, mindenképpen célszerű bekapcsolni a csomagok kötelező összeszerelését. Ha csak egy hagyományos géppel vagy kiszolgálóval állunk szemben, akkor sem ártunk vele:

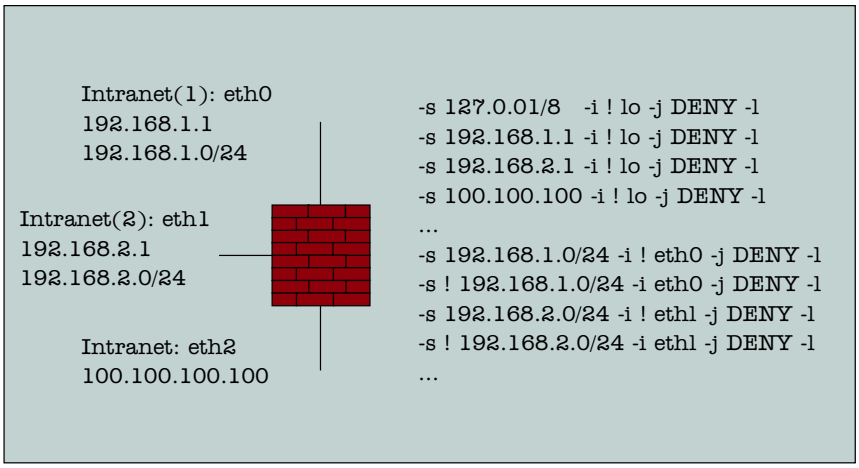
```
(bozo@dragon) ~ # echo 1 >
       /proc/sys/net/ipv4/ip_always_defrag
```

A *source routed* csomagokat ne fogadjuk el. Ugyanígy dobáljuk el a REDIRECT kategóriájú ICMP csomagokat, hiszen egy jól beállított gépen nem kaphatunk ilyeneket (természetesen nagyobb és bonyolultabb hálózaton előfordulhatnak):

```
(bozo@dragon) ~ # echo 0 >
/proc/sys/net/ipv4/conf/all/accept_source_route
(bozo@dragon) ~ # echo 0 >
/proc/sys/net/ipv4/conf/all/accept_redirects
(bozo@dragon) ~ # echo 0 >
/proc/sys/net/ipv4/conf/all/secure_redirects
```

Kapcsoljuk be a gyanús csomagok naplózását. Ehhez a rendszermagot a CONFIG_IP_ROUTE_VERBOSE szolgáltatással kell fordítani (ehhez be kell kapcsolni a CONFIG_IP_ADVANCED_ROUTER szolgáltatást):

```
(bozo@dragon) ~ # echo 1 >
       /proc/sys/net/ipv4/conf/all/log_martians
```



4. ábra A hallgatózás elhárítása

Kapcsoljuk be a rendszermagba épített IP-címhamisítás-érzékelőt. Természetesen a második sort ismételjük az összes hálózati csatolóra.

```
(bozo@dragon) ~ # echo 1 >
       /proc/sys/net/ipv4/conf/all/rp_filter
(bozo@dragon) ~ # echo 1 >
       /proc/sys/net/ipv4/conf/eth0/rp_filter
```

Megjegyezzük, hogy ez a megoldás bizonyos más, magasabb szintű biztonsági eszközök működését (például IPSEC), illetve a számozatlan IP-csatoló (unnumbered IP interface) használatát kizárja. Gondot okozhat még egyes csatornaegyesítési (például PPP multilink) eljárások használatában – pedig ez utóbbi megoldással tudunk 2x64kbit/sec ISDN elérést használni. Amennyiben az említett feltételek valamelyike fennáll, a rendszermag csomagszűrő szolgáltatásával tudjuk az *rp_filter* szolgáltatást kiváltani, azokon a csatolókon, ahol alkalmazása szükséges. Ez egyébként mindig hasznos lehet, a többszintű védekezés sosem árt, az egyetlen hátránya a fokozottabb felügyelet.
Két hasznos tanács a fentiekhez: a legtöbb jelenlegi Linuxban nem szükséges „kézzel” kiadni ezeket a parancsokat, induláskor a rendszer a `sysctl(8)` vagy `systune(8)` parancsok valamelyike egy állományból is képes felolvasni és beállítani. A másik hasznos tanács, hogy ne egyesével állítsuk hálózati csatolóinkra, hanem módosítsuk a /proc/sys/net/ipv4/conf/default/ könyvtár alatt levő bejegyzéseket. Amennyiben ez a hálózati csatolók elindítása előtt történik meg, a csatolóra vonatkozó értékeket innen fogja venni.
Nézzük meg, hogy miként állíthatunk be állandó ARP-bejegyzéseket. Erre az `arp(8)` parancs szolgál. Használata az alábbi:

```
(bozo@dragon) ~ # arp -s 192.168.1.1
       11:22:33:44:55:66
```

ahol 192.168.1.1 a gép IP-címe, a második érték pedig az ethernet-címe. Az ARP-gyorstár tartalmát megnézhetjük az alábbi paranccsal:

```
(bozo@dragon) ~ # arp -a
```

Amennyiben szeretnénk kikapcsolni valamely csatolónkon az ARP lehetőséget, adjuk ki az `ifconfig(8)` parancsot a `-arp` kapcsolóval, például:

```
(bozo@dragon) ~ # ifconfig eth0 -arp
```

Az ARP-táblák változásait célszerű figyelni, hiszen így észlelhetjük az újonnan megjelenő gépeket, valamint az IP-címüket megváltoztató gépeinket. Erre hasznos segédeszköz lehet például az `arpwatch(8)` segédprogram. Amennyiben egy megadott gép TCP sorszám-előállítója érdekel minket, adjuk ki az alábbi parancsot:

```
(bozo@dragon) ~ # nmap -O linux
Starting nmap V. 2.54BETA7
(www.insecure.org/nmap/)
Interesting ports on linux.nowhere (192.168.1.1):
(The 1531 ports scanned but not shown below are
in state: closed) Port State Service 22/tcp open
sshTCP Sequence Prediction: Class=random positive
increments Difficulty=1693649 (Good luck!)
Remote operating system guess: Linux 2.1.122 -
2.2.16
Nmap run completed
-- 1 IP address (1 host up) scanned in 2 seconds
```

Szóval ez a jó. Ha az alábbihoz hasonlót látunk, kezdhetünk aggódni:

```
(bozo@dragon) ~ # nmap -O loose95
Starting nmap V. 2.54BETA7
(www.insecure.org/nmap/)
Interesting ports on loose95.nowhere
(192.168.1.2): (The 1522 ports scanned but not
shown below are in state: closed) Port State
Service 139/tcp open netbios-ssn
TCP Sequence Prediction: Class=trivial time
dependencyDifficulty=2 (Trivial joke)
Remote operating system guess: Windows
NT4/Win95/Win98
Nmap run completed
-- 1 IP address (1 host up) scanned in 1 seconds
```

Irodalomjegyzék

[1.] Andrews S. Tannenbaum: Számítógép-hálózatok
[2.] W. Richard Stevens: TCP/IP Illustrated, Volume 1
[3.] Linuxvilág magazin 2001. február–márciusi száma
[4.] RFC826: An Ethernet Address Resolution Protocol
[5.] RFC2827: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing.
[6.] CERT Advisory CA-1996-21 TCP SYN Flooding and IP Spoofing Attacks
➲ http://www.cert.org/advisories/CA-1996-21.html
[7.] CERT Advisory CA-1995-01 IP Spoofing Attacks and Hijacked Terminal Connections
➲ http://www.cert.org/advisories/CA-1995-01.html
[8.] Tsutomu Shimomura: Technical Details of the Attack Described by Markoff in NYT
➲ http://www.netsys.com/firewalls-9501/0900.html
[9.] Linux Blind TCP Spoofing
➲ http://www.securityfocus.com/templates/archive.pike?list=1&mid=12805
[10.] Linux blind TCP Spoofing, act II + others
➲ http://www.securityfocus.com/templates/archive.pike?list=1&mid=20979
[11.] Fyodor: Remote OS detection via TCP/IP Stack FingerPrinting
➲ http://www.insecure.org/nmap/nmap-fingerprinting-article.html
[12.] Steven M. Bellovin: Security problems in the TCP/IP protocol suite, Computer Communications Review 2:19, pp. 32-48, April 1989 ➲ http://www.research.att.com/~smb/papers/ipext.ps
[13.] IP-spoofing Demystified (Trust-Relationship Exploitation)
➲ http://www.phrack.com/search.phtml

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.

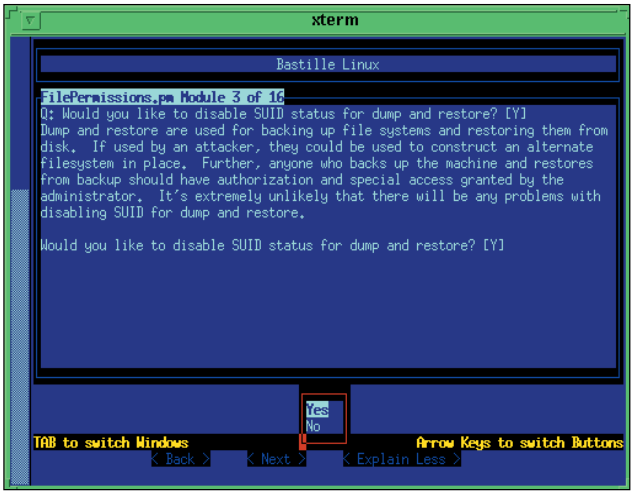
Borbély Zoltán (bozo@andrews.hu), okleveles mérnök-informatikus. Főként Linuxon futó számítógépes biztonsági rendszerek tervezésével és fejlesztésével foglalkozik.
A 1.0.9-es rendszermag ideje óta linuxozik. Szabadidejét barátaival tölti.

Hogyan erősítsük rendszerünk biztonságát?

A biztonsági rések betömése a Bastille segítségével.

Vajon Ön is azok közé közé tartozik, akiknek, amikor a Linux telepítése után először kiadják a `ps -ef` parancsot, fogalmuk sincs, hogy mit csinál a megjelenő listában felsorolt folyamatok fele? Ne szégyellje mindenkinek el kell kezdenie valahol, rengeteg súgóoldalt át kell olvasni, mire megértjük a Unix-rendszert felépítő rengeteg alkalmazás és démon működését. A súgóoldalak olvasgatását semmivel sem lehet kiváltani, de az is túlságosan nagy könnyelműség lenne, ha biztonsági hézagokban bővelkedő rendszereket üzemeltetnénk mindaddig, amíg el nem érjük a guru állapotot.

A Bastille Linux a rendszer megerősítését szolgáló Perl parancsfájlgyűjtemény, mely biztonságosabbá teszi az operációs rendszert, és ha interaktív üzemmódban használjuk, oktat is. Tiszta és célratörő kérdéseket tesz fel, melyek megválaszolásával egyedi biztonsági szabályozást alakíthatunk ki rendszerünkben. Minden kérdésre részletes választ kapunk, így mire megismerkedünk a Bastille-jal, jó néhány dolgot megtudunk a Linux/Unix biztonságáról is.



InterActiveBastille.pl

Aki pedig már tisztában van a rendszer biztonsági kérdéseivel, és a Bastille használata közben szeretne megtakarítani egy kevés időt, kérheti a Bastille-t, hogy kevesebb magyarázatot jelenítsen meg. Ebben az esetben is ugyanazokat a kérdéseket teszi fel nekünk, viszont nem látjuk a kérdésekhez tartozó hosszadalmas magyarázatokat. Ha „tiszta” Linux telepítésen futtatjuk a Bastille-t, akkor akár ki is hagyhatjuk a kérdéseket és választhatunk egyet az előre összeállított biztonsági sablonok közül. Természetesen, ha igazi nagymenők vagyunk, akkor saját beállításablont írunk majd a Bastille-hoz (vagy ami ennél valószínűbb, a programhoz mellékelt sablonokat fogjuk saját igényeinkhez igazítani).

Miért hasznos a Bastille?

Talán megfordult már a fejünkben az, hogy vajon miért kell olyan sok szolgáltatást engedélyezni a rendszerben alapértelmezés szerint? Nem butaság az, hogy külön parancsfájlokra van szükségünk a felesleges dolgok lefaragásához? Nem lenne-e egyszerűbb, ha ezek már eleve nem lennének ott?

Véleményem szerint a legtöbb Linux-válozat alapértelmezés szerint túlságosan sok dolgot engedélyez. Az azonban tény, hogy a Linux-felhasználók között egyre több a kezdő, és ha a rendszertelepítés után azt tapasztalják, hogy a Linux túlságosan kevés dologra képes (vagy ami még rosszabb, egyáltalán nem is működik), akkor valószínűleg igen kevesen fognak megmaradni mellette, és még kevesebben fognak azzal foglalkozni, hogy rendszerüket megtanulják biztonságosan üzemeltetni.

Más szóval a Linux-változatokat (például a RedHat, a Caldera stb.) összeállítók általában a használhatóságra fektetik a hangsúlyt, nem pedig a biztonságra. Én jobban szeretném, ha a jelenleginél több változat telepítője kínálna a lehető legtöbb szolgáltatást biztosító és „biztonságos” beállítási lehetőségeket is. (A RedHat 7.0 telepítője tartalmaz ugyan ilyen lehetőségeket, de a „biztonságos” lehetőség választása esetén sincs olyan biztonságban a rendszer, mint a Bastille üzembe helyezése után.)

Hogyan született a Bastille?

A Bastille fejlesztőcsapatát *Jon Lasser* és *Jay Beale* vezette. Eredeti céljuk az volt, hogy létrehozzanak egy olyan Linux-változatot, amely a RedHatre épül, de sokkal biztonságosabb annál. Úgy tűnt, hogy céljukat a legkönnyebben úgy érhetik el, ha egy átlagos RedHat rendszerből indulnak ki, és különböző Perl parancsfájlok segítségével fokozatosan biztonságosabbá teszik azt. Ez sokkal rugalmasabb megoldásnak bizonyult, ugyanis olyan parancsfájlokat tudtak létrehozni, amelyek több különböző változat megerősítésére is felhasználhatók.

A Bastille-t felépítő parancsfájlok meglehetősen értelmesek, és sokkal kevesebbet feltételeznek a rendszerünkről, mint korábbi testvéreik (kezdetben kizárólag újonnan telepített RedHat rendszerre tervezték e parancsfájlokat). Egyáltalán nem fontos, hogy a rendszer frissen legyen telepítve, sőt még az sem, hogy RedHat-változat legyen. A Bastille az 1.1.x változat óta számos adatot begyűjt rendszerünkről, mielőtt módosításokat végezne rajta.

A Bastille beszerzése és telepítése

Remélem, mindenki felkészült lelkileg arra, hogy némi oktatás után megerősítse rendszerét. Fontos megjegyezni, hogy az eredményesség változó lehet. Habár a Bastille szinte minden Linux-változattal használható, eredetileg a RedHat segédprogramjának készült, és továbbra is ezeken a rendszereken a leghatékonyabb. A cikk során kitérek majd a nem RedHat-változatokkal kapcsolatos kérdésekre is, az eredményért azonban nem állok jót. Ha bizonytalanok valamiben, akkor látogassanak el a Bastille honlapjára. Ha már itt járunk, meg kell említenünk, hogy ez a hely a Bastille Linux, illetve a hozzá kapcsolódó leírások legbiztosabb forrása. A ➲ http://www.bastille-linux.org/ lap tetején mindig megtaláljuk azt a nagy kövér betűkkel szedett hivatkozást, amely a Bastille Linux legfrissebb változatára mutat. Miután letöltöttük a tar fájlt, helyezzük a /root könyvtárba, és csomagoljuk ki:

```
tar -xzvf ./Bastille-X.Y.Z.tgz
```

Ez az, sikerült a telepítés!

Ne felejtsük el, a Bastille azt feltételezi, hogy a /root könyvtárba telepítettük. Valószínűleg át tudjuk alakítani a Bastille parancsfájlokat úgy, hogy másik könyvtárban is jól érezzék magukat, ezt azonban senkinek sem ajánljuk, mivel a hatás kiszámíthatatlan (ráadásul igen sok parancsfájlról van szó).

A Perl 5 parancsnyelvre is szükségünk lesz ahhoz, hogy a Bastille-t futtatni tudjuk. Ha szeretnénk meggyőződni arról, hogy tartalmazza-e a megfelelő változatot a rendszerünk, akkor egyszerűen adjuk ki a `perl -v` parancsot. Ha a Perl 5.0-nál alacsonyabb változatszámmal válaszol, vagy pedig a *perl: command not found* üzenet jelenik meg a képernyőn, akkor telepítenünk kell rendszerünkre vagy frissíteni kell Perl változatunkat. Egyetlen újabb Linux-változatból sem hiányzik a Perl 5. Nézzünk szét a CD-ROM-on vagy a változathoz tartozó webhelyen.

Az első lépések

A Bastille használata egyszerű. Futtassuk le a /root/Bastille könyvtárban található InteractiveBastille.pl parancsfájlt (lásd *képünkön*). Hosszú kérdéssorozatot kell megválaszolnunk, hogy milyen szolgáltatásokat engedélyezünk. Hogyan kell beállítanunk azokat, hogy megtalálhassuk a szolgáltatások és a biztonság közötti egyensúlyt. Ezek a kérdések különböző témák szerint vannak csoportosítva. A kérdésekre adott feleleteket a *config* nevű fájlban tárolja a rendszer.

Ezután futtassuk a BackEnd.pl parancsfájlt, amely sorra meghívja az InteractiveBastill.pl egyes részeihez kapcsolódó megfelelő parancsfájlokat. A parancsfájlok működését meghatározó változókat a *config* fájl alapján állítja be a rendszer. A válaszoktól függően előfordulhat, hogy egyes parancsfájlokat egyáltalán nem futtat le.

Ha nem szeretnénk ezekkel a kérdésekkel bajlódni, akkor futtassuk az AutomatedBastill.pl parancsfájlt, amely lehetőséget ad arra, hogy különböző alapértelmezett beállítások szerint erősítsük rendszerünk biztonságát. Az AutomatedBastille.pl nagyon egyszerű parancsfájl. Mindössze annyit tesz, hogy meghívja a BackEnd.pl parancsfájlt az előre legyártott beállítási fájllal.

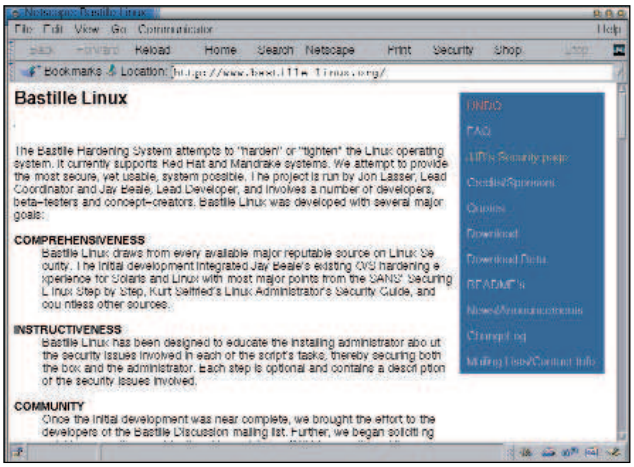
Ezeket az előkészített fájlokat (a Bastille v.1.1.0 változatában Default_Workstation és Default_Workstation_plus_Firewall) könnyedén a saját igényeinkhez igazíthatjuk. Természetesen mi magunk is létrehozhatunk ilyen fájlokat, és az AutomatedBastille.pl megkerülésével közvetlenül a BackEnd.pl parancsfájlt is használhatjuk:

```
./Backend.pl ./beállítási_fájl /root/naplófájl
```

Megjegyzések az InteractiveBastille.pl használatához

Az InteractiveBastille.pl bőséges magyarázatot tartalmaz. Ha elegendő időt szánunk az egyes kérdésekhez tartozó válaszok tanulmányozására, akkor közben igen sokat megtudhatunk a rendszer megerősítésének módszereiről. Ha úgy gondoljuk, hogy már kellőképpen járatosak vagyunk a témában, akkor bármikor átállhatunk a magyarázatokban szegényebb üzemmódra (ez természetesen fordítva is igaz). Az alábbi általános tanácsok hasznosak lehetnek a kezdők számára:

- *Első modul: IPChains.pm* – Az IPChains linuxos tűzfal. Ha gépünkhöz hozzáférhetnek majd az Interneten keresztül más számítógépek, mindenképpen célszerű használnunk. Néhány egyszerű csomagszűrő szabállyal is lényegesen növelhetjük rendszerünk biztonságát.
- *Második modul: PatchDownload.pm* – Ha RedHat rendszert használunk, akkor a Bastille képes arra, hogy RPM-eket töltsön le és telepítsen azokhoz a programokhoz, amelyek telepítésük óta megváltoztak.
- *Harmadik modul: FilePermissions.pm* – Ez a modul korlátozza a hozzáférést bizonyos segédprogramokhoz és fájlokhoz. Ezt elsősorban a SUID jelző kikapcsolásával éri el. A SUID képes arra, hogy úgy indítson el különböző folyamatokat, mintha a rendszergazda hívta volna meg azokat. Így az egyszerű felhasználók is lehetőséget kapnak arra, hogy olyan segédprogramokat futtassanak, amilyen például a mount, a ping vagy a traceroute. Azoknak a felhasználóknak, akik nem rendelkeznek a megfelelő jogosultságokkal, általában nincs is szükségük ezekre a segédprogramokra. A SUID jelző kikapcsolásával ezeket a visszaélésre is alkalmat adó programokat kizárólag a rendszergazda futtathassa.
- *Negyedik modul: AccountSecurity.pm* – Ez a modul lehetőséget ad arra, hogy új rendszerfelügyelői jogosultságokkal felruházott azonosítót hozzunk létre, és általában véve biztonságosabbá tehessük a felhasználói azonosítók kezelését. Ezeket a lépéseket mindenképpen célszerű megtennünk.
- *Ötödik modul: BootSecurity.pm* – Amennyiben rendszerünk elé ismeretlen személyek is leülhetnek, újraindíthatják azt, és az indítási folyamat megváltoztatásával illetéktelen jogokat gyakorolhatnak. E modul segítségével megnehezíthetjük a rendszer ilyen típusú veszélyeztetését.
- *Hatodik modul: SecureInetd.pm* – Ebben a részben internetes szolláltatásainkat tudjuk megerősíteni.
- *Hetedik modul: DisableUserTools.pm* – Tanácsos letiltani a fordító használatát abban az esetben, ha a rendszergazdai jogosultságokkal nem rendelkező felhasználóknak nincs kifejezetten szükségük rá. Amint a legtöbb hasonló esetben, a „letiltás” itt is azt jelenti, hogy az adott szolgáltatást kizárólag a rendszergazda veheti igénybe.
- *Nyolcadik modul: ConfigureMiscPAM.pm* – Itt különféle, a felhasználói azonosítókra vonatkozó korlátozásokat állíthatunk be.
- *Kilencedik modul: Logging.pm* – A legtöbb rendszer alapértelmezés szerint csak igen szűkszavú naplózást engedélyez. Ezzel a modullal kiterjeszthetjük a naplózást és lehetőséget kapunk arra, hogy a naplóadatokat távoli gépekhez továbbítsuk. A folyamatok nyomon követését is bekapcsolhatjuk itt, a legtöbb rendszer esetében azonban erre nincs szükség.
- *Tizedik modul: MiscellaneousDaemons.pm* – Ebben a részben olyan szolgáltatásokat tudunk kikapcsolni, amelyek alapértelmezés szerint bárki számára hozzáférhetők, holott a legtöbb felhasználónak valójában nincs is szüksége rájuk.
- *Tizenegyedik modul: Sendmail.pm* – Magáért beszél.
- *Tizenkettedik modul: RemoteAccess.pm* – Ha az SSH (Secure Shell) még nincs jelen a rendszerünkben, akkor a Bastille képes letölteni és telepíteni azt. Az SSH biztonságos szolgáltatásokat tartalmaz, amelyek képesek helyettesíteni a telnet, az rsh és az



➲ www.bastille-linux.org/

➲ www.enteract.com/~lspitz/

rlogin szolgáltatásokat. Fontos megjegyezni, hogy a Bastille az i386-os gépeken futó RedHat rendszerekhez fordított RPM-eket próbálja meg telepíteni. Ha Linuxunk nem PC-megfelelő környezetben fut, vagy pedig olyan változatot használunk, amely nem képes kezelni a RedHat RMP-eket, akkor ezt a modult nem tudjuk majd használni.

Naplózás és a Bastille futtatása nem RedHat-változatokon

Mi történik azután, hogy az InteractiveBastill.pl létrehozta a config fájlt, a BackEnd.pl pedig megtette a megfelelő lépéseket? Honnan tudhatjuk meg, hogy mi zajlik a rendszerünkben? A Bastille kiváló naplózási lehetőségeinek köszönhetően igen egyszerűen kideríthetjük azt, hogy mely változtatásokat sikerült végrehajtani, és melyeket nem. Ha már itt járunk, célszerű néhány szót ejteni arról is, hogyan használhatjuk a Bastille-t nem RedHat-változatok esetén.
Amint azt korábban már említettük, a Bastille a RedHat-változatra, illetve ennek leszármazottaira (Mandrake, Yellow Dog stb.) lett kialakítva (optimalizálva), mégis elég értelmes ahhoz, hogy más rendszerekkel is elboldoguljon. Tulajdonképpen saját tapasztalataim is ezt az állítást igazolják: lefuttattam a Bastille 1.1.0-t SuSE Linux 6.3 alatt, és örömmel állapítottam meg, hogy a programnak több olyan része volt, amely megfelelően működött, mint ami nem.
A SuSE 6.3 használata során elsőként azt vettem észre, hogy a Bastille helyenként meglehetősen RedHat-központú. A PatchDownload.pm kizárólag RedHat alatt használható, és amint azt korábban már megjegyeztem, a RemoteAccess.pm az SSH 1.2.27 Red Hat i386-os változatát telepíti.
Figyelmeztetéseket kaptam akkor is, amikor a namedet chroot szolgáltatással próbáltam futtatni. Szerencsére a csomagok letöltése és a named futtatása olyan feladatok, amelyek kézzel is könnyen elvégezhetők.
Habár az InteractiveBastill.pl futtatása során kizárólag ez a két nehézség jelentkezett, voltak más olyan részei is, amelyek nem voltak képesek együttműködni a SuSE-változattal. A BackEnd.pl például hiba nélkül lefutott. A futás során hibákat csak akkor vettem észre, amikor átolvastam a Bastille naplóit.
Ezeket a naplókat mindenképpen érdemes átnézni, még akkor is, ha úgy gondoljuk, hogy a Bastille mindent megfelelően végrehajtott. Az értelmes naplózás egyike a Bastille leghasznosabb szolgáltatásainak. Függetlenül attól, hogy kezdők vagyunk vagy Linux-szakértők, nemcsak arról kell tudnunk, hogy mit csinál a Bastille, hanem utána kell néznünk annak is, hogy hogyan csinálja azt.
Ésszerű módon a Bastille a /root/Bastille/log mappába írja a naplóit. A BackEnd.pl két naplót hoz létre: action-log és error-log néven. Az action-log átfogó és részletes leírást tartalmaz a Bastille tevékenységéről, a hibák és a többi váratlan esemény pedig az error-log naplóba kerül.
A SuSE Linux használata során a hibát az okozta, hogy bizonyos fájlok nem voltak ott, ahol a Bastille kereste őket, ugyanis a SuSE egyes fájlokat máshol tárol, mint a RedHat. Igen egyszerűen rá lehet jönni, hogyan lehet kézzel kijavítani a hibát. A Bastille kérdéseinek listája (/root/Bastille/Questions.txt) számtalan ötletet tartalmaz, és ha ismerjük a Perl nyelvet, akkor a parancsfájlokat magunk is módosíthatjuk.
A legegyszerűbb megoldás az, ha módosítjuk a Bastille parancsfájlokban az elérési útvonalakat, majd újra lefuttatjuk a BackEnd.pl-t. A legtöbb elérési út, amelyet a SuSE esetében módosítanom kellett, a /root/Bastille/Bastille/FilePermissions.pl parancsfájlban volt. Minden olyan fájlt, ami a Bastille hibanaplója szerint hiányzott, megkerestem a which segítségével. Ha a keresett fájl létezett, épp csak egyszerűen rossz helyen volt, akkor módosítottam a /root/Bastille/Bastille/FilePermissions.pl megfelelő részét.
Három fájlt gyakran nem talál SuSE alatt a Bastille: a setserial, a chkconfig és az ifdown fájlokat. A chkconfig és az ifdown nem is léteznek a SuSE rendszerekben; ezekkel kizárólag a RedHat illetve az abból származó rendszerekben találkozhatunk. Ezekkel a hibákkal tehát nem kell foglalkoznunk. De mi a helyzet a setseriallal?
A setserial pontos helye a `/sbin/setserial`. Ha kiadjuk a

```
grep setserial /root/Bastille/Bastille/
```

parancsot, akkor a következőt láthatjuk:

```
/root/Bastille/Bastille/FilePermissions.pm:
&B_chmod(0750,"/bin/setserial");
```

Viszonylag kevés nem létező fájl esetén a dolog tehát egyszerű. Sorra vettem az egyes fájlokat, és kiderítettem, hogy léteznek-e, és ha igen, akkor hol vannak. Ezután már csak a megfelelő módosításokat kellett elvégezni a FilePermissions.pm modulban. A hibanaplóban felsorolt fájlok felderítése és a BackEnd.pl újrafuttatása nem tartott tovább húsz percnél.
Volt egy másik gond is, amit meg kellett oldanom. Ezt mutatja be a következő példa:

```
#open /etc/httpd/conf/httpd.conf.bastille
#failed...
#open /etc/httpd/conf/httpd.conf failed.
Couldn´t replace line(s) in
/etc/httpd/conf/httpd.conf because open failed.
#open /etc/httpd/conf/httpd.conf.bastille
#failed...
#open /etc/httpd/conf/httpd.conf failed.
Couldn´t replace line(s) in
/etc/httpd/conf/httpd.conf because open failed.
```

Ezt a hibát a RedHat és a SuSE Apache környezetének az eltérése okozza, amelyről korábban már beszéltünk. Ebben az esetben is a `grep` parancsot használtam:

```
grep httpd.conf /root/Bastille/Bastille/*.pm
```

A parancs kimenete a következő volt:

```
API.pm:
$GLOBAL_FILE{"httpd.conf"}="/etc/httpd/conf/httpd
.conf";
API.pm:
$GLOBAL_FILE{"httpd_access.conf"}="/etc/httpd/con
```

```
f/httpd.conf";
API.orig.pm:
$GLOBAL_FILE{"httpd_access.conf"}="/etc/httpd/con
f/access.conf";
Apache.pm:  my
$httpd_file=$GLOBAL_FILE{"httpd.conf"};
```

Ezekből a sorokból kiderül, hogy az Apache modul egy GLOBAL_FILE nevű változóban tárolja a httpd.conf elérési útját, és ez a változó az API.pm modulban kerül beállításra. Nem kellett mást tennem, mint megváltoztatni ezt az elérési útvonalat az API.pm modulban, majd újra lefuttatni a BackEnd.pl-t (amelyben az Apache.pm hívását tartalmazó soron kívül minden mást megjegyzésbe helyeztem).
Azok a segédprogramok, amelyek biztonságát nem erősítettük meg a Bastille segítségével, könnyen módot adhatnak az egyszerű felhasználóknak arra, hogy rendszergazdai jogokat szerezzenek maguknak a rendszerben. Mindenképpen érdemes tehát rászánnunk az időt ezeknek a hibáknak a felderítésére és kijavítására, főleg abban az esetben, ha egynél több rendszer biztonságról van szó. Nyilvánvaló, hogy minél több azonos típusú rendszer biztonságáról kell gondoskodunk, annál inkább megtérül a munkával töltött idő.

Most már teljes biztonságban vagyunk. Vagy mégsem?

Gondosan átolvastuk és megválaszoltuk az InteractiveBastill.pl által feltett kérdéseket, lefuttattuk a BackEnd.pl-t, átnéztük a Bastille által létrehozott naplófájlokat, és miután kijavítottuk az esetleges hibákat, újra lefuttattuk a BackEnd.pl parancsfájlt. Készen vagyunk?
Nem, és soha nem is leszünk! A biztonsági rendszer karbantartása folyamat, nem pedig egyszeri feladat. A legbiztosabb módja annak, hogy sebezhető rendszert hozzunk létre az, hogy az üzembe helyezés után egyszerűen magára hagyjuk. Ez még akkor is igaz, ha mielőtt valóban magára hagynánk a rendszert, először megerősítjük azt.
Természetesen még a Bastille sem tud előre felkészülni minden olyan programcsomagra, amelyet az adott rendszer alá telepíthetünk. Egyes változatok, például a Debian és a SuSE rengeteg csomagot tartalmaznak, olyan sokat, hogy egy nemrégiben megjelent beszélgetésben Jon Lasser, a Bastille egyik atyja, külön kihangsúlyozta, hogy mekkora gondot okoznak ezek biztonsági szempontból nézve. Hozzátette, hogy ez nem azt jelenti, hogy azokat a rendszereket, amelyek sok különböző programcsomagot tartalmaznak, ne lehetne biztonságossá tenni; mindössze annyit jelent, hogy több munkát igényel.
Van még néhány feladat, amelyeket feltétlenül el kell végeznünk a Bastille futtatása után:

- *A megmaradt szolgáltatások közül tiltsuk le azokat, amelyekre nincs szükségünk.* Nézzünk utána a /etc/rc.d/rc{{n}}.d fájlban (az {{n}} az alapértelmezés szerinti futtatási szintet jelöli, ennek az értékét a `grep initdefault /etc/inittab` parancs kiadásával tudhatjuk meg), hogy melyek azok a szolgáltatások, amelyek továbbra is önműködően elindulnak a rendszer indítása során. A rendszer indításával együtt induló parancsfájlokat a nevük előtt álló nagy S betű jelzi. Ha nem tudjuk, hogy mire jó egy adott démon, akkor adjuk ki a `man` parancsot a démon nevével. Ha nincs szükségünk az adott szolgáltatásra, akkor nevezzük át az `mv` parancs segítségével (ha a név nem S-sel kezdődik, akkor a szolgáltatás nem indul el önműködően).
- *Az alkalmazásaink biztonságossá tételére is szánjunk elegendő időt.* A Bastille segítségével a rendszerünket anélkül is biztonságossá tehetjük, hogy mindent tudnánk a rendszerek biztonságáról. Ez az oka annak, hogy a készítők olyan sok időt szántak az oktatási anyagok beillesztésére. A programfájlgyűjtemény által biztosított alkalmazások azonban nem lehetnek igazán biztonságosak, ha nem értjük a működésüket.

A BIND például, amely a hálózati alkalmazások számára szükséges DNS névfeloldást biztosítja, számos biztonsági szolgáltatást tartalmaz. A Bastille sokat beállít ezek közül, de számos olyan marad, amit nem. A megfelelő man oldalak átolvasása tehát kötelező minden rendszergazda számára.

- *Tiltsuk le a nem használt felhasználói azonosítókat.* A SuSE egyik legbosszantóbb sajátossága az, hogy a /etc/passwd fájl rengeteg, különböző alkalmazásokhoz tartozó jelszóbejegyzést tartalmaz, függetlenül attól, hogy telepítettük-e az alkalmazásokat vagy sem. Ezek között az alkalmazások között sok olyan van, amelyek interaktív bejelentkezésre is lehetőséget adnak (nem a /bin/false héj van megadva mellettük).

Ezzel kizárólag a SuSE-változatban találkozhatunk. Éppen ezért győződjünk meg arról, hogy a /etc/passwd fájlban minden felesleges bejegyzést megjegyzésbe helyeztünk. Ha kétségeink merülnének fel, akkor figyeljünk oda arra, hogy az utolsó mezőben (default shell – alapértelmezés szerinti héj) ne egy valódi héj, hanem a /bin/false szerepeljen – csak a tényleges felhasználói azonosítók esetén van szükség héjra.

- *Folyamatosan frissítsük a rendszert.* Nem lehet eléggé kihangsúlyozni annak fontosságát, hogy mindig naprakészek legyünk. Kövessük nyomon a változatokhoz megjelenő biztonsági javításokat. Ahogy egyre jobban megismerjük a Linux biztonsági rendszerét, azonnal alkalmazzuk az új tudást saját rendszerünkre is. Semmiképpen se hagyjunk magára egy Internetre kapcsolt gépet.
- *Telepítsünk egy betörésészlelő programot.* Az operációs rendszer telepítése után minél előbb telepítsünk betörésészlelő programot is, ilyenek például a tripwire vagy a snort. Ezzel csökkenthetjük annak esélyét, hogy mástól (rendőrség, ideges rendszergazdák stb.) kelljen tudomást szereznünk arról, hogy betörtek a rendszerünkbe, ráadásul más rendszerek megtámadásához is felhasználták.
- *Figyeljük a naplókat!* Nem szabad elfeledkeznünk arról sem, hogy folyamatosan nyomon kövessük a Bastille által készített naplóbejegyzéseket.

A naplófájlok böngészése igen fárasztó is lehet, éppen ezért célszerű olyan parancsfájlokat készíteni, amelyek időnként önműködően átnézik a naplófájlokat és kigyűjtik a gyanús bejegyzéseket, vagy telepíteni olyan eszközt, mint például a swatch, amely képes elvégezni ezt a feladatot.

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.

Kapcsolódó címek

A Bastille hivatalos honlapja
➲ http://www.bastille-linux.org/

Interjú Jonnal és Jayjel, a Bastille készítőivel
➲ http://slashdot.org/interviews/00/11/08/1616204.shtml

Lance Spitzner Armoring Linux (A Linux felfegyverzése) című cikke ➲ http://www.enteract.com/~lspitz/linux.html

A Linux Documentation Project hivatalos Linux biztonsági lapja ➲ http://www.linuxdoc.org/HOWTO/Security-HOWTO.html

Levelezőrendszer kisebb irodáknak

Hogyan rejtőzködhet egy egész vállalat egyetlen levélcím mögött?

A vállalatot, ahol elhelyezkedtem, nyugodtan nevezhetjük fukarnak is, némely berendezésünk még az 1930-as évekből származik. Már három hónapja itt dolgoztam, amikor rájöttem, hogy jé, van internet-előfizetésünk (ez azért utólag nem túl kellemes felfedezés egy rendszergazda számára…). Ez volt az első ilyen jellegű munkám és azt gondoltam, hogy legalább kipróbálom magam, és tapasztalatokat szerzek. Később rájöttem, hogy az internet-hozzáférést titokban a tulajdonos fia és annak haverjai használják a tervezőirodában.

Ahogy telt-múlt az idő, a vásárlók egyre többször keresték a korszerű vásárlás lehetőségeit, honlapot akartak megnézni, leveleket küldeni stb. Sok helyen tapasztaltam már, hogy egy linuxos gép hálózatba helyezésével könnyen kisegíthető a vállalat belső rendszerének működése, s a befektetés hamar visszahozza az árát. Meg is kaptam az engedélyt egy linuxos gép beszerzésére, melyet fájl-, fax-, belső web- és levelező kiszolgálóként, valamint internetes átjáróként szándékoztam üzemeltetni. A tervezőirodában dolgozókon kívül a cégnél még e-mailt sem használt senki, s az ottaniak is csak elvétve. Azt gondolták, hogy az Internet egyenlő a Webbel (és most inkább hadd ne térjek ki az általuk gyakran látogatott honlapokra…).

Írásomban foglalkozom a sendmail, a fetchmail és a procmail használatával, ugyanis ezek segítségével egy egész vállalat „rejtőzködhet” egyetlen levélcím mögött. Mielőtt belevágnánk, azért nézzünk utána, hogy a szolgáltatónknak mi erről a véleménye. Esetünkben a napi levélmennyiség általában nem emelkedik húsz darab fölé. Saját postafiókom napi 100-400 levelet kitevő forgalmával összehasonlítva ez szinte semmi, s ezért a szolgáltatónak valószínűleg nincs oka panaszra. Azonban minden szolgáltató más és más elvek alapján dolgozik, ezért inkább legyünk körültekintőek.

Megfigyelték már, hogyan néznek ki a levelekben található válaszcímek? Ha az illető beállította teljes nevét a levelezőprogramban vagy a rendszergazda beleírta a /etc/passwd fájlba – a `vezetéknév keresztnév <valaki@valami.com>` alakot találjuk.

Meg voltam győződve arról, hogy ezt valahogy a javunkra lehetne fordítani. Ha a kimenő válaszcímet úgy írjuk át, hogy a válaszleveleket egy szűrő képes a megfelelő személyhez továbbítani, akkor a szolgáltatótól csupán egy címet kell igényelnünk. Átböngésztem az O'Reilly Kiadó sendmailról szóló könyvét és két hét kutatómunka után rájöttem, hogy a megoldás már két és fél éve létezik, és tökéletesen alkalmazható. Mivel címenként havi öt dollárt takaríthatunk meg, és tizenöt felhasználónk van, ezért úgy gondolom, hogy megérte a fáradságot.

Következzen a terv (a neveket természetesen én találtam ki):

- fukarcég@szolgáltató.com – a vállalatom levélcíme, amit a szolgáltatótól kapott.
- smtp.szolgáltató.com – a szolgáltató SMTP kiszolgálója (a kimenő levelek számára).
- pop3.szolgáltató.com – a szolgáltató POP3 kiszolgálója (a bejövő levelek számára).
- linuxgép.fukarcég.com – a fájlkiszolgálónk (a fukarcég.com kizárólag belső használatra van fenntartva, ez nem bejegyzett név).
- mrpkiszolgáló.fukarcég.com – ez egy Sun gép, melyen az MRP rendszer található. Ez nem elengedhetetlen része a tervnek, viszont a legtöbb felhasználó XTERM-mel csatlakozik e gépre, ezért miattuk telepítettem egy Pine-t a Sun gépre. A Pine a Pine Is Not Elm rövidítése (ezek az értelmes nevek…), és egy szöveges alapú levelezőprogramot takar. A Pine rendkívül jól használható levelezőrendszer, én ezt szeretem a legjobban. Ha grafikus programot használnék, akkor napokba telne a leveleim átnézése.
- fukarcég@linuxgép.fukarcég.com – ez egy álca levélcím a linuxos gépen. Az összes bejövő levél ezen a címen halad át, s ezeket a procmail osztja szét a felhasználók között.

Kicsit bűvészkedtem a /etc/sendmail.cf fájllal, és elkészítettem egy procmail szűrőt a bejövő oldalon, és lám, a felhasználók ugyanazt a levélcímet használva is biztonságosan, illetéktelen személyek hozzáférése nélkül kezelhetik levelezésüket.

1. lista A „drága” levelek meghatározása és visszatartása

```
O HoldExpensive=True
#####   @(#)smtp.m4       8.33 (Berkeley) 7/9/96   #####

Msmtp,  P=[IPC], F=mDFMuXe, S=11/31, R=21, E=\r\n, L=990,
        T=DNS/RFC822/SMTP,
        A=IPC $h
Mesmtp, P=[IPC], F=mDFMuXae, S=11/31, R=21, E=\r\n, L=990,
        T=DNS/RFC822/SMTP,
        A=IPC $h
Msmtp8, P=[IPC], F=mDFMuX8e, S=11/31, R=21, E=\r\n, L=990,
        T=DNS/RFC822/SMTP,
        A=IPC $h
Mrelay, P=[IPC], F=mDFMuXa8, S=11/31, R=61, E=\r\n, L=2040,
        T=DNS/RFC822/SMTP,
        =IPC $h
```

A kimenő levelekkel az alábbiak történnek:

1. A felhasználó megírja a levelét a Sun Pine-jával, vagy az Outlookkal, esetleg a Netscape levelezőjével. Csak azok számára hozok létre a grafikus programokban fiókot, akik csatolt fájlokat várnak vagy küldenek.
2. Ha a levelet a PC-n írják, akkor az először a Sunra kerül, és innen is olvassa be a beérkező leveleket.
3. Ha egy belső címről van szó, akkor a Sun gép továbbítja a levelet.
4. Ha a cím külső, akkor a levél a linuxos gépre kerül.
5. A linuxos gép a kimenő leveleket sorba rendezi és félóránként továbbítja, miután a bejövő leveleket letöltöttük.
6. A megfelelő időpontban lefut a `/usr/sbin/sendmail -q -v` parancs, mely alatt a /etc/genericstable.db fájl adatai és a /etc/sendmail.cf bizonyos álcázási szabályainak felhasználásával a felhasználó válaszcímét a „vezetéknév keresztnév <fukarcég@szolgáltató.com>” alakra módosítjuk.

2. lista A sendmail felépítése m4 forrásokból

```
# 93. szabály - a fejlécneveket a álcázott
# alakra írja át

S93

# a tartományfüggő álcázás kezelése
R$* < @ $=M . > $*       $: $1 < @ $2 . @ $M >
$3

# 94. szabály - a borítékneveket is álcázza

S94
R$+                      $@ $>93 $1
```

3. lista A sendmail.cf fájl

```
# 93. szabály - a fejlécneveket az álcázott
# alakra írja át

S93

# a generics adatbázis kezelése
R$+ < @ $=G . >      $: < $1@$2 > $1
➥< @ $2 . > @     mark
R$+ < @ *LOCAL* >    $: < $1@$j > $1
➥< @ *LOCAL* > @ mark
R< $+ > $+ < $* > @  $: < $(generics $1 $: $) >
➥$2 < $3 >
R< > $+ < @ $+ >     $: < $(generics $1 $: $) >
➥$1 < @ $2 >
R< $* @ $* > $* < $* >      $@ $>3 $1 @ $2
➥found qualified
R< $+ > $* < $* >    $: $>3 $1 @ *LOCAL*
➥found unqualified
R< > $*        $: $1  not found
```

A bejövő levelek kezelése a következőképpen történik:

1. A kiszolgáló a cronban meghatározott időpontban a szolgáltatóhoz kapcsolódik (ha már van kapcsolat, akkor természetesen nem). Ezt a pppd és a diald kezeli, de ezekre most nem térek ki, hiszen nem kapcsolódnak szorosan a témánkhoz.
2. A bejövő leveleket letöltjük a `fetchmail` paranccsal, majd az alapértelmezett fiókhoz (fukarcég@linuxgép.fukarcég.com) továbbítjuk azokat.
3. A fiók procmail szűrője a levelek fejlécében megkeresi a neveket, és ennek megfelelően szétosztja a leveleket a felhasználók között. Ha valamelyik levél nem tartozik sehova, az alapértelmezett fiókban marad. Ebben a fiókban, a biztonság kedvéért, minden felhasználó számára fenntartok egy mappát.
4. Minden helyi levelet, kivéve a *root* és a *fukarcég* felhasználókét, a Sun géphez továbbítunk.
5. A felhasználók leveleiket vagy valós időben kapják meg a Sun Pine-jában, vagy saját levelezőprogramjukkal töltik le azokat a Sunról.

Ez valószínűleg elég bonyolultnak tűnik elsőre, de igazából nagyon egyszerű az egész. Ne felejtsük el, hogy a Sun gép csupán kényelmi szempontból van jelen, de a rendszer működéséhez nem feltétlenül szükséges. Úgy is felállíthattuk volna a levelezést, hogy minden felhasználó a linuxos gépről töltse le a leveleit.

Akkor most nézzük meg a szükséges fájlokat. Ezek legtöbbje (ha nem az összes) megtalálható valamelyik Linux-változat CD-jén. Aki elakad valahol, az böngéssze át a *Kapcsolódó címek* részben található weboldalakat.

A helyi levelezés kezelése

Nézzük először a Sun és a linuxos gép közötti levéltovábbítást – akinek nincs szüksége a második kiszolgálóra, az hagyja ki ezt a részt.

A Sun /etc/mail/sendmail.cf fájljában nem változtattam semmit. Amikor ezt az egészet kiötlöttem, még nem ismertem eléggé a Solarist és nem akartam sokat bajlódni a központi gép beállításaival, szóval egyszerűen az alábbiak szerint oldottam meg a levélküldést.

Bejegyzés a /etc/mail/sendmail.cf fájlban:

```
DRmailhost
Crmailhost
```

Bejegyzés a /etc/hosts fájlban:

```
192.9.200.2 {{linuxgép}} mailhost*
```

A változtatások előtt a /etc/hosts fájlban a mailhost bejegyzés a Sun IP-címe után állt:

```
192.9.200.1 {{mrpkiszolgáló}} mailhost
```

Ezzel a változtatással a kimenő leveleket a linuxos gépre irányítottam.

A linuxos gépen beállítottam, hogy a helyi levelek a Sun gépre kerüljenek, ahol a felhasználók levélmappái találhatók. Rajtam kívül egyetlen felhasználó sem szokott a linuxos gépre közvetlenül bejelentkezni, csak a Sambával megosztott fájlokat használják. A telnetet és az ftp-t letiltottam:

```
DHrelay:mrpkiszolgáló.fukarcég.com
```

Kivétel természetesen a *root* és a *fukarcég* felhasználó, akik továbbra is ezen a gépen maradnak:

```
CL root fukarcég
```

Körülbelül ennyit kell tudnunk a két helyi unixos gép közötti fájlforgalomról. A helyi levelezés a Sun gépen marad, a külső rendszerekből érkező levelek a linuxos gépre kerülnek, majd sorba állítódnak a következő kapcsolathoz. A külvilágból érkező levelek átcímzés után a helyi felhasználókhoz kerülnek (a Sun gépre). A *root* és a *fukarcég* felhasználók levelei a linuxos gépen maradnak, ezeket mindennapi munkám során ellenőrzöm.

A külső levelezés

Ezen beállítások egy részét különböző HOWTO fájlokból szedtem össze az évek során, a többit meg az Internetről és a sendmailról szóló könyvekből.

A sendmail beállítása

Mivel itt behívásról van szó, nyilván nem szeretnénk, hogy a sendmail minden egyes levél elküldésekor kapcsolódjon a szolgáltatóhoz. Ezért az alábbiak szerint kell módosítanunk a sendmailt indító parancsfájlt.

Régi változat: `/usr/sbin/sendmail -bd -q15m`
Új változat: `/usr/sbin/sendmail -bd -os`

4. lista A levél továbbítása a megfelelő felhasználóhoz

```
PATH=$HOME/bin:/usr/bin:/usr/ucb:/bin:
      ➥/usr/local/bin:.
MAILDIR=$HOME/mail
# Biztos, ami biztos, ellenőrizzük, hogy
# létezik-e
DEFAULT=$MAILDIR/mbox
LOGFILE=$MAILDIR/from
LOCKFILE=$HOME/.lockmail

:0 c
    backup

:0 ic
      | cd backup && rm -f dummy `ls -t msg.*
      ➥| sed -e 1,32d`

:0 c
* ^TO*Stew Benedict
!stew

    :0 A
     stew

:0 c
* ^TO*Joe User
!joe

    :0 A
     joe

# az összes többi az alapértelmezetthez kerül
```

RedHat esetén ezt a /etc/sysconfig/sendmail és/vagy a /etc/rc.d/init.d/sendmail sorban kell állítanunk. Az *1. listán* látható /etc/sendmail.cf fájlban állíthatjuk be, hogy melyik levél számít „drágának”, illetve mit kell tennie ezekkel a levelekkel.
Figyeljük meg az smtp, esmtp és smtp8 *F=* részekben látható *e* karaktert. Ez a drága (expensive) jele és ezt lehagyjuk a helyi hálózatnál. Az Mlocal és Mprog részeknél sem szerepelhet ez a jel, hogy a helyi levelezés azonnal továbbítódjon.
A cron félóránként csatlakozik az Internetre és e feladat részeként a sorba állított leveleket is kiküldi:

```
/usr/sbin/sendmail -q -v
```

Most térjünk rá a kimenő levelek továbbítására. Mivel nem rendelkezünk érvényes tartománynévvel, módosítanunk kell a visszatérési címeket. Álcáznunk kell mind a visszatérési címet, mind a borítékot. Ahhoz pedig, hogy a feladó megkapja a válaszokat, a „From:” címet is át kell írnunk.

Álcázás

Ha a sendmail.cf fájlt m4 forrásokból építjük föl, akkor a helyi .mc fájlban szerepelnie kell az alábbiaknak:

```
MASQUERADE_AS(szolgáltató.com)
FEATURE(masquerade_envelope)
FEATURE(limited_masquerade)
FEATURE(genericstable)*
```

Csak a CM-ben meghatározott gazdákat álcázzuk. Ha csak a /etc/sendmail.cf fájlt szerkesztjük, akkor a következő sorokat így kell módosítanunk:

```
# amit álcázunk (ha semmi nincs itt, akkor semmit)
# (lásd még: $=M)
DMszolgáltató.com
```

Valószínűleg azt is szeretnénk, hogy a levelek a szolgáltatón keresztül érkezzenek, így a fogadó levélkiszolgálók azt látják, hogy a levelezés érvényes tartományból érkezik:

```
DSsmtp:smtp.szolgáltató.com
```

Határozzuk meg azon tartományneveket, melyeket az álcázott címre kell alakítani:

```
CG mrpkiszolgáló.fukarcég.com
CM mrpkiszolgáló.fukarcég.com
```

(Ha csak egy gépünk van, akkor itt a linuxgép.fukarcég.com legyen.)
Így a sendmail.cf fájlálcázást végző sorai egy kicsit összekavarodhatnak. Talán jobb megoldás, ha a sendmail.cf-et a *2. listán* látható m4 forrásokból építjük föl.
A kimenő forgalom kirakós játékának utolsó darabja a felhasználó visszatérési címének visszaírása. Ha a Sun gépen fogalmaznám a levelet, a Pine a következő visszatérési címet hozná létre:

```
Stew Benedict <stew@mrpkiszolgáló.fukarcég.com>
```

Ez így jól néz ki, de sajnos ez a tartománynév nincs közvetlenül az Interneten, így soha nem kapnék választ a leveleimre. Sőt, a legtöbb levelezőrendszer vissza is dobná az általam küldött leveleket, hiszen a levélcím nem létező tartománynevet tartalmaz.
Szóval az lenne az igazi, ha valahogy így nézne ki a cím:

```
Stew Benedict <fukarcég@szolgáltató.com>
```

Itt jön a képbe a sendmail genericstable nevű szolgáltatása. Ismét megjegyezném, hogy ha a sendmail.cf fájlt az m4 forrásokból állítjuk össze, akkor a következő sor a megoldás:

```
FEATURE(genericstable)
```

A sendmail.cf fájlba a következőket írjuk:

```
Kgenerics hash -o /etc/genericstable
```

A -o azt jelenti, hogy nem kötelező, tehát a sendmail nem fog leállni, ha nem találja a fájl. A helyi .mc fájl bővítése hozza létre a sendmail.cf fájlnak a *3. listán* látható részét.
A genericstable.db fájlt egy szöveges állományon alapul, melynek formátuma a következő:

```
stew    fukarcég@szolgáltató.com
joe     fukarcég@szolgáltató.com
```

Ez a fájl ezután a makemap programhoz kerül, mely db fájlt készít belőle:

```
makemap hash genericstable.db < genericstable
```

Ennyit a kimenő levelekről. A sendmail.cf fájl létrehozása vagy

5. lista A bejövő és kimenő levelek figyelése

```
# !/bin/sh
ST=/etc/sendmail.st
MS=/usr/sbin/mailstats
MSO=/tmp/mailstats.txt
if [ -s $ST -a -f $MS ]; then
      echo "Általános levelezési adatok" > $MSO
      echo "" > $MSO
      echo "helyi = A (linuxserver) nevű helyi
      ➥fájlkiszolgálóhoz tartozó levelek" > $MSO
      echo "smtp  = internetes levelek" > $MSO
      ➥echo "relay = Az (mrpserver) nevű Sun
      gépre/gépről érkező levelek" > $MSO
      echo "" > $MSO
      $MS > $MSO
      cp /dev/null $ST
fi
echo "" > $MSO
echo "Levélszűrés és -továbbítás" > $MSO
echo "" > $MSO
/usr/bin/mailstat -l
➥/home/thriftycompany/mail/from > $MSO
chown thriftycompany
➥/home/thriftycompany/mail/from
cat $MSO | mail -s "Napi e-mailösszegzés"
➥myboss stew
rm $MSO
exit 0
```

módosítása és a genericstable.db elkészítése után újra kell indítanunk a sendmailt. Egy RedHat-alapú rendszerben ezt a

```
/etc/rc.d/init.d/sendmail restart
```

paranccsal végezhetjük el.

A külvilágból érkező levelek

A bejövő leveleket a fetchmail segítségével töltjük le a szolgáltatótól. Az általam beállított cron feladat létrehozza a kapcsolatot, majd egy ehhez hasonló parancsot futtat:

```
su -c "/usr/bin/fetchmail -a -f /home/fukarcég"
```

A .fetchmailrc fájl tartalma:

```
poll pop3.szolgáltató.com proto pop3 user
➥fukarcég password JELSZÓ
```

A sendmail.cf fájl beállításai szerint a procmail a helyi MDA (levélkézbesítő):

```
Mlocal,       P=/usr/bin/procmail,
                ➥F=lsDFMAw5:/|@qShP, S=10/30,
                ➥R=20/40,
              T=DNS/RFC822/X-Unix,
              A=procmail -a $h -d $u
```

Az összes bejövő levél a fukarcég fiókba kerül, melynek .procmailrc fájlja a levelek fejlécének „To:” sorát olvassa be, ez alapján dönti el,

6. lista A cron feladat eredménye

```
Date: Mon, 2 Oct 2000 05:00:01 -0400
From: root@linuxserver.thriftycompany.com
To: stew@linuxserver.thiftycompany.com
Subject: Napi e-mailösszegzés

Általános levelezési adatok

helyi = A (linuxserver) nevű helyi
       ➥fájlkiszolgálóhoz tartozó levelek
smtp  = internetes levelek
relay = Az (mrpserver) nevű Sun gépre/gépről
       ➥érkező levelek

Statistics from Fri Sep 29 05:00:06 2000
M  msgsfr  bytes_from  msgsto  bytes_to  Mailer
3      81       5004K      56     3775K   local
4       1         98K       6      102K    smtp
7       0          0K      14      132K   relay
=============================================
=
       T          82   5102K        76   4009K

Levélszűrés és -továbbítás
Össz      Átlag      Szám              Könyvtár
-----    -------    ------              ------
3755       1877            /home/ali/mail/mbox
1759       1759         1                 alex
47208     23604         2                 dave
5912       2956         2                laurie
1464       1464         1                marge
27616     13808         2                 stew
```

hogy az adott levelet melyik felhasználóhoz kell továbbítani (*4. lista*). E procmail szűrőt az ILOVEYOU típusú vírusok elleni védelemre, a csatolt fájlok korlátozására vagy tiltására, és még sok más hasznos dologra egyaránt használhatjuk. A procmail leírásában bővebben is olvashatunk erről. A felhasználók levelei a fukarcég fiókban, csoportosítva tárolódnak, így a véletlen törlés ellen is védettek. Ezeket a könyvtárakat alkalomszerűen én magam törlöm.

A naplózás

A főnököm még kissé tétova és bizalmatlan az Internettel kapcsolatban és megkért arra, hogy készítsek teljes körű kimutatást a kapcsolat kihasználtságáról. Ennek egy része, hogy az összes bejövő és kimenő levél méretéről és számáról feljegyzést készítek, minden egyes felhasználóra külön-külön. Ezt egy reggelenként lefutó cron parancsállomány végzi (*5. lista*), melynek eredménye a *6. listán* látható.
Körülbelül ennyiről lenne szó. A dolog húzós része az, hogy a levelező feleket valahogyan rá kell venni arra, hogy megfelelően címezzék meg a cégünkhöz küldött leveleket. A legtöbb levelezőprogram esetében ehhez mindössze annyi szükséges, hogy a vezetéknevet, a keresztnevet és az e-mailcímet a címjegyzékben tárolják, az illető helyes nevével és a szolgáltatónktól kapott levélcímmel. Az olyan munkatársaim számára, akik sorozatosan kapnak nem megfelelően címzett leveleket, bevezettem egy procmail szabályt, mely a „From:” mezőket is ellenőrzi. Azt is javasoltam, hogy mindenki küldjön levelet az összes ismerősének az új beállításokkal, és kérjék meg őket arra, hogy e levél feladóját tárolják a címjegyzékükben.

Karbantartás

A rendszer kevés karbantartást igényel. Írás közben éppen azon töröm a fejem, minként lehetne e feladatokat is önműködővé tenni, de igazából nem jelent túl nagy terhet az egész.
A fukarcég fiókba érkezett és a procmail szűrőn átjutott leveleket naponta átnézem. Ez sem olyan nagy munka, de akár tovább is küldhetném magamnak későbbi átnézésre, esetleg a KBiff nevű levéljelző programmal is ellenőrizhetném ezt a fiókot.
A fukarcég fiókban lévő, felhasználónként csoportosított régi leveleket néha törlöm. Ez is lehetne önműködő, biztosan létezik valahol egy Perl parancsfájl, mely az „n” napnál régebbi leveleket törli.
Új felhasználó hozzáadásakor pillanatnyilag a genericstable.db fájlt is új bejegyzéssel kell bővítenem, futtatnom kell a `makemap` parancsot, és egy új procmail szűrőt is be kell állítanom. Ezt is önműködővé tehetném, de minek, hiszen évente legfeljebb négy új felhasználónk van. Remélem, hogy a fentiek alapján bárki képes felállítani hasonló rendszert olyan kisebb vállalkozás számára, mely még nem érzi érettnek magát arra, hogy DSL vagy T1 kapcsolatot és tartománynevet vásároljon. Ha az Internet arra kell, hogy vásárlóinkkal, üzletfeleinkkel kapcsolatba léphessünk, akkor ez lesz a legtakarékosabb és a legválasztékosabb megoldás.

Kapcsolódó címek

sendmail:
➲ http://freshmeat.net/projects/sendmail/homepage/

makemap: ez általában a sendmail része
➲ http://freshmeat.net/projects/sendmail/homepage/

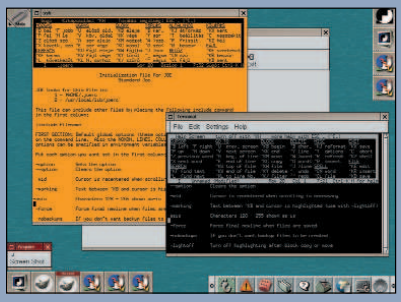
procmail:
➲ http://freshmeat.net/projects/procmail/homepage/

fetchmail:
➲ http://freshmeat.net/projects/fetchmail/download/

Stew Benedict
egy clevelandi autógyártó cég rendszergazdája, emellett szabadúszó tanácsadó, és az AYS Enterprises vezetője. Ez a cég nyomtatott áramkörök tervezésével, adatbázisokkal foglalkozik, s a Linuxra a drága operációs rendszerek és programok felváltójaként tekint. 1994 óta használja és népszerűsíti a Linuxot. Amikor nem a monitor előtt sütkérezik, akkor felesége, kislánya és két kutyája társaságában múlatja az időt a Tennessee állambeli Norris-tóra néző otthonában.

Szerkesztők háborúja I.

Mindenki esküszik valamilyen szerkesztőre. Én a `joe`-t kedvelem. Hogy miért? Első találkozásunk a következőképpen zajlott. Fejemet csóválva ültem a gépteremben, és minden arra járót megkérdeztem, hogy ezt vagy azt a műveletet hogyan lehet megoldani vi-ban. Egy ismerősöm jót nevetett, és ajánlotta, hogy inkább használjak más szerkesztőt, például a `joe`-t. Azt mondta, az legalább felhasználóbarát. De miben felhasználóbarátabb, mint más karakteres szerkesztő? – kérdeztem. Legalább kiírja, hogy hogyan jeleníti meg a súgót.



Lássuk, mit is tud ez a kis csoda! Karakteres üzemmódban dolgozó szerkesztőprogram. Nem színes, nem grafikus indítása egyszerű:

```
joe {{fájlnév}}
```

Ha már eddig eljutottunk, a legfontosabb parancs a Ctrl+C, ezzel tudunk ugyanis kilépni a programból (a vi-ból a :q begépelésével, majd az Enter lenyomásával léphetünk ki). Amit még ismernünk kell ezenkívül, azt a felső állapotsor kiírja: a Ctrl+K,H kombinációra jelenik meg a Súgó. A következő oldalra az Esc, majd a . (pont) lenyomásával léphetünk, az előzőre az Esc, majd a , (vessző) billentyűkkel.

Beállítás

Sokkal izgalmasabb a program beállítása. Két fontos fájlt kell itt megemlíteni, a /etc/joe/joerc-t, valamint a felhasználói könyvtárban lévő .joerc-t. Ha a program induláskor talál .joerc fájlt, akkor már nem is keresi meg a központi beállítófájlt, úgyhogy ha egyéni beállításokat akarunk használni, először másoljuk le a központi fájlt, majd azt változtassuk.

Ha például azt szeretnénk, hogy a program mindig a `-asis` kapcsolóval induljon (e nélkül a joe a 128 feletti karaktereket alsóbb karakterekkel, inverzben jeleníti meg), keressük meg a "-asis" kezdetű sort, majd töröljük a sor elejéről a szóközt. Ezzel a módszerrel kapcsolhatjuk be a különböző szolgáltatásokat. De mi történik, ha egy adott fájlnál szeretnénk kikapcsolni a szolgáltatást? Például ha ki szeretnénk kapcsolni az önműködő sortörést az összes .sh kiterjesztésű fájlnál? Ehhez megoldás lehet, ha fájlunk második szakaszába beírjuk a következő két sort:

```
*.sh
--wordwrap
```

Egyéb lehetőségek

Szintúgy a beállítási fájlban található az egész Súgó is, ezt mi magunk is vígan átírhatjuk. Én is nekifeküdtem és nagy vonalakban lefordítottam a Súgót, ezt bárki letöltheti a ➲ www.linuxvilag.hu/cikkek/joe/joerc-MINTA címről. Emellett fontos megjegyezni, hogy a parancsokat is szabadon átrendezhetjük, és a joe rendelkezik makrórögzítő résszel is. Próbáljuk is ki! Ctrl+K, majd "]" billentyű, és adjunk egy 2-est az új makró sorszámának (0 és 9 között választhatunk). Ezután írjuk be a "Linuxvilág" szót, és zárjuk le a rögzítést a Ctrl+K,] billentyűkkel. Kész is! Bármikor futtatható a Ctrl+K,2 billentyűkkel.
Nem elég nekünk kétszer a felirat? Próbáljuk ki a következőt: Ctrl+K,\,3, majd Ctrl+K,2. Hopp, három Linuxvilág! És ez még csak a kezdet!

Szy György

Weboldal kialakítása (2. rész)

A GIMP alkalmazásának bemutatása valós példákon keresztül.

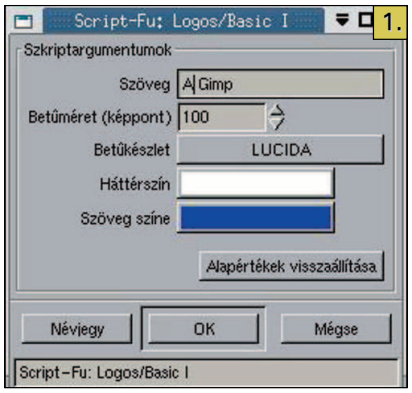
A cikk első részében egy weboldal kialakításának grafikai munkáit végzem el. A mintapéldák kiválasztása, elkészítési módja a GIMP használatát hivatott szemléltetni. Több megjelenítést is készítek: gyors megoldást a beépített függvények segítségével, 3D-s felületet, a jelenleg legszebb MacOS X grafikus felületének formáját, majd később áttekintem a szolgáltatások írásának lehetőségét a Script-Fu segítségével. Miután elkészítem a weboldal kezelői felületét, megalkotom a fényképgyűjteményembe szánt képeket. Bemutatom, milyen grafikai hatásokat lehet használni, majd továbbfejlesztem a weboldalt.
A munka során mindig az utolsó állandó GIMP változatot használom (jelenleg ez az 1.2.1). A menük és a szolgáltatások a jelenleg fordítás alatt álló magyar változat szerintiek, még találunk bennük egy-két angol szót.

Az oldal tervezése

Első feladatként egy GIMP-pel foglalkozó weboldal kialakítását mutatom be. Az első és legfontosabb lépés az oldal tartalmának és szolgáltatásainak megtervezése. Ezt nem szabad kihagyni, elnagyolni, hiszen ezen áll vagy bukik az oldal használhatósága. Ha körülnézünk az Interneten, rengeteg elrettentő példát is találhatunk, a sok szép, jól megtervezett és kialakított oldal mellett. Írásom témája azonban nem a weboldaltervezés, ezért csupán vázolom a szükséges teendőket.
Első lépésként meg kell fogalmazni, hogy milyen adatokat szeretnénk megjeleníteni az oldalon. Én az alábbi témaköröket választottam: bemutatás, ismertetők, galéria, kapcsolatok, kérdések és válaszok, vendégkönyv.
Ezt követően – a kialakítás ismeretében – meg kell tervezni az oldal megjelenését. Általában két módszert használunk, az egyik a hagyományos (papír és ceruza használatával), a másik pedig grafikai program segítségével történő tervezés. Mindkét módszernek megvan az előnye és hátránya is. Én a második módszerrel dolgozom, melyhez természetesen a GIMP-et használom.
Nézzük meg, hogyan!
Ezt az oldalt a 800x600-as felbontásra készítem el. Létrehozok egy új képet: a szélessége 740 képpont, a magassága 600 képpont, a képtípusa RGB, a kitöltés típusa pedig *háttérkép*. Ezzel létrehoztam a leendő weboldal képernyőterületét. A GIMP-ben található segédvonalak segítségével nagyon jól elválaszthatók egymástól a különböző területek. A segédvonalakat a *mozgatás* eszközzel lehet beállítani. Igazítsuk be a segédvonalakat vízszintesen 90, 100, 120, 140 képpontra, valamint függőlegesen 200, 220 képpontra. Új vonalat létrehozhatunk, ha a vonalzóra állva lenyomjuk a bal egérgombot, majd a kívánt helyre húzzuk az új vonalat. A vonalzó megjelenítését a *Nézet/Vonalzó be-ki* menüponttal lehet kapcsolgatni. A pontos beállításában hatékony segítséget nyújt a képernyő bal alsó sarkában található tájékoztató ablak. Ha nem jelennének meg a segédvonalak, akkor a *Nézet/Segédvonalak ki-be* menüpont állását kell ellenőrizni. Ezután a főbb területek kijelölése és a megfelelő területek finomítása következik.

Logó, felirat készítése

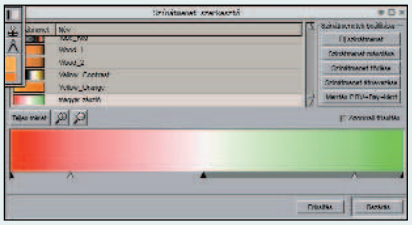
Elsőként tegyük helyére a logót. Ha már van kész logónk, nyissuk meg. Ha nincs, akkor a GIMP segítségével könnyedén készíthető logó vagy felirat. A legfrissebb megbízható

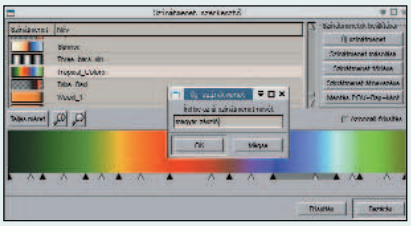


változatban 29 különböző, jól paraméterezhető feliratkészítő eszköz található (*lásd például az előző számunk 87. oldalán látható 12. képen a „bovination" típust*). Nagyon hasznos időtöltés végigböngészni ezeket az eszközöket, kipróbálgatni a lehetőségeiket. Így rengeteg időt takaríthatunk meg későbbi munkánk során, hiszen tudjuk, hogy melyik feladatra melyik eszközt érdemes használni. Az újabb változatok megjelenését követően

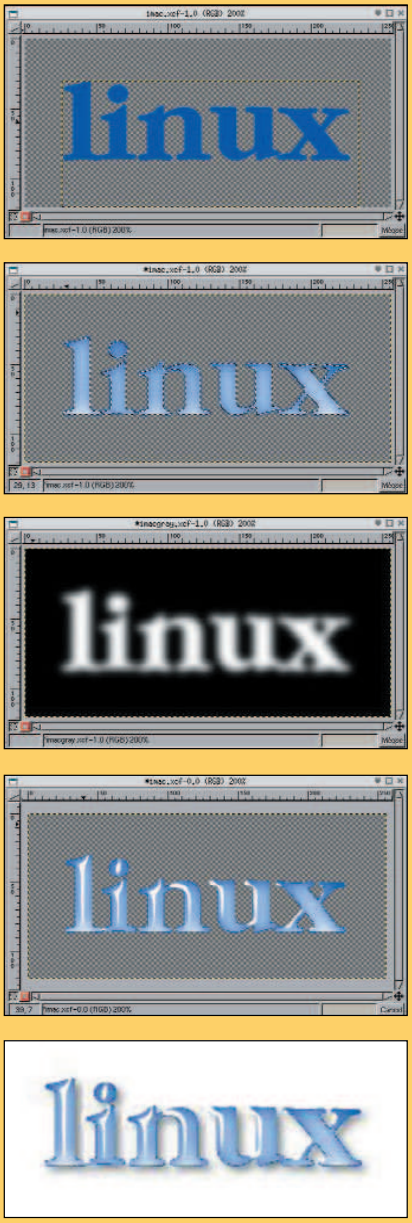
Saját színátmenet

Állítsunk elő egyéni, magyar zászlót jelképező átmenetet. A színátmenet-kezelőben kattintsunk az *Új színátmenet* menüre, majd írjuk be a saját színátmenetünk nevét: „Piros-Fehér-Zöld". Láthatjuk, hogy a színátmenet-kezelő egy fekete-fehér átmenetet készített.





Az elkészült színátmenetek alatt találunk egy nagyon hasznos eszközt, a nagyítót. Ha összetett színátmenetet szeretnénk készíteni, ránagyíthatunk a számunkra fontos részletre. Az egerünk jobb gombjával tudjuk előhozni azt a menüt, ahol a színátmenet bal és jobb szélének a színét lehet beállítani. Ha új színt szeretnénk keverni, kattintsunk a színre (például *Bal végpont színe*), s a már megismert színkeverő menüben találjuk magunkat.
Készítsük el a háromszínű színátmenetet. A bal végpont legyen piros, a jobb végpont legyen zöld színű. A három színből álló átmenet létrehozásához a *Szelet kettévágása középpontban* menüpont segítségével oszszuk ketté a színátmenetünket. Állítsuk be a színátmenet közepének színét fehérre. Az egyenletes hatás eléréséhez, húzzuk a színátmeneteink középső határolóját 1:3, 2:3 arányban a szelet színt tartalmazó végéhez.



iMac stílusú gomb készítése

Nézzük meg az iMac stílusú térbeli, domború hatású feliratok készítésének módját.
1. Készítsünk egy új képet, átlátszó háttérrel (kitöltés típusa). A kép mérete 250x120 képpont. Ezután írjuk bele az alábbi szöveget 80 képpont mérettel.
Jelöljük ki a szöveget, hozzunk létre egy új, átlátszó réteget, csökkentsük a kijelölést egy képponttal (*Kijelölés/Csökkentés*), majd töltsük ki fehér színnel, és alkalmazzunk egy Gauss-elmosást (*Szűrők/Elmosás/Gauss-elmosás*) körülbelül 18-as értékkel.
Hozzunk létre egy új réteget a kép felső szintjén és töltsük ki feketével. A rétegnél állítsuk be a *Mód kivetítésre* (screen) szolgáltatást. Ez lesz a kiemelésünk.
Ezután hozzunk létre egy új szürkeárnyalatos képet. Írjuk bele a szöveget, és alkalmazzunk ismét Gauss-elmosást 8-as értékkel.
Lépjünk a kiemelési rétegünkhöz, valamint használjuk a lighting bővítményt. Válasszuk ki a *Bump Map* képet:
max height=0.05, min height=0;
Directional light (-1, -0.5, 0.5);
Intensity Levels: Ambient=5,
Diffuse=5,
Reflectivity: Diffuse=10, Specular=1,
Hightligh=10.
Ezután nincs más hátra, mint árnyékot adni a feliratunknak (*Drop Shadow*):
OffsetX és Y: 3,
Blur radius: 8,
Color: fekete,
Opacity: 50%,
értékekkel és lapítsuk (flatten) a képet.
A fentiek alapján próbálgassunk ki többféle beállítást is. Ezzel a módszerrel könnyedén elkészíthetjük feliratainkat a weboldalhoz.

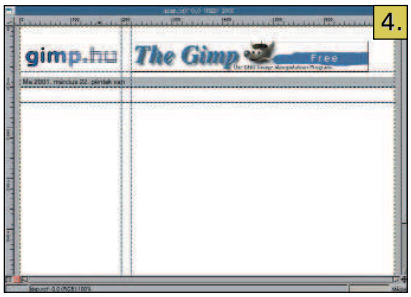
célszerű megnézni a frissen napvilágot látott eszközöket is. A cikk terjedelme sajnos nem teszi lehetővé minden feliratkészítő függvény részletes ismertetését. Természetesen kipróbálunk közülük néhányat. Válasszuk ki a *Basic I.* típust (*1. kép*)
A parancsfájl beállításai magukért beszélnek:

- *Szöveg* – a logón megjelenő felirat szövege,
- *Betűméret* – a felirat betűmérete képpontban,
- *Betűkészlet* – a kívánt betűkészlet típusa, válasszuk ki a *Betűkészlet-választó* segítségével a felirat típusát, stílusát.
- *Háttérszín* – a felirat hátterének színe.
- *Szöveg színe* – a felirat színe. Abban az esetben, ha a beállításkor elrontottunk valamit, az *Alapértékek visszaállítása* segítségével visszatérhetünk az eredeti beállításra. Egy fontos tulajdonságra azonban fel kell hívnom a figyelmet: ez a szolgáltatás a felirat színét balról jobbra sötétíti. A végeredmény a *2. képen* látható. Nézzünk meg egy másik, némiképp összetettebb feliratkészítő eszközt.

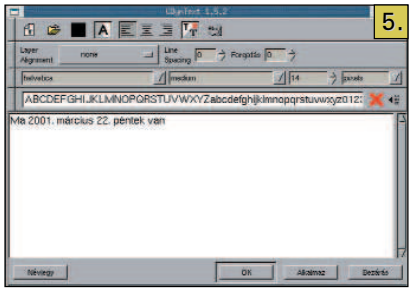
Nagyon látványos és egyedi logót lehet készíteni a *Cool Metal* parancsfájllal is. Ennél az eszköznél hasonló lehetőségekkel találkozunk, újdonsága a *Színátmenet* kezelése. Ezzel a lehetőséggel tudjuk a feliratunk színét beállítani. Szerencsére hosszú listából választhatjuk ki a feliratunkhoz leginkább

illő átmenetet. Ha mégsem találunk kedvünkre valót, akkor magunk is készíthetünk. Az elkészített színátmentettel készítsük el feliratunkat a korábban említett *Cool Metal* típussal. (*3. kép*).

Ha elkészült a logónk, vagy már megnyitottunk egyet, a *Rétegek, csatornák és útvonalak* (Párbeszédablakok/Rétegek, csatornák és útvonalak) ablakban kiválasztjuk a *Képek* közül a logónkat tartalmazó képet. A *Rétegek* között megtaláljuk a logót, amit csak át kell húzni a weboldaltervünkre. A mozgatás eszközzel igazítsuk helyére a logónkat a készülő weboldalunkon (*4. kép*).



Az oldal második sorát, a 100–120 képpontok közti területet teljes szélességben jelöljük ki a *téglalap kijelölése* eszköz (rectangular selection) segítségével. A *Színválasztás* menüben válasszunk ki egy semleges szürke színt, majd a kijelölt területet a *kitöltés* eszközzel színezzük. Ezt követően a *szöveg* eszköz fekete betűjével írjuk a keltezést a szürke területre. Az új GIMP-változatban (1.2.x) jelent meg a dinamikus szövegkezelés (dynamic text, *5. kép*), ennek segítségével a már elkészített feliratainkat, szövegeinket is módosíthatjuk. Választhatunk igazítást, betűszínt, forgatást, sortávolságot, betűélmosást, új réteg létrehozását, valamint a betűtáblázatból is válogathatunk. Elmondhatjuk, hogy a GIMP 1.2.x megfelel azoknak az igényeknek, amit egy retusálóprogramtól elvárhatunk. Még akadnak benne hibák (például még nem tökéletes a magyarítása), de ezek nem akadályozzák a használatát.

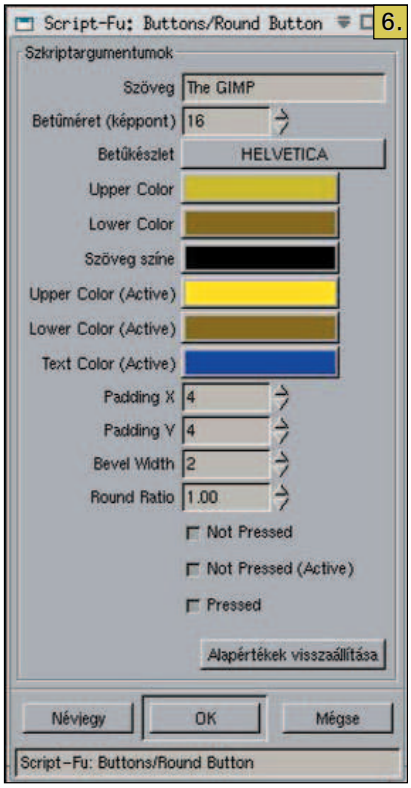


A menük

Miután sikeresen elkészítettük az oldal fejlécét, kicsit hátradőlhetünk és gyönyörködhetünk benne. A következő lépés a menü kialakítása. Erre is számos lehetőségünk nyílik.

1. A menüt sima szövegként írjuk ki. Ez nem túl szép megoldás, és a GIMPre sincs szükség hozzá.
2. A menüt gyorsan, meglévő eszközök segítségével szeretnénk elkészíteni. Ekkor a GIMP-be épített kiterjesztések (kit.), Script-Fu kiegészítők, előre megírt utasítássorok alkalmazásával készíthetjük el a menüpontokat.
3. Mi magunk is megalkothatjuk az igényeinknek legmegfelelőbb menüt. Például a 3D-s, az Aqua kinézetű gombokat, és még sorolhatnám. Ezek látványának, szolgáltatásainak csak a képzeletünk és a rendelkezésre álló idő szab határt.
4. Az előbbi pontban ismertetett tulajdonságok szerint Script-Fu függvényt írunk. Amivel gyorsan egységes megjelenésben tudjuk „gyártani” a gombokat. Nem kell megijedni tőle, nem olyan nehéz feladat.

Röviden áttekintjük a beépített gombkészítő eszközöket. (Nézzük át alaposan a GIMP *Irányítópult/kit.* menüjében a részleteket.) A jelenlegi változatban két gombkészítő függvény van, melyek a *kit./Script-Fu/Buttons* alatt találhatók, sajnos a fordításuk még nincs teljesen készen. A *Simple Beveled Button* az egyszerűbb. A szöveg, a betűméret, a betűkészlet és a szöveg színe vélhetően nem szorul magyarázatra. Az *Upper-left color* (bal felső szín) és a *Lower-right color* (jobb alsó szín) értékeivel a gomb hátterének a színét tudjuk beállítani. Ha egyszínű gombot szeretnénk, akkor állítsuk mindkét színt azonosra. A *Padding* értékével lehet állítani a szöveg körüli keret méretét. Ezt célszerű az alapértelmezett két képpontról nagyobb értékre állítani. A *Bevel Width* (ez a gomb magasságát jelenti) értékkel az árnyék szélességét állíthatjuk. A *Pressed* kapcsolóval tudjuk a gombot „benyomni”. Ha egy működő, mozgó gombot szeretnénk készíteni, akkor ezzel a függvénnyel két képet kell készítenünk.



A másik függvény (Round Button) jóval több állítási lehetőséget kínál (*6. kép*). Ezzel a szolgáltatással egyszerre elkészíthetjük a gombunk háromféle változatát: az alaphelyzetben lévőt, a működőt (*Active*) és a benyomott állapotban lévőt (*Pressed*). Nézzük meg ezeket. A szöveg, a betűméret és a betűkészlet nem szorul magyarázatra. A gombok színének meghatározásához két értéket kell beállítanunk. A felső színt *Upper Color* és az alsó színt *Upper Color* értelemszerűen, ha a páros értékek azonosak, egyszínű gombot kapunk. A szöveg színe mellett állíthatjuk a működő gombon levő szöveg színét is a *Text Color (Active)* értékével. A *Padding X* és *Y* értékével állíthatjuk be a gomb távolságát a szövegtől. A *Bevel Width* mint az előzőekben, az árnyék szélességét változtatja. Újdonság, hogy a *Round Ratio* értékével a legömbölyítés mértékét tudjuk módosítani. Minél nagyobb ez az érték, annál elnyújtottabb gombot kapunk. Három kapcsolót találunk az értékek alatt: a *Not Pressed*, a *Not Pressed (Active)* (nem benyomott gombok alaphelyzetű és működő változata) és a *Peressed* (benyomott) változatot. Itt tudjuk kiválasztani, hogy milyen gombokat szeretnénk készíteni, de legalább egyet ki kell választani ahhoz, hogy létrejöjjön a gomb. A két szolgáltatás közül mindenki kiválaszthatja a számára legmegfelelőbbet.

A következő részben az Aqua kinézetű gombok elkészítésével foglalkozunk, valamint saját Script-Fu függvény alkotásával. Addig is mindenkinek jó kísérletezést!

➲ gimp.hu

Süveg Gábor
Régóta használ Linuxot és BSD-t. Hobbija a búvárkodás, vitorlázás és a számítógépes grafika. Elérhető a (gsuveg@sgmobil2000.hu) címen.

Magyar karakterek az X alatt

Röviden megnézzük, hogy miként tudunk X alatt magyar TrueType betűkészleteket használni.

Először is nézzük át, hogy milyen betűkészletek léteznek. Két fő tulajdonságot kell megvizsgálni, az egyik a betűkészletet tartalmazó fájl(ok) formátuma, a másik pedig az alkalmazott kódlap. Számos formátum létezik, két fő csoportra osztjuk őket, a méretezhetőkre (scalable), és a kötött méretűekre. A nem méretezhető betűtípusok csak a megadott méretben (általában 10, 12, 14 pont) szépek, nyomtatásban sokszor szörnyűség jelenik meg helyettük. Ezeket a típusokat elsősorban a rendszer használja a menük, fejlécek és hibaüzenetek megjelenítésére (bdf, pcf, fon fájlok). A méretezhető betűtípusok között a legismertebb az Adobe Type 1, ezt régen főként a nyomdaiparban használták, ugyanis nyomtatásban rendkívül tetszetős (feltéve, hogy a betűtípus rendesen el van készítve). E két véglet között átmenetet képvisel a TrueType, mely vektoros alapú és méretezhető, és mind monitoron, mind nyomtatásban mutatós. A TrueTypeon belül is többféle rendszer létezik.

A kódolás határozza meg, hogy a megadott sorszámhoz a betűkészletben milyen betű tartozik. Ez számunkra azért különösen fontos, mert például a hosszú ű betű az egyik kódolásban a 251-es sorszám alatt található, a másikban viszont teljesen másutt van. Sőt, előfordulhat, hogy egy kódrendszerben nincsen helye a magyar ű betűnek. A helyzetet tovább bonyolítja, hogy a betűtípusok alkotói – annak ellenére, hogy a nekünk megfelelő kódrendszert használják – gyakran nem készítik el más nyelvek egyedi betűit.

Írásunkban kifejezetten a TrueType betűkkel foglalkozunk, és feltételezzük, hogy jó kódolással készült (például ISO8859-2), magyar betűket is tartalmazó készletet kívánunk telepíteni. Honnan lehet ilyet szerezni? Nos, ha valakinek van hivatalos Windows rendszere a gépen, akkor könnyű a dolga. Ha nincs, akkor vagy az Interneten kotorászik (lásd például a Kapcsolódó címek között), vagy beugrik egy boltba, és vásárol egy betűkészleteket tartalmazó CD-t. Ilyen például a WoodStone Kiadó Magyar fontok című lemeze (bolti ára kb. 6000 forint,a kényelmesebbek megrendelhetik a könyvesboltban is – lásd *Kapcsolódó címek*). Az ehhez hasonló gyűjtemények rengeteg ingyenes vagy szabadon használható betűkészletet tartalmaznak. Azt viszont el kell mondanom, hogy az ilyen gyűjteményekben található készleteknek jó, ha a fele rendesen működik! Ha valaki profi betűkészleteket szeretne vásárolni, mélyen a zsebébe kell nyúlnia, egy-egy komoly betűkészlet ára ugyanis elérheti akár a százezer forintot is. Ha a nyomtatónkhoz kaptunk lemezeket, azokat is nézzük végig, ugyanis gyakran tartalmazzák a nyomtató által ismert betűtípusok ttf vagy Type1 változatait. Ha Windows alatt is dolgozunk, elugorhatunk a Microsoft ftp-kiszolgálójára, és további betűtípusokat is letölthetünk (lehet, hogy hasznosnak találjuk majd a cabextract programot). Tehát megvannak a használni kívánt betűtípus állományai, ezeket célszerű egy könyvtárba összegyűjteni (például `/usr/local/myfonts`). A további lépések függenek attól, hogy az X melyik változatát használjuk. Nézzük először az új, 4-es változatot.

XFree86 4 alatt

Az állományokat tartalmazó könyvtárban adjuk ki az alábbi parancsot:

```
ttmkfdir > fonts.dir
```

Ez a parancs elkészíti a `fonts.dir` fájlt, mely az X számára szükséges betűkészlet-meghatározásokat tartalmazza (lásd az *1. listát*). A program minden betűfájlhoz annyi bejegyzést készít, ahány kódtáblával használhatónak ítéli meg az adott betűkészletet, így érdemes törölni a használaton kívüli változatokat. Az `iso8859-1` és `2` végű

```
[/usr/local/myfonts]# cat fonts.dir
90
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-tcvn-5712
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-viscii1.1-1
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-microsoft-cp1255
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-tatar-cyr
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-microsoft-cp1251
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-koi8-u
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-koi8-r
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-9e
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-15
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-14
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-13
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-10
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-9
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-8
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-7
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-5
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-4
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-3
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-2
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-1
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-tcvn-5712
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-viscii1.1-1
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-microsoft-cp1255
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-tatar-cyr
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-microsoft-cp1251
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-koi8-u
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-koi8-r
```

```
[/usr/local/ujfonts]# cat fonts.dir
12
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso10646-1
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-2
ariali.ttf -monotype-Arial-medium-i-normal--0-0-0-0-p-0-iso8859-1
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-iso10646-1
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-iso8859-2
arialbi.ttf -monotype-Arial-bold-i-normal--0-0-0-0-p-0-iso8859-1
arialbd.ttf -monotype-Arial-bold-r-normal--0-0-0-0-p-0-iso10646-1
arialbd.ttf -monotype-Arial-bold-r-normal--0-0-0-0-p-0-iso8859-2
arialbd.ttf -monotype-Arial-bold-r-normal--0-0-0-0-p-0-iso8859-1
arial.ttf -monotype-Arial-medium-r-normal--0-0-0-0-p-0-iso10646-1
arial.ttf -monotype-Arial-medium-r-normal--0-0-0-0-p-0-iso8859-2
arial.ttf -monotype-Arial-medium-r-normal--0-0-0-0-p-0-iso8859-1
```

Kapcsolódó címek

A Magyar fontok nevű multimédiás termék a Kiskapu virtuális könyvesboltban
➲ http://www.kiskapu.hu/kiskapu/search.phtml?detailed=501201

Microsoft TrueType fontok
➲ http://www.microsoft.com/truetype/fontpack/win.html

Freeware Connection: Free Fonts Sites
➲ http://www.freewareconnection.com/fonts.html

Cabextract weblapja ➲ http://www.kyz.uklinux.net/cabextract.php3

sorokra vigyázzunk, ezekre szükségünk lesz. Az új X nagy előnye, hogy nemcsak laponként, hanem egyben is tudja kezelni a 8859-es kódolású Unicode készleteket, az ilyen készletekhez tehát érdemes létrehozunk még egy bejegyzést, `iso10646-1` végződéssel.
A rendszernek meg kell adni, hogy az új könyvtárban is keresse a betűtípusokat. Ezt az X beállítási állományában (`/etc/X11/XF86Config-4`) kell megtenni. Keressük meg a *Files* részt, ez alatt van egy vagy több `FontPath` kezdetű sor. Itt adjuk meg a saját új könyvtárunkat:

```
FontPath "/usr/local/myfonts"
```

Az X újraindítása után már vígan használhatjuk is új betűtípusainkat. Megnézhetjük az újoncokat, például az `xfontsel` programmal. Arra vigyázzunk, hogy ha a rendszer nem találja induláskor a *fixed* nevű betűtípust, nem képes elindulni.

XFree86 3.3.6 alatt

Kicsit nehezebb a dolgunk a régebbi változatnál. Ez alatt betűkiszolgálót kell telepítenünk, hogy TrueType betűket használhassunk. Ezek közül az egyik legjobb és legismertebb az *xfs*, mely képes a magyar betűket is kezelni (betűkiszolgálóra az újabb rendszereken nincs is szükség, csak például hálózatos megosztás esetén). A betűkiszolgáló használatakor az `XF86Config` fájlban egyetlen bejegyzés szerepel, mely az alábbihoz hasonló:

```
FontPath "unix/:-1"
```

A számunkra fontos sor ebben az esetben a kiszolgáló beállítási állományában található (xfs használata esetén: `/etc/X11/xfs/config`), itt kell beírnunk a megfelelő elérési utat, a `catalogue=` után szépen felsorolva és vesszővel elválasztva, ha több könyvtárban helyezkednek el fájljaink. A módosítás után a betűkiszolgálóval is tudatni kell a változásokat, ezt általában az alábbi parancs meg is teszi (ha mégsem, használhatjuk a `restart` kapcsolót):

```
/etc/init.d/xfs-xtt reload
```

Megjegyzem, hogy a Debian rendszereken előfordulhat, hogy az XF86Config-4 fájlban a *Section „Module”* részbe be kell még szúrni egy `Load "freetype"` sort, hogy a TrueType fontokat kezelő modul is betöltődjön.
Arra az apróságra figyeljünk oda, hogy a betűkiszolgáló még az X előtt elinduljon, különben nem fog működni a grafikus felületünk.
Ha átverekedtük magunkat a beállítás nehezebb szakaszain, és ha ügyesek voltunk, használatba is vehetjük programjainkkal TrueType betűkészleteinket.

Finomhangolás

Működik a grafikus felületünk, használja is a frissen beállított betűtípusokat, de külalakra nem az igazi…
A „telepítés” során a ttmkfdir programmal létrehoztuk a fonts.dir fájlt, most ismételten ezt a fájlt fogjuk felhasználni, hogy betűcsaládjaink méretezhetőségét ki tudjuk használni, csupán egy hivatkozást kell létrehoznunk fonts.scale névvel (`ln -s fonts.dir fonts.scale`) Vizsgáljuk csak meg közelebbről ezt a fonts.dir fájlt! Mit tudunk eddig a betűtípusokról? Tudjuk, hogy jó esetben Unicode-megfelelőek (vagy legalábbis WGL4 karakterkészletűek) és tudjuk a betűcsaládok nevét. Lássuk milyen adatokat gyűjtött össze a ttmkfdir program (lásd *1. listát*)!
Szép, szép, de mire való ez a sok butaság? Számunkra általában elegendő az iso8859-1 és iso8859-2 végű bejegyzés, viszont hiányzik az iso10646-1 végű bejegyzés.
Akinek ez a három tábla nem elegendő, pontosan tudja, mire van szüksége. A gyorsabb feldolgozáshoz célszerű csökkenteni a fonts.dir fájl méretét, csak arra ügyeljünk, hogy a fájl mindig a bejegyzések számával kezdődik, sose felejtsük el módosítás után a számot is megváltoztatni (2. lista). A feladat nagy részét elvégzi a következő pár parancs:

```
$ sed s/koi8-r/iso10646-1/ fonts.dir | grep -E
➥"iso8859-[12]$|iso10646-1$" >fonts.dir.uj
$ wc -l fonts.dir.uj | tr -d " " | cut -f 1
➥--delimiter="f" > szam
$ cat fonts.dir.uj >> szam
$ mv szam fonts.dir
$ rm fonts.dir.uj
```

Ha mindez megvan, indítsuk újra az X-et és készen is vagyunk. Most már elkezdhetjük átállítgatni a különböző programjainkat az új betűkészletek használatára…
Megérte, igaz?

Ficska Sándor
(fricska.sandor@linuxvilag.hu) rendszergazdaként dolgozik. Szabadidejének egy részét a természetben pingvinekkel, hüllőkkel és nőkkel tölti :) fennmaradó részét pedig az LME aktív tagjaként.

A SOAP

A SOAP olyasmi, aminek akkor is nagy hasznát vehetjük, ha valójában nem is érdeklődünk a háromrétegű webes alkalmazások iránt.

A két előző Kovácsműhelyben egy egyszerű háromrétegű webalkalmazást mutattam be adatbázis, webkiszolgáló és a mod_perl Mason sablonrendszer használatával. Megvizsgáltuk a háromrétegű alkalmazások előnyeit és hátrányait, és összehasonlítottuk a kétrétegű alkalmazásokkal.
Azonban – ahogy arra a múlt hónapban is rámutattam – a mi kis háromrétegű alkalmazásunk még nincsen teljesen készen, ezért nem lehet megfelelően bemutatni a működését. Emiatt a Perl köztes objektumrétege ugyanazon a számítógépen található, mint a mod_perlre épülő Mason sablonrendszer segítségével készített HTML-elemek. Annak ellenére kétrétegű alkalmazásnak tekinthetjük, hogy a rétegeket objektumközpontú elvonatkoztatási réteg választja el egymástól.
Ahhoz, hogy külön számítógépre tegyük a Mason-elemeket és a Perl-objektumokat, meg kell oldanunk a hálózaton keresztüli objektumhívást. A következő sornak például attól függetlenül működnie kell Perlben, hogy a `$object` ugyanazon az Apache kiszolgálón van-e, vagy valahol máshol az Interneten:

```
$object->method($arg1, $arg2);
```

Az elosztott objektum módszer és a távoli eljáráshívás jó néhány éve elérhető a különböző felületeken. A legtöbb esetben ez a módszer egy független nyelvre vagy felületre korlátozódik. A DCOM modell (Distributed Component Object Model) lehetővé teszi az objektumok kapcsolattartását, de csak Windows alatt. A Java távoli eljáráshívása (Remote Method Invocation – RMI) csak Java-objektumokkal hajlandó együttműködni. A CORBA kivételesen lehetővé teszi a különböző nyelvek és felületek közötti kapcsolattartást, de nagyon összetett nyelv, és jelenleg a legtöbb programozó nem ismeri eléggé.
Ezekre az egyedi és összetett protokollokra válaszul az internetes közösség létrehozta saját egyszerű objektumelérési protokollját, a SOAP-ot (Simple Object Access Protocol). Segítségével pofonegyszerű elosztott alkalmazásokat létrehozni. A SOAP két legnagyobb támogatója a weblog hírlevélből ismert *Dave Winer* és a Microsoft. Bár e cég nem arról híres, hogy támogatja a nyílt szabványokat és a felületfüggetlen protokollokat. A linuxos közösség véleménye szerint a Microsoft azért támogatja nyilvánosan a SOAP-ot, mert a .Net programjának is ez az egyik alapköve.

A SOAP története és alapgondolata

A SOAP alapgondolata az, hogy az Interneten található bármilyen két számítógép a HTTP protokoll használatával képes kapcsolatba lépni egymással. Jelenleg a SOAP majdnem mindegyik magas szintű protokollon továbbítható, mint például az SMTP és a POP3, de a HTTP a leggyakoribb. Képes adatot továbbítani az XML használatával (ez a nyelv lehetővé teszi, hogy saját tagokat és dokumentumszabványokat hozzunk létre). A kiszolgáló a beérkező XML parancsokat lefordítja objektum eljáráshívásokra, majd az objektum válaszát befordítja egy XML dokumentumba, és HTTP válaszként továbbítja. Mivel a HTTP és az XML is olyan nyílt szabvány, amit a World Wide Web Consortium hozott létre, ezért minden gond nélkül megvalósítható és alkalmazható minden felületen.
A SOAP ősét XML-RPC-ként ismerjük. Ez egyszerűen lehetővé tette a távoli eljáráshívást (Remote Procedure Call) XML formátumú adatokkal HTTP protokollon keresztül. Az XML-RPC azonban nem kezeli a fejlett adatszerkezeteket, ezért a W3C létrehozta a SOAP-ot. Számos nyelv és felület továbbra is támogatja az XML-RPC-t, ennek következtében az néhány esetben némi fejfájást is okozhat. Röviden összefoglalva a SOAP annak köszönheti a kiemelt figyelmet, hogy a programkönyvtárak, a függvények használata és a hibakeresése fejlettebb, mint XML-RPC-é. A SOAP szélesebb körben alkalmazható, mint az XML-RPC, ez azt is jelenti, hogy szabadon választható a felület, a nyelv, vagy akár a protokoll.
A SOAP, ahogy a nevéből is sejthető, objektumokkal, és nem egyszerű eljáráshívásokkal dolgozik. A SOAP ügyfél a kiszolgálón található objektum bármelyik tagfüggvényét meg tudja hívni. A tagfüggvényt az XML dokumentum törzsében adhatjuk meg, az objektumot magát pedig a HTTP fejléc „SOAPAction” sorában. Természetesen meg kell határozni a számítógép nevét és a kaput is, amelyre a SOAP-kérést irányítjuk.
A kiszolgálót (a gép nevét és a kapu címét), ahová a kérést küldjük, SOAP proxynak hívjuk. Az elnevezés érthető, ha arra gondolunk, hogy a HTTP kiszolgáló egyszerűen csak továbbítja az elvégzendő parancsokat, ő maga nem végez feladatot. Ne tévesszük össze a SOAP proxyt és a HTTP proxyt. A HTTP proxy továbbítja a HTTP ügyfél kéréseit a HTTP kiszolgálónak, és gyakran biztonsági ellenőrzéseket hajt végre, ezenkívül gyorstárként is működik. Ezzel ellentétben a SOAP proxy az ügyfél és az objektum között zajló üzeneteket továbbítja.
A SOAP kiszolgálón található objektumra gyakran *végpont* névvel is hivatkozunk, ezt a fejléc „SOAPAction” sorában kell megadnunk. A végpont neve tulajdonképpen bármilyen szöveges változó lehet, beleértve a hierarchikus elválasztó jeleket is (mint például :: és /). A végpont írásmódja általában követi azt a programnyelvet, amiben a SOAP proxy is készült. Perlben a végpont lehet modjuk Foo/Bar, vagy hasonló, ami a Foo/Bar.pm-ben található Foo::Bar objektumra utal.

A SOAP belső felépítése

Most nézzünk egy egyszerű SOAP „beszélgetést”. Példánkban a SOAP a HTTP protokollon ül, mivel ez a leggyakoribb. Más protokollok használatánál némi eltérést tapasztalhatunk.
A HTTP nem állapotfüggő, ami azt jelenti, hogy két számítógép között létrejött kapcsolat egyetlen kérésből (az ügyféltől a kiszolgáló felé) és egyetlen válaszból (a kiszolgálótól vissza az ügyfélhez) áll. A kérés és a válasz is két részre bontható: a fejlécre és a törzsre. Természetesen az ügyfél és a kiszolgáló tetszőleges, más fejléceket is hozzátehet, ajtót nyitva ezzel számos különleges kapcsolattartó protokoll előtt.
A SOAP kérés és válasz törzse XML formátumú lesz. Ha sohasem dolgoztál még XML-lel, akkor sem kell megijedni. Bár ez egy nagy és furmányos témakör, ahhoz, hogy a SOAP-ot használd, nem kell túl

1. lista Caps.pm, a Perl modul

A Caps.pm szolgáltatja a SOAP végpontot.

```
package Text::Caps;

use strict;
use diagnostics;
#A végleges programban iktassuk ki.

# Egyetlen változót várunk. A kapott változót
# nagybetűssé alakítjuk
# a beépített uc függvénnyel, majd visszaadjuk.

sub capitalize
{
   my $self = shift;
   my $word = shift;

   return uc ($word);
}

# A capitalize_array egy listát vár.
# Az eljárás a kapott listához igazodó
# listát ad vissza, az elemeket nagybetűssé
# alakítva.

sub capitalize_array
{
   my $self = shift;
   my @words = @_;

   return [map {uc $_} @words];

}
```

sokat tudnod az XML-ről. Minden SOAP üzenet – akár kérés, akár válasz – tartalmaz egy kiegészítő SOAP fejlécet és egy kötelező SOAP törzset a SOAP borítékba csomagolva. A boríték azonosítja a SOAP csomag tartalmát, és meghatározza a névmezőt, amit a későbbiekben az üzenet további részéhez használhatunk fel. A fejlécek leírják a törzsben található adatokat, a törzs pedig tartalmazza a tagfüggvényhívást, vagy annak eredményét.

Hogy a SOAP segítségével egy távoli objektumot használjunk, először egy http kapcsolatot kell létrehoznunk a kívánt URL felé, és a „SOAPAction” sorban meg kell adjuk a használni kívánt objektum pontos nevét. Egy XML dokumentumot küldünk, benne a SOAP borítékkal, mely tartalmazza a SOAP fejlécet és a törzset. A törzsben megadjuk a meghívni kívánt tagfüggvényt és az összes további értéket. Az ügyfelet fel kell készíteni a SOAP kiszolgáló által visszaküldött adatszerkezetek kiértékelésére és további felhasználására.

A SOAP kiszolgáló hasonló műveleteket hajt végre, fogadja a SOAP kéréseket, elemzi azok tartalmát, és meghívja a megfelelő tagfüggvényt az átadott értékekkel együtt. Ezután visszaküldi a szükséges értékeket tartalmazó választ az ügyfélnek.

Most, hogy megismertük a SOAP szókincsét, a többségét el is felejthetjük. A SOAP elvonatkoztatási rétegekkel történő megvalósításánál lehetőségünk nyílik arra, hogy figyelmen kívül hagyjuk a HTTP kapcsolattartást és az XML kérés és válasz részleteit. Ha a programunk a SOAP segítségével kéri le a távoli objektumfüggvényt, a kérések és válaszok csomagolása a SOAP feladata.

2. lista Az egyszerű önálló SOAP kiszolgáló

Ez a program a 8080-as kapun várakozik a SOAP kérések fogadására, és átadja őket a megfelelő objektumnak.

```
#!/usr/bin/perl -w

use strict;
use diagnostics;
# A végleges programban iktassuk ki

use SOAP::Transport::HTTP;
# Az objektum fogadása a kiszolgáló számára

my $SERVER_PORT = 8080;
my $SERVER_NAME = 'localhost';

# A SOAP kiszolgáló objektum létrehozása
my $soap_server = SOAP::Transport::HTTP::Daemon
    -> new (LocalAddr => $SERVER_NAME,
           ➥LocalPort =>$SERVER_PORT)

# Mi az objektum alapkönyvtára?
# (Ne feledjük, az alapértelmezett Perl @INC
# útvonal le van tiltva.)

# Ne használjuk a /tmp útvonalat egy valódi
# kiszolgálónál!
    -> dispatch_to('/tmp/');

# Kiírjuk, melyik kapun várjuk a SOAP kéréseket.
print "SOAP server is waiting on port
$SERVER_PORT...\n";

# Most kezeljük a beérkező SOAP eljáráshívást,
# és visszaküldjük a megfelelő SOAP választ.
$soap_server->handle();
```

A kiszolgálóoldali objektum

Írtam néhány példaprogramot Perlben a *Paul Kulchenko* alkotta kitűnő SOAP::Lite modul segítségével. Ez adhat néhány ötletet a SOAP kiszolgálók és ügyfelek megírásához, valamint a webalkalmazásainkba építésük módozataihoz. Nevének dacára, a SOAP::Lite rengeteg szolgáltatást kínál számunkra, és remek eszköz lehet a Perl programok SOAP-pal történő bővítéséhez. Hasonló SOAP függvények és objektumok a jelentősebb programnyelveken is megtalálhatók, ne gondoljuk, hogy a SOAP csak a Perllel használható.

Mivel a SOAP az objektum számára proxyként működik, először létre kell hoznunk egy objektumot, mely elérhető a hálózaton. Az *1. lista* tartalmazza az egyszerű Text::Caps Perl modult, ez a modul két eléggé hasznavehetetlen eljárást tartalmaz:

- *capitalize* – a kapott szöveget nagybetűsre alakítva küldi vissza.
- *capitalize_array* – ez ugyanazt csinálja, mint a capitalize, de a kapott tömb összes elemével.

Jegyezzük meg, míg a SOAP minden kifejezést objektumalapú megnevezéssel ír le, ez a mintamodul inkább a hagyományos Perl írásmódot használja. Szóval, ha azt emlegetem, hogy a Text::Caps objektum capitalize eljárása, akkor valójában a Text::Caps::Capitalize eljárás meghívására gondolok.

3. lista SOAP-kérés

```
POST http://localhost:8080/
Content-Length: 509
Content-Type: text/xml
SOAPAction: "Text/Caps#capitalize"

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:xsi=
➥"http://www.w3.org/1999/XMLSchema-instance"
xmlns:SOAP-ENC=
➥"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema"
SOAP-ENV:encodingStyle=
➥"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV=
➥"http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<namesp3:capitalize xmlns:namesp3="Text/Caps">
<c-gensym19 xsi:type="xsd:string">abc</
➥c-gensym19>
</namesp3:capitalize>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Egy önálló SOAP kiszolgáló

A SOAP általában a HTTP-t használja, amely viszont a TCP/IP protokoll tetején „utazik”. Ez azt jelenti, hogy a Perl beépített, TCP/IP foglalatokat (socket) kezelő részének használatával akár egy egyszerű SOAP kiszolgálót is létrehozhatunk. Így a SOAP::Lite elvégzi helyettünk a piszkos munka nagy részét, nem nekünk kell létrehozni a foglalatot, vagy várakozni rá. Inkább, készítsünk egy SOAP::Transport::HTTP::Daemon-típusú objektumot, mely tudja, hogyan viselkedjen megfelelő típusú SOAP kiszolgálóként. Az egyszerű kiszolgáló forráskódja a *2. listában* látható.
A forráskód viszonylag egyszerű, ennek ellenére még a gyakorlott Perl programozók számára is furcsának tűnhet. Talán mert a SOAP::Lite-hoz kapcsolódó objektumok a sikeres működést többnyire úgy jelzik, hogy saját magukat adják meg visszatérési értékként. Ez teszi lehetővé számunkra, hogy egynél több eljárást is lekérhessünk egyetlen hívásban. Szóval, azt mondhatjuk, hogy

```
$object->method1()->method2();
```

az alábbi hagyományos írásmód helyett

```
$object->method1();
$object->method2();
```

A SOAP::Lite-ban mindkét formátumot használhatjuk, de az első változat gyakrabban fordul elő a leírásban. Amikor a SOAP::Transport::HTTP::Daemon „new” létrehozóját (constructor) hívjuk meg, két értéket adunk át neki: azt a gépnevet és kapucímet, ahol a démonnak figyelnie kell.
A kiszolgáló létrehozása után meg kell adnunk, hogy hol találja meg a keresett objektumokat. Erre biztonsági okokból van szükség, bár elsőre ez nem túl érthető. Általában a Perl a modulokat a @INC-ben keresi a könyvtárnevek sorrendjében. Amikor használni kívánunk egy modult, a Perl a @NIC minden elemét egymás után addig vizsgálja, amíg meg nem találja a keresett modult. Ha a keresés eredménytelen, hibaüzenetet kapunk.

4. lista SOAP-válasz XML-ben

```
HTTP/1.1 200 OK
Date: Sun, 07 Jan 2001 20:31:35 GMT
Server: libwww-perl-daemon/1.21
Content-Length: 523
Content-Type: text/xml
Client-Date: Sun, 07 Jan 2001 20:31:35 GMT
Client-Peer: 127.0.0.1:8080
SOAPServer: SOAP::Lite/Perl/0.44

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelopei xmlns:xsi=
➥"http://www.w3.org/1999/XMLSchema-instance"
xmlns:SOAP-ENC=
"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema"
SOAP-ENV:encodingStyle=
➥"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV=
➥"http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<namesp1:capitalizeResponse xmlns:namesp1=
➥"Text/Caps">
<s-gensym5 xsi:type="xsd:string">ABC</
➥s-gensym5>
</namesp1:capitalizeResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Bár a SOAP-modulok az egész világon megtalálhatók, jobb, ha óvatosak vagyunk, mielőtt elérhetővé teszünk egy modult. Megtörténhet, hogy egyes modulok bizalmas adatokat továbbítanak, vagy adatokat módosítanak egy relációs adatbázisban. Ezért győződjünk meg róla, hogy csak a működéshez szükséges modulok érhetők el a SOAP számára, így a SOAP::Lite teljesen figyelmen kívül hagyja az @INC-et a SOAP-kérések érkezésekor. Csak azok a modulok szerepeljenek a dispatch_to() hívásban, vagy a dispatch_to() nevű könyvtárban, melyeket elérhetővé kívánunk tenni a SOAP számára. Tulajdonképpen a dispatch_to() helyettesíti az @INC változót a SOAP számára. Ha egy modul egy olyan könyvtárban található, ami nem szerepel a dispatch_to()-ban, a SOAP számára láthatatlan lesz. Ez nem ugyanaz, mintha az @INC értékét módosítanánk. Megjegyzem, miközben én a példákban /tmp könyvtárat használom, rossz ötlet ezt a könyvtárat használni egy éles fejlesztés során, vagy működő rendszernél. Ha a /usr/lib/perl könyvtárból külön könyvtárba szeretnénk elhelyezni a SOAP-hoz tartozó Perl-modulokat, határozottan javaslom, hogy a központi fájlrendszeren tartsuk azokat, például a /usr/lib/soaplite könyvtárban.

Kiszolgálónk kipróbálása

Most, hogy az önálló SOAP kiszolgálónk üzemel, ki kell próbálnunk hogy valóban működik-e. Ehhez először létre kell hoznunk egy SOAP-kérést, el kell küldeni a kiszolgálónak, és meg kell vizsgálni a visszaérkező XML kódolású üzenetet. Szerencsére a SOAP::Lite tartalmazza a SOAPsh.pl nevű segédprogramot. Ennek segítségével tudunk készíteni és küldeni SOAP-kéréseket, és interaktív módon az eredményeket azonnal meg is jeleníti a képernyőn. Már csak a SOAPsh.pl kedvéért is megéri letölteni a SOAP::Lite-ot, még akkor is, ha más SOAP függvényekkel fogunk majd dolgozni.
Ha a SOAP kiszolgálónkat ugyanazon a számítógépen futtatjuk, mint

5. lista A hibakeresés eredménye

```
POST http://localhost:8080/
Content-Length: 628
Content-Type: text/xml
SOAPAction: "Text/Caps#capitalize_array"

<?xml version="1.0"encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:xsi=
➥"http://www.w3.org/1999/XMLSchema-instance"
xmlns:SOAP-ENC=
➥"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema"
SOAP-ENV:encodingStyle=
➥"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV=
➥"http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<namesp4:capitalize_array
➥xmlns:namesp4="Text/Caps">
<c-gensym24 xsi:type="xsd:string">
reuven
</c-gensym24>
<c-gensym26 xsi:type="xsd:string">
shira
</c-gensym26>
<c-gensym28 xsi:type="xsd:string">
atara
</c-gensym28>
</namesp4:capitalize_array>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

HTTP/1.1 200 OK
Date: Sun, 07 Jan 2001 20:41:03 GMT
Server: libwww-perl-daemon/1.21
Content-Length: 738
Content-Type: text/xml
Client-Date: Sun, 07 Jan 2001 20:41:03 GMT
Client-Peer: 127.0.0.1:8080
SOAPServer: SOAP::Lite/Perl/0.44

<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:xsi="http://www.w3.org/1999/
XMLSchema-➥instance"
xmlns:SOAP-ENC=
➥"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema"
SOAP-ENV:encodingStyle=
➥"http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV=
➥"http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
<namesp2:capitalize_arrayResponse
➥xmlns:namesp2="Text/Caps">
<SOAP-ENV:Array xsi:type="SOAP-ENV:Array"
SOAP-ENC:arrayType="xsd:string[3]">
<s-gensym10 xsi:type="xsd:string">
REUVEN
</s-gensym10>
<s-gensym10 xsi:type="xsd:string">
SHIRA
</s-gensym10>
<s-gensym10 xsi:type="xsd:string">
ATARA
</s-gensym10>
</SOAP-ENV:Array>
</namesp2:capitalize_arrayResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

a SOAPsh.pl-t, és a 8080-as kaput használjuk, a következő kérést alkalmazzuk:

```
perl SOAPsh.pl http://localhost:8080/ Text/Caps
```

Megjegyzem, az SOAPsh.pl első paramétere a SOAP kiszolgáló címe, a második pedig a használni kívánt objektum. Sok nehezen lenyomozható hibától mentjük meg magunkat, ha a második paramétert az URL-ekben megszokott módon, tehát perjellel tagoljuk. A Text::Caps helyett inkább a Text/Caps formát használjuk. Ha a SOAPsh.pl sikeres, a következőt láthatjuk:

```
Usage: method[(parameters)]
>
```

A > jel azt jelenti, hogy kapcsolatba léptünk az objektummal, és bármelyik tagfüggvényét meghívhatjuk. Egy szó nagybetűssé alakításához egyszerűen csak be kell gépelni:

```
> capitalize(´abc´)
```

6. lista soap-client.pl

```
#!/usr/bin/perl -w

use strict;
use diagnostics;  # A végleges programban
                  iktassuk ki
use SOAP::Lite;


# Adjuk meg a kapott változót az átalakításhoz
my $result = SOAP::Lite -> uri('Text/Caps') ->
    proxy('http://localhost:8080') ->
    capitalize($ARGV[0]) -> result();

# Írassuk ki az eredményt
print "Result = '$result'\n";
```

Mivel a SOAP kiszolgáló és az ügyfél azonos számítógépen van, ezért a válasz szinte azonnal megérkezik. A SOAPsh.pl ezt írja ki:

```
--- SOAP RESULT ---
$VAR1 = ´ABC´;
```

Hú, ez nem semmi! Hálózaton keresztül kértem objektumeljárást. Nem is volt olyan vészes, igaz?
A SOAP könnyűnek látszik, ha csak egyszerű változókat küldözgetünk oda-vissza. De küldhetünk különböző típusú adatokat is. Például próbáljuk ki a `capitalize_array` parancsot egy tömb küldésével:

```
> capitalize_array(´abc´, ´def´, ´GHi´)
```

A visszaérkező értékek az alábbiak lesznek:

```
--- SOAP RESULT ---
$VAR1 = bless( [
         ´ABC´,
         ´DEF´,
         ´GHI´
               ], ´Array´ );
```

Az érkezett adatok egy kicsit viccesen néznek ki, mivel abban a formátumban látjuk, ahogy azt a SOAP::Lite küldi és fogadja. Hamarosan megláthatjuk, hogy a programunk hogyan kerüli meg ezt a „szakszerűtlen” formátumot, és hogyan cserélgeti észrevétlenül az összetett adatszerkezeteket az Interneten.

A SOAP vizsgálata

Ahogy azt az imént is láthattuk, a SOAP-pal lehet anélkül is dolgozni, hogy mélységeiben ismernénk az XML kódolás rejtett titkait. Mégis, a hibakeresés közben gyakran alakul úgy, hogy ismernünk kell az XML részleteit, legalább annyira, amennyire a HTTP fejléceket.
A SOAP::Lite objektumok támogatják az on_debug( ) eljárást, mely egy eljárás címét várja. Ezt az eljárást azután minden tranzakció után meghívja, lehetőséget adva ezzel számunkra, hogy az eseményekről naplót készítsünk, vagy folyamatosan megjelenítsük azokat a képernyőn. Egy egyszerű példa erre a következő:

```
on_debug(sub{print STDERR @_})
```

Megkérjük a SOAP::Lite-ot, hogy küldjön mindenről másolatot az STDERR-re. Így megtudhatjuk, hogy tulajdonképpen mi is történik a színfalak mögött. Az eljárás futtatása után a SOAPsh.pl emlékeztet, hogy egy helyi eljárást hívtunk meg, és nem a SOAP-on keresztül hívtuk azt meg:

```
--- METHOD RESULT ---
SOAP::Lite=HASH(0x82e1174)
```

Most láthatjuk, ahogy a capitalize(abc) kérés SOAP-kéréssé alakul át (lásd a *3. listát*).
Amint azt láthatjuk, a kérés két részből (a fejlécből és a törzsből) áll, mint minden HTTP-kérés. És mint minden átlagos HTTP-kérésnél, meg kell jelölnünk a tevékenységet („POST”), a teljes címet („URL”), a tartalom hosszát (Content-Length), és a tartalomtípust (Content-Type, ez mindig text/xml).
Most jön a lényeg: az utolsó sor a fejlécben a SOAPAction, amely megnevezi az objektumot és a hívás módját. A SOAPAction működését úgy alakították ki, hogy a vállalati tűzfalak kiszűrhessék a veszélyes objektumokat és tagfüggvényeket. Ennek ellenére jelenleg

7. lista cgi-soap.pl

```
#!/usr/bin/perl -w

use strict;
use diagnostics;     # A végleges programban
                       iktassuk ki

use SOAP::Transport::HTTP;

SOAP::Transport::HTTP::CGI
     -> dispatch_to('/tmp')
     -> handle ;
```

viszonylag nehéz SOAPAction támogatást találni. Akárhogy is, az objektum és a tagfüggvények adatai is be vannak ágyazva az XML törzsbe, ezért a fejlécet gyakran felesleges kielemezni.
Maga az XML egy XML bevezetővel kezdődik, majd egy SOAP borítékkal. A borítékon belül egy fejléc (nem kötelező rész) és egy törzs (kötelező rész) szerepel. A törzs nevezi meg az objektumot, valamint a használandó tagfüggvényt, az összes átadandó értékkel. Ezután a kapott anyagot a kiszolgáló átalakítja az operációs rendszer által értelmezhető formába, majd továbbítja azt a célobjektum felé. Az objektum által visszaadott értéket a kiszolgáló XML formátumra alakítja (*4. lista*).
A válasz, akárcsak a kérés, a HTTP-t használja, és a HTTP fejléc tartalmaz olyan metaadatokat is, mint a kiszolgáló típusa, a dátum, a tartalom hossza, a típus (text/xml), és végül a futtató SOAP kiszolgáló típusa. A boríték ebben az esetben (akárcsak a kérésben) nem tartalmaz fejlécet. Így a törzs tartalmazza a visszatérő értéket (xsd:string típusú). Míg a kérés a namesp3:capitalize névteret használja, a válasz a namesp1:capitalizeResponse-t. Ez egy szabvány a SOAP-ban; az XML névtereket használja, hogy azonosítsa, hogy kérdésről, vagy válaszról van szó, illetve, hogy milyen kéréshez tartozik a válasz.
Az *5. listán* látható (magyarázat nélkül) a capitalize_array(reuven, shira, atara) hívás hibakeresési kimenete.

Egy SOAP ügyfél

SOAPsh.pl jól példázza az interaktív kérések működését, de a SOAP sokkal hasznosabb, ha a saját programunk készíti el és dolgozza fel a kéréseket. A *6. listán* látható, ahogy a SOAP ügyfél a 8080-as kapun csatlakozik a kiszolgálóhoz. Látható, hogy az URI-ban még egyszer az objektum neve, a proxyban pedig a SOAP kiszolgáló neve szerepel.
Éppen ez a megoldás oly különleges a SOAP::Lite-ban, ahogy lehetővé teszi a hálózaton keresztül az objektumok tagfüggvényeinek lekérdezését. Az uri, a proxy és a result nyilvánvalóan léteznek a SOAP::Lite objektumban. De a capitalize eljárás csak a távoli Text/Caps objektumban létezik. A SOAP::Lite elég okos ahhoz, hogy észrevegye a különbséget, és képes kiválasztani azokat az eljárásokat is, amelyek helyben nem oldhatók meg.

A CGI-alapú SOAP kiszolgáló

A mi kis önálló kiszolgálónkat egyszerűként emlegettük, mert az is. Vajon mi történik akkor, ha naponta több millió kérés fut be? Ezt a mi egyszerű kiszolgálónk nem bírja, és a felhasználók kérelmeit nem teljesíti.
Célszerű megoldás a HTTP-forgalom kezelésére készített program, név szerint az Apache használata. Az Apache segítségével könnyedén kezelni tudjuk a kisebb és nagyobb forgalmat is. Az Apache haszná-

lata esetén a programunk nem kell foglalkozzon ezekkel a kérdésekkel, összpontosíthatunk a SOAP csomagok fogadásának részleteire és a Perl-modulok közötti kapcsolatokra.
A CGI-alapú proxy készítése nem sokban különbözik az önállóan futtatható program készítésétől. A legfontosabb dolog, amit fejben kell tartanunk az, hogy ne használjuk a CGI.pm-et, vagy egyéb ehhez hasonló CGI-s modult! A mi esetünkben a CGI-feladatokat a SOAP proxy valósítja meg, és a CGI protokoll használatával tudjuk az XML-kódolású üzeneteket oda-vissza küldözgetni.

Egyéb nyalánkságok

A SOAP::Lite rengeteg hasznos segédprogramot tartalmaz, ezért elég nehéz megállni, hogy az ember fel ne sorolja mindet. Például azoknak, akik a mod_perlt használják a CGI helyett, jó hír, hogy a SOAP::Lite beépített mod_perl-, valamint CGI-támogatással is bír. Saját programunkban kihasználhatjuk az „autodispatch” szolgáltatást, melyet az előbbiekben is láthattunk működés közben, ha egy helyileg ismeretlen tagfüggvényt hívunk meg, azt a rendszer továbbítja a távoli objektumnak.
A SOAP::Lite képes kezelni a legtöbb, SOAP által is támogatott adatszerkezetet, beleértve az objektumokat is. Például meghívhatjuk egy távoli objektum new( ) függvényét, majd a visszakapott objektummal további feladatokat hajthatunk végre. Ez a szolgáltatás már évek óta létezik számos különleges felületen, de az a tény, hogy a SOAP felületfüggetlen, valóban bámulatra méltó.
Végül, a SOAP növekedésére jellemző, hogy nem is olyan régen még csak a HTTP protokollt támogatta, napjainkban már olyan gyakori protokollal is együttműködik, mint például a POP3 és az SMTP, és a lehetőségek tárháza folyamatosan bővül.

A SOAP és a háromrétegű alkalmazások

Most, hogy láttuk, hogyan működik a SOAP egyszerű körülmények között, megfigyeljük, hogyan használható összetettebb környezetben. Tegyük fel, hogy szeretnék építeni egy hatalmas weboldalt egy relációs adatbázissal. A legtöbb esetben – mint ahogy a Kovácsműhely rendszeres olvasói is tudják – a feladatot a mod_perl és a HTML::Mason párossal szoktam megvalósítani.
Mivel a kiszolgálóoldali Java-piac is fellendült, néhány vagy az összes háttérművelet megoldható JavaBeanekkel is. Továbbá az Enterprise JavaBeans (EJB) egyre inkább fejlődő módszer a tranzakciókat használó elosztott alkalmazások számára, éppen ezért mindig is előnyben részesítem a javás megoldásokat.
A SOAP-pal, most kedvem szerint, szabadon keverhetem a nyelveket és a felületeket. Ha létrehozok egy SOAP kiszolgálót, amely hozzáfér a megfelelő Java-objektumokhoz, nincs semmi oka, hogy a Mason-elemek ne tartsanak kapcsolatot a középső Java-réteggel. Néhány esetben ez még jobb eredményekkel is járhat, mintha csak egyetlen nyelvet használnánk. Mivel a Perl nem támogatja a hálózati tranzakciókat, ezért a kezdetekben Perlben készített programot később majd az EJB segítségével szeretnénk továbbfejleszteni.
A SOAP használatával mindez lehetséges, sőt, egyenesen kívánatos.

Forrásanyagok

Számos weboldal és írás foglalkozik a SOAP-pal, mégsem láttam túl sok példaprogramot a használatára. Remek kiindulási pont *Dave Winer* oldala
➲ http://soap.weblogs.com/
Itt találhatunk magyarázatokat SOAP-megvalósításokra, és a SOAP hibák felderítésére a weblog segítségével.

A World Wide Web Consortium által közzétett SOAP szerkezet ➲ http://www.w3.org/TR/SOAP/
A SOAP::Lite modul letölthető az alábbi címről

➲ http://www.soaplite.com/

Paul Kulchenko, a SOAP::Lite szerzője, keményen dolgozik a modul fejlesztésén, és megfizethetetlen segítséget nyújtott a hibakeresésben e cikk megszületése során.

Végezetül, a webes alkalmazáskiszolgálók már elkezdték támogatni a SOAP-ot. Nem csupán azt teszik lehetővé, hogy más gépeken lévő, különböző nyelveken írt objektumok kapcsolatot tartsanak egy kiszolgálóval, de megnyitják a teret olyan internetalapú szolgáltatások felé, melyek nincsenek szükségszerűen hozzákötve a Webhez. Lehet, hogy hamarosan némelyik webes újság SOAP-alapú hírkezelő rendszert biztosít, amin keresztül egyetlen kéréssel lehívhatjuk az össze bennünket érdeklő hírt. Egy ilyen szolgáltatásra alapozva a felhasználó kirakhat az asztalára egy (nem webes) programot, mely folyamatosan a legizgalmasabb híreket mutatja.

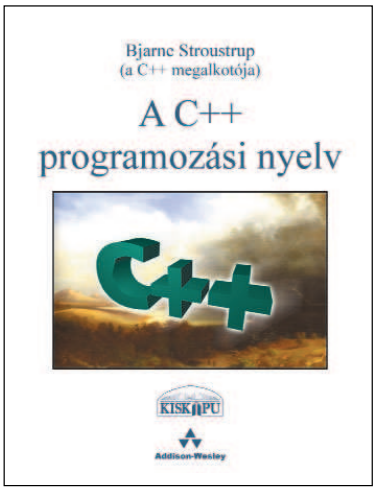
Zárszó

A SOAP az új típusú elosztott internetes alkalmazások előfutára, jelenleg az egyetlen, amely képes távoli eljáráshívásra operációs rendszerek és programnyelvek között. Nem kell tovább vesződni a „nehéz megérteni”, vagy a „nehéz meghívni” típusú eljárásokkal, és akár egy délutáni oktatáson el lehet készíteni egy egyszerű elosztott alkalmazást. Hogy mindez mit jelent, az Internet és a Web jövőjére nézve? Jó kérdés, de már készülnek azok az asztali alkalmazások, amelyek grafikus felülete SOAP-kéréseket továbbít a központi kiszolgálók felé. Attól függetlenül, hogy mit hozhat a jövő, az a tény, hogy a Perl és a többi szabad nyelv használja a SOAP-ot, azt jelenti, hogy hamarosan könnyebb lesz a kapcsolattartás, mint valaha. Vajon ki ne akarná az egész Internetet? (Ide nekem az oroszlánt…)

Reuven M. Lerner
(reuven@lerner.co.il) egy izraeli web- és internet-tanácsadó cég tulajdonosa és vezetője. A Kovácsműhely rovat honlapja a ➲ http://www.lerner.co.il/atf/ címen érhető el.

Közkívánatra

Április elején jelent meg Bjarne Stroustrup könyvének magyar fordítása A C++ programozási nyelv címmel.



Szerző: Bjarne Stroustrup
Terjedelem: két kötet, 1340 oldal
Ára: 9800 forint
Kiadó: Kiskapu (Addison-Wesley)
E-mail: kiskapu@kiskapu.hu
Honlap: ➲ www.kiskapu.hu
ISBN: 963 9301 19 1

A C++ nyelv közkedveltségének megfelelően számos ilyen tárgyú könyv jelent meg eddig is, mégis, ez a könyv mind közül kitűnik, hiszen a nyelv szerzőjének írásáról van szó. A német, spanyol, japán, orosz, francia, kínai, svéd, finn, lengyel, portugál, olasz és görög kiadás után most végre sort kerítettek eme alapvető mű magyar kiadására is.

Bjarne Stroustrup azért találta ki az osztályokkal bővített C nyelvet, hogy neki és barátainak könnyebb legyen moduláris felépítésű, eseményvezérelt feladatokat megoldani, amelyekre a Simula67 eszményi lett volna a hatékonysági gondjaitól eltekintve. A nyelv megjelenése óta (1980) rengeteget fejlődött. Bjarne később hatékonyabb nyelvet kívánt megvalósítani, melynek a C++ nevet adta. Az új nyelv alapjául a C-t választotta, és a Simula67-ből vett osztályfogalommal bővítette.

A kezdeti állapotokhoz képest a nyelv sokat fejlődött, új elemekkel bővült (többszörös öröklődéssel, statikus és állandó tagfüggvényekkel, a „protected” hozzáférési kategóriával, sablonokkal, kivételkezeléssel, futási idejű típusazonosítással), valamint a meglévő elemeken is csiszoltak, pontosították azokat (például a túlterhelés feloldási szabályait, a memóriakezelést, a C szabályainak jobb követését).

A nyelvet 1983-ig csak az AT&T Bell Labs kutatóintézetben használták, majd C++ néven vált ismertté. Azóta több független változata révén felületek és alkalmazási területek sokaságán terjedt el.

A nyelv fejlődésének fontos állomása volt a szabványosítás: az amerikai (ANSI) szabványosítási folyamat 1998-ra vezetett a nyelv szabványának nemzetközi szintű (ISO) elfogadásához. A szabvány részévé vált – a standard könyvtár keretében – a korábban Standard Template Library néven ismert sablongyűjtemény is.

A nyelv fejlődését tükrözik a könyv változásai is: a második kiadás adatainak körülbelül egyharmada származik az első kiadásból, és a harmadik kiadás még nagyobb arányú átírás gyümölcse. A harmadik kiadás megjelenését követően kezdődött meg a magyar nyelvű fordítás elkészítése, de az Addison Wesley-Kiadó jóvoltából a tényleges kiadás az időközben napvilágot látott „special edition”-változaton alapul.

A könyv szerkezete

A könyv a nyelv ismertetésén túlmutató témákat is tárgyal, többek között általános programozási mintákat is.

A C++ nyelvet röviden bemutató rész (*Kirándulás a C++-ban*) a nyelv által támogatott programozási minták (procedurális, moduláris, típusalapú, objektumközpontú, generikus) szerint az egyszerűbbtől a bonyolultabb felé halad. Ezt a standard könyvtárat bemutató fejezet követi (még a bevezető részen belül) *Kirándulás a standard könyvtárba* címmel.

A bevezető rész után a könyv további főbb részei:

- *Alapok* (alaptípusok, kifejezések, utasítások, függvények stb.)
- *Absztrakciós módszerek* (osztályok, operátorok, származtatás, sablonok stb.)
- *Standard könyvtár* (a könyvtár szerkezete, szabványos tárolók, algoritmusok és függvényobjektumok stb.)
- *Tervezés a C++ segítségével* (fejlesztés, tervezés, programozás)
- *Függelék*

Az utolsó fejezet jó példa arra, hogy a könyv a nyelv közvetlen ismertetésén jóval túlmutató témákat is igyekszik részletesen tárgyalni. A túlságosan szakmai tudnivalók, amelyek kevéssé illeszkedtek a könyv fő témaköreihez, a függelékbe kerültek. Ilyenek például a nyelvtan, a kompatibilitás, a szakmai részletek, a helyi sajátosságok (locale) és a kivételbiztosság a standard könyvtárban.

A részletes tárgymutató a könyv eredeti kiadásának is fontos és hangsúlyos része volt, így a fordítás tárgymutatójában angol címszavak is szerepelnek. A kiadvány részletes tartalomjegyzéke elérhető a cikk végén feltüntetett címeken.

Az egyes fejezeteket tanácsok és feladatok zárják. Formai érdekesség, hogy a kódpéldákat (ezek az eredeti kiadvány forrásából származnak) az elírások, illetve a nyomdahibák veszélyének csökkentése érdekében nem állandó, hanem változó szélességű, enyhén dőlt betűvel szedték. A magyar kiadás újítása, hogy a kiadványhoz szellemes, a C++ operátorait összefoglaló könyvjelző jár.

A különféle programozási trükkök ismertetésének, az egyes résztémák részletes kifejtésének, és a szakmai függelékeknek, valamint a részletes (részben kétnyelvű) tárgymutatónak köszönhetően a könyv azok számára is hasznos olvasmány, akik már ismerik a nyelvet.

A nyelvvel most ismerkedőknek akkor ajánlható a könyv, ha már legalább egy olyan nyelvet ismernek, mely a típusokat fordítási időben ellenőrzi (ilyen például a Pascal, a C stb.). Tekintve, hogy a könyv eleve nehéz és sűrű anyagot tárgyal, ezért a programozási alapfogalmakat nem magyarázza el, így első programozási könyvként nem ajánljuk.

Kapcsolódó címek

Bjarne Stroustrup honlapja
➲ http://www.research.att.com/~bs/

Az angol könyv honlapja
➲ http://www.aw.com./cseng/titles/0-201-88954-4

A magyar változat szaklektorának honlapja
➲ http://www.zolix.hu

A könyv részletes leírása (Kiskapu Könyvesbolt)
➲ http://www.kiskapu.hu/kiskapu/search.phtml?detailed=120233201

Üdvözlet

Egy induló rovat első megjelenésekor általában az író bemutatkozik – Helló, a nevem Leslie, és én leszek a rovatszerkesztő – és kijelöli a számára kívánatos témakört – Üdvözöl a .org-Figyelő! A fejlesztések és irányzatok nyomon követésével kívánunk foglalkozni. Nagyító alá veszünk egy-egy .org-ot vagy megvitatjuk az irányzatokat, esetenként rövid híreket jelentetünk meg. Tényleg, ki vagyok én, hogy ellenálljak ennek a nemes hagyománynak? Tehát, üdvözöl a .org-Figyelő! Mindenekelőtt, amiről beszélni fogok, nagyon-nagyon kézenfekvő, a GNU/Linux .org közösség története egyben a Linux története is. A fejlesztések és irányzatok nyomon követése – folyjék az bárhol, bármilyen kis csoportban – létfontosságú, mert számos kitűnő újítás a .org-okból jön.

A Linux Greenhouse

A Linux Greenhouse ➲ http://www.linuxgreenhouse.org/ „virtuális inkubátorként” jellemzi magát. De ne hagyd, hogy megijesszen az „inkubátor” szó. A Linux Greenhouse teljesen különbözik azoktól, melyeket nemrég tűz alá vettek. Indulásképpen, ez egy nonprofit szervezet, mely nem fizettet tagsági díjat, valamint nem igényli saját tőke kockáztatását sem. Sok tekintetben a Linux Greenhouse sokkal inkább egy laboratórium, mintsem inkubátor.

A csatlakozni kívánó cégeknek, illetve projekteknek nyílt forráskódú programokat kell használniuk, valamint magas színvonalúnak kell lenniük. Ha e követelményeknek eleget tesznek, kapcsolatba léphetnek Linuxon alapuló különféle üzleti megoldások vezetőivel. A Linux Greenhouse feladata, hogy létrehozzon egy olyan befektetési és fejlesztési bank- , marketing- és PR-vezetőkből álló csoportot, mely hajlandó egy kis időt és szakértelmet ajándékozni a többi Greenhouse-résztvevőnek. A cél az, hogy a részt vevő cégek hozzáférjenek olyan erőforrásokhoz is, melyeket a kezdeti időszakban elég nehéz elérni. Ezen túlmenően, a vállalati vezetőknek előnyös helyzetet teremt arra, hogy hasznos szövetséget kössenek kicsi, de magas színvonalú cégekkel, melyek eddig kívül estek a látókörükön.

Az első Linux Greenhouse osztálytalálkozó – mely egy hétig tartott – Korea fővárosában, Szöulban volt a Global Linux 2000 rendezvényen. Belga, kínai, angol, finn, francia, német, koreai, svéd és amerikai cégek és projektek vettek részt. A rendezvény témaköreiből: Linux-alapú intranetek Kínában, egy francia cég által fejlesztett kifinomult Linux-alapú környezet kézi eszközökhöz és webtervező rendszerhez, egy svéd projekt bemutatása, mely Linuxot tanít továbbtanuló diákoknak. A 2001. osztályt áprilisban választják ki (a jelentkezési lap február közepétől a Hálóról elérhető).

Rövid hírek

Szabad Program Alapítvány Európai (FSF Europe)

Az Európai Szabad Program Alapítvány ➲ http://www.fsfeurope.org/, a bostoni (Massachusetts) székhelyű Szabad Program Alapítvány (FSF) bejegyzett testvérszervezete. A MandrakeSoft 2500 euróval támogatta a szervezetet.

Köszönjük a MandrakeSoftnak a támogatást, mellyel a felmerülő költségeinket fedezni tudjuk – mondta Georg C. F. Greve, az FSF elnöke. Reméljük, hogy a jövőbeni tevékenységünkhöz jó együttműködést, tudunk kialakítani a Szabad Programok és a GNU/Linux érdekében.

Az FSF Europe jelenleg is fejlesztés alatt áll. Márciustól elkezdték a munkát Németországban, Franciaországban, Svédországban és Olaszországban. Más európai országokban (Anglia, Belgium, Hollandia és Spanyolország) röviddel ezután jelenik meg. Reméljük hazánk is hamarosan beléphet az alapítvány tagjai sorába.

A GIMP

A GIMP (GNU Image Manipulation Program) ingyenesen elérhető program (lásd még a Linuxvilág február-március számának 86. oldalán, illetve a 68. oldalon), mely kitűnően használható egyszerű rajzprogramként, kiváló minőségű fotóretusálóként, kötegelt feldolgozáshoz, nagytömegű leképezőrendszernek, képátalakítónak és így tovább… Nemrégiben megjelent a legújabb megbízható változat, az 1.2, melyet a ➲ http://www.gimp.org/-ról lehet letölteni.

A GIMP tavaly elnyerte a Linux Közösség leghasznosabb irodai programja díját Münchenben, a Systems 2000 rendezvényen. *Carey Banks*, *Marc Lehmann* és *Simon Budig* vette át a díjat – a háromezer márkáról kiállított csekket.

KDE

KDE ➲ http://www.kde.org/ kibocsátotta az Xparts csomagot, melyet *Matthias Ettrichm*, *Simon Hausmann* és *Lars Knoll* írt. Ez kibővíti a Kpartot, képessé teszi külső elemek beágyazására. Az Xparts együttműködési lehetőséget biztosít a fő Unix/Linux eszközkészletek és alkalmazások között, mint amilyen például a Mozilla. A csomaggal lehetőség van arra, hogy a Mozilla megjelenítő motorját (Gecko) használjuk a Konquerorban a KHTML helyett. Ezt futásidőben, egy párbeszédablakban állíthatjuk be, a Konqueror megváltoztatása nélkül. Az Xparts lehetővé teszi bármilyen területen dolgozó Unix- és Linux-fejlesztő számára, hogy KDE-elemeket készítsen – mindegy, melyik eszközkészletet vagy környezetet használja.

Leslie Proctor (lesproctor@yahoo.com) a .org közösség tevékeny résztvevője, a világ számos .org szervezetében tanácsadóként tevékenykedik.

Descent 3 Linuxra

Véleményezésünk tárgya ismét egy remek Loki játék. Becsszóra, semmi elfogultság, de a Loki annyira gyorsan fejlődik, és oly sok üzlet polcain megtalálhatók játékai, hogy időről időre elkerülhetetlen a vele való találkozás. Vár a sorára néhány nem lokis játék is, valamint nemsokára az ingyenes játékoknak is helyet szentelünk hasábjainkon.
Komolyan gondolom, hogy ez egy remek játék. A Descent 3 az első a sorozatból, amely megjelent Linuxra, de úgy tűnik, hogy mind közül ez sikerült a legjobban. Köszönettel tartozunk az Outrage és a Loki fejlesztőinek, hogy végre hozzájuthattunk a dobozhoz. Néhányan biztosan emlékeztek még a Descent első részére, azokra a földalatti útvesztőkre, a súlytalan lebegésre a járatokban, a szédületesen valósághű 3D-s irányításra, amit a játék hatutas mozgásirányító rendszerének köszönhettünk. Az első Descentnek azonban akadtak gyenge pontjai is, ilyenek voltak az ismétlődő, nagyon hasonló pályák és a szegényes grafika. A második rész sem törekedett arra, hogy ezeket a hiányosságokat orvosolja, de bővítették a támogatott eszközök körét, javítottak a játék vezérlésén, és hozzáadtak néhány új fegyvert az arzenálhoz. A Descent 3 végre meghozta azt, amire mindnyájan vártunk: a játék története érdekes, a pályák egyediek, a feladatok sokrétűek, mindez úgy, hogy a játék nyújtja azt a megszokott érzést, amit az ember elvárt a Descent-sorozattól.

A történet

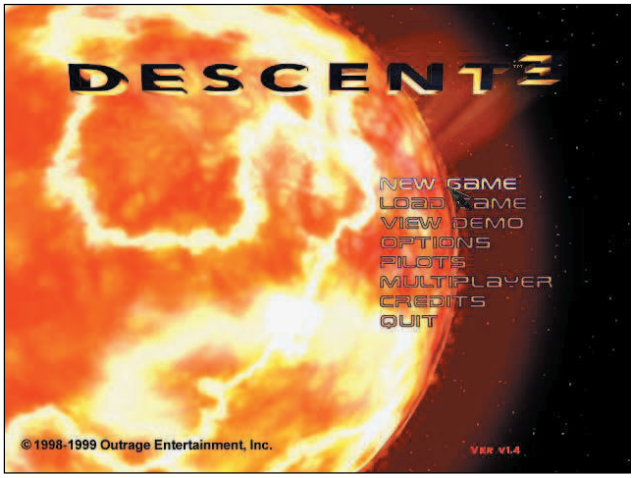
A történet nem túl összetett, de szórakoztató és jól írták meg (végre!). A látványos nyitófilm, majd az azt követő bejátszás öt-ötpercnyi 3D-ben felvázolt történet. A Descent 3 onnan folytatódik, ahol az előző rész befejeződött: eszméletlenül sodródsz az űrben, a világodat – a Descent 2 végén – megrázó robbanás után. Az utolsó pillanatban, amikor a hajóddal éppen tehetetlenül a Napba sodródnál, néhány új barát megment. Elárulják, hogy régi alkalmazód, a PTMC, igazán ronda dolgokat művelt. Nanovírusokat fejlesztettek ki, melyek a bányászrobotokból vérszomjas katonai eszközöket varázsoltak. Ráadásul ők tehetők felelőssé a hajód megrongálásáért, sőt ők próbáltak megölni! Jól van na, nem egy Herman Melville, de a történet elég jó ahhoz, hogy fenntartsa az érdeklődésedet. A játék fennmaradó része arról szól, hogy repkedsz az új hajóiddal – akár hármat is kaphatsz –, megpróbálod áthúzni a PTMC számításait, megmented a vírus fejlesztésében részt vevő „key detector”-okat, nyilvánosságra hozod a PTMC disznóságait, és igyekszel minél több borsot törni a volt alkalmazód (és leendő gyilkosod) orra alá. A történet gördülékenyen bontakozik ki, hála a 3D-s bejátszásoknak és a rádióadásoknak, sőt váratlan fordulatokban is bővelkedik.
A Descent 3-at a történet ismerete nélkül is lehet játszani, és a cselekmény elég jó ahhoz, hogy hosszabb időre lekösse a figyelmedet.

Előnyök

- Érdekes történet
- Bámulatos grafika
- Csúcs hangok
- Figyelemre méltó többjátékos üzemmódok

Hátrány

- Néhányan valószínűleg jobban örülnének a botkormánynak, mivel a Descentet elég nehéz irányítani

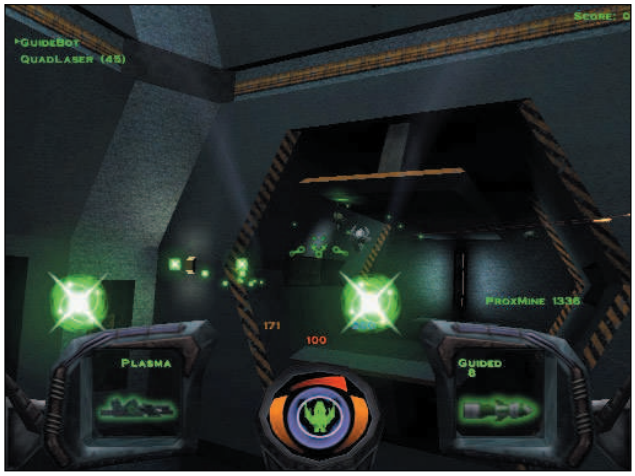


Adatok

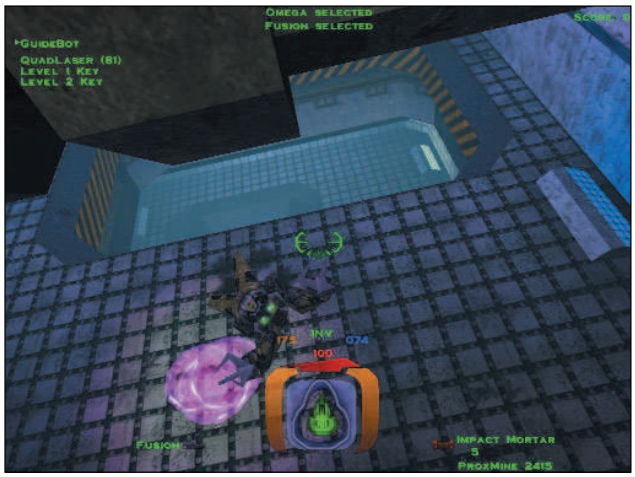
Descent 3 linuxos változat
Gyártó: Loki Entertainment Software
E-mail: sales@lokigames.com
➲ http://www.lokigames.com/
Ár: kb. 9000 Ft

A játékmenet

A Descent már a kezdetektől remekül ötvözte az egyszemélyes lövöldözős játékok egyszerű kezelhetőségét az űrhajó mozgásszabadságával. Ennek azonban megvan az ára, és sok játékos számára sokkal nehezebb a billentyűzetet használni, mint a botkormányt.
A mozgásnak három tengelye van, és három irányban haladhatunk, ezért nem nehéz elképzelni, hogy küzdelem közben nem könnyű a helyzetmeghatározás és az összehangolt mozgás, miközben szűk alagutakban ellenséges robotok támadnak ránk.
A játék 15 küldetésből áll és mindegyiknek saját célja van. Annak ellenére, hogy még mindig az összes küldetésre jellemző a „találd meg a kulcsot, mely a célhoz vezető ajtót nyitja” indíték, azért a legtöbb pályán már más, érdekes másodlagos célokat is el kell érnünk, valamint alkalmanként rejtvényeket is meg kell oldanunk, ezek egy része igazán kemény dió. Az ellenfél mesterséges értelme kifejezetten jó, ez nagy fejlődés a régi Descent-változatokhoz képest. Hamar rájössz, hogy most csoportosan támadnak és terv szerint cselekszenek – ez gyakran annyit jelent, hogy veszély esetén átrendezik a csoportot. Az ellenfelek kapcsolatban állnak egymással és jelenlétedről tájékoztatják a körzetben lévő robotokat, ennek eredményeként megérkezik az erősítés, ami még több fejtörést okoz. Vannak ugyanis olyan robotok, amelyek egyenesen rád támadnak, mások lesben állnak és várják, hogy elhaladj mellettük, a legroszszabb esetben akár a hátad mögé is lopakodnak. Szerencsére a Descent 3-ban rendelkezésedre áll néhány teljesen új fegyver a hitvány fráterek sütögetésére.
Az Outrage másik diadala az új Fusion megjelenítőmotor, mely új távlatokat nyit, ugyanis lehetővé teszi a felszíni küldetéseket. Nem kell többé a bezártság érzésével küzdened valami sötét bánya mélyén, szárnyalhatsz a bolygó felszínén is. A földalatti pályák is jól tervezettek, változatos a díszlet, gyakran hihetetlenül összetett a térbeosztás és az összhatás bámulatos. Minden szint óriási és bonyolult, így az egyszemélyes játék tizenöt küldetésének mindegyike komoly erő- és időráfordítást igényel. A Descent 3-ban nem vagy egyedül, hiszen ismét itt van és segít a kis haver, a „guidebot”



A plazmaágyú: amikor mindenképpen minden ellenséges PTMC-robotot meg kell semmisíteni a teremben



Vedd le a HUD-ot az olyan küzdelmek során, amikor a térlátás nagyon fontos

a Descent 2-ből. A guidebotok olyan kis robotok, amelyek elkísérnek az alagutakban, sőt a hajónkon is jó hasznát veheted a gályázó robotsegítőnek. Szükség esetén elindíthatod egy-egy feladattal, például kivezethet a végtelennek tűnő útvesztőből, hogy megtaláld a következő célod (türelmesen megvárja, hogy kövesd), tud tüzet oltani, vagy elvezet a következő robotellenséghez. Alkalmanként a guidebot kicsit megzavarodik, elfelejti a következő célt, és néha beszorul lehetetlen helyekre, de mindent összevetve nélkülözhetetlen segítőtárs.

A Descent 3 kép- és hangminősége igazán magával ragadó: a grafika csodálatos, a környezet befolyásolható és rombolható, a különleges hanghatások valósághűek és kristálytiszták (ezeknek köszönhetjük, hogy igazi alagút hatását kelti a környezet). A háttérzene sem csak annyira jó, hogy bekapcsolva hagyjuk, érdemes még felhangosítani is. A Descent OpenAL-rendszerű hangjához csak annyit tennék hozzá: kár hogy nem volt nagyobb hangfalam és megértőbb szomszédom.

Többjátékos üzemmód

A Descent 3 többjátékos támogatása egyszerűen káprázatos. Kilenc játékbeállítás közül választhatsz, ezek között megtalálható az elterjedt Capture the Flag és Deathmatch üzemmódoktól a sokkal kevésbé elterjedt (de gyakran hiányolt) Cooperative módig minden. A Monster Ball mód olyan, mintha fociznál az űrhajókkal. Játszhatsz TCP/IP-alapú üzemmódban közvetlen kapcsolattal vagy kereshetsz játékpartnert az Outrage Parallax Online (PXO) játékhálózatán. A descent3 binárisa futtatható kiszolgáló üzemmódban is, azoknak a megszállottaknak, akik saját Descent 3-kiszolgálót akarnak üzemeltetni.

Linuxos különlegességek

A Descent 3 telepítése helyigényes, elfoglalhat akár egy gigát is, de a Loki telepítőjével lehetőség nyílik sokkal helytakarékosabb megoldásra: a filmbejátszások egyenesen a CD-ről is lejátszhatók. A játék nem igényel különösebben erős gépet, főleg ahhoz képest, hogy milyen élményt nyújt. A Loki 300 MHz-es processzort ajánl (sőt, akár 200 MHz-eset, mint legalsó korlát) és 64 mega memóriát. A Loki azonban elfelejt figyelmeztetni arra a buktatóra, hogy ha a filmbejátszásokat CD-ről futtatod, kicsit gyorsabb CD-meghajtóra lesz szükséged, mint az általuk megjelölt hatszoros alsó határ. Még az én régi nyolcszoros meghajtóm is nyüszített egy-egy mozgalmasabb jelenetnél. A programkörnyezettel szemben támasztott igények is a szokásosak: Linux 2.2.x rendszermag és Glibc 2.1, és a hálózatos játékhoz szükséges ezenkívül a beállított TCP/IP-támogatás, valamint internetes vagy helyi hálózati kapcsolat. A Descent 3-hoz elengedhetetlen egy jó 3D gyorsítókártya. A játék számos kártyaváltozatot támogat, ezeket rendesen ki is próbálták, többek között ilyen a népszerű 3dfx Voodoo sorozat (a szép, erre a célra megírt, Glide megjelenítőnek köszönhetően), a Matrox G200/G400 sorozat (az Utah-GLX és a Mesa meghajtók segítségével) valamint az nVidia OpenGL-képes X kiszolgálói által támogatott nVidia kártyák (a GeForce GTS tökéletesen működött). A teljes listához érdemes ellátogatni a Loki honlapjára: ➲ http://www.lokigames.com.
A felszerelés további kiegészítője lehet a Rock'n Ride játékülés, ami beszerezhető innen: ➲ http://www.rocknride.com/.
Nem kell magyaráznom, hogy óriási ez a szerkezet! A Loki volt olyan szíves, és elárulta azt is, hogy hogyan lehet az új Descent 3: Mercenary bővítő csomagot használni a linuxos változattal.

Összegzés

Mit mondhatnék még? Ez a játék nagyon szórakoztató, nagyon jól írták és valósították meg, remek élményt nyújt. A grafika kivételes, a hanghatások tiszták és jó minőségűek, még a zene is jó. A mozgás kicsit összetett, de nem is várhatunk mást, hiszen a hajó legfinomabb mozgását is irányítani tudjuk. Az általános vezérlőelemek egyszerűek, könnyű kezelni őket és logikus az elrendezésük. A többszereplős üzemmód csúcs, a 3D támogatás hibátlan. Igazán kiváló játék: az egyik legjobb az egyszemélyes lövöldözős műfajban.

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).

Heavy Gear II/nVidia frissítés

Mike Phillips a Loki Entertainment Software-től írt nekünk, hogy elújságolja: „a Heavy Gear II esetében a 2D-3D váltásnál tapasztalt gondokat kijavítottuk a legújabb nVidia meghajtókban (változat 0.9-6)." (Lásd Linuxvilág február-márciusi szám, 112. oldal.) A GeForce-os rendszeremen végzett próba ezt igazolni látszik, köszönjük Mike!

Tanácsot adni vagy nem adni...

Manapság egyáltalán nem könnyű feladat linuxos szaktanácsadóként dolgozni. Számos különböző rendszerrel dolgoztam már: Solaris, SunOS, Ultrix, OpenServer, AIX, HP-UX, és természetesen Linux. Mi a leglényegesebb különbség az imént felsorolt Unix-változatok és a Linux között? A legnagyobb különbség valószínűleg az, hogy bármelyik Unix-változatot tekintjük is, a különböző telepítések alapvetően megegyeznek egymással. A telepítés és a rendszer felügyelete egységes. Ezzel szemben a Linux esetében minden változat saját telepítőeljárásokkal és felügyeleti parancsfájlokkal bír. Szerencsére azonban mindegyikük Linux, tehát a változatszámtól eltekintve nagyon sok dolog megegyezik. A DNS ugyanaz; a DHCP ugyanaz; és természetesen maga a rendszermag is ugyanaz. Ha linuxos szaktanácsadással foglalkozunk, vagy ezzel szeretnénk foglalkozni, akkor a Linuxszal történő ismerkedés során célszerűbb alulról felfelé haladni (például a parancssorral kezdeni), mint felülről lefelé (X kiszolgáló és az adott változathoz tartozó felügyeleti eszközök megismerése). Tanuljunk meg parancsfájlokat olvasni, és értsük meg a működésüket. Ahol mód van rá, keressünk és telepítsünk változatfüggetlen eszközöket, ilyen például a webmin, amely SSL-en keresztül biztonságossá tehető. Ha a /etc/named.conf olvasása már jól megy, akkor a DNS-sel is könnyen megbirkózunk majd. Ugyanez igaz a rendszer összes többi szolgáltatására is. A webmint nem saját magam, hanem az ügyfeleim számára telepítem. Én tulajdonképpen soha nem használom. Azt hiszem, hogy ez a megközelítés sok fejfájástól kímélhet meg bennünket, mivel így az ügyfeleink azt a változatot használhatják, amelyik nekik a legjobban tetszik (vagy amelyik a leginkább megfelel a céljaiknak), és nem kell újratelepítenünk az egész rendszert csak azért, mert nem ismerjük az ügyfél által használt változatot.

MonMotha IPTables-kezelő parancsfájl

Általában senkinek sem javaslom a tűzfaleszközök és a tűzfal-parancsfájlok használatát, mire azonban az Olvasó ezt a cikket olvassa, valószínűleg már több olyan Linux-változat is létezik, amelyik az új 2.4-es rendszermagot, és vele együtt az új hálózati szűrőt tartalmazza. A fentebb említett parancsfájl sokat segíthet, ha gondjaink akadnak az IPTables kezelésével. Ez a parancsfájl jó kiindulópontot ad és egyúttal megfelelő egyezséget is kínál. A szerző jól kihasználja a hálózati szűrő nyújtotta előnyöket. A program tehát egy jó kiindulópont, célszerű azonban átnézni és a saját igényeinkhez szabni.
Szükséges: iptables, sh.
➲ http://t245.dyndns.org/~monmotha/firewall/index.php

plbackitup

Ezzel a Perl programmal biztonsági másolatot készíthetünk könyvtárainkról. Meghatározhatjuk azt, hogy a kiválasztott könyvtár alatt elhelyezkedő könyvtárak és fájlok közül melyek kerüljenek be a másolatba és melyek ne. A biztonsági másolat egy helyi fájl formájában jön létre. Ha a másolatot máshol szeretnénk tárolni, akkor FTP segítségével továbbíthatjuk azt egy központi kiszolgálóra. Mivel én magam több rendszerhez használom ugyanazt a központi tárolókiszolgálót (és ez az egyetlen rendszerem, melyhez szalagos egység csatlakozik), nagyon jó hasznát veszem ennek a segédprogramnak.
Szükséges: Perl, szabványos unixos eszközök (tar, touch, rm stb.).
➲ http://www.glandrake.com/scripts.html

poppy

Egy (majdnem) egyetemes parancssori levelezőprogram, amely POP3- és IMAP-kiszolgálókkal egyaránt képes működni. A program igazán jól használható. Gyakran megesik, hogy hatalmas méretű leveleket kell letöltenem a lassú telefonos kapcsolaton keresztül. E helyett ezzel a kis programmal gyorsan végigfuthatok az üzenetek tárgyain, illetve magukon a leveleken is, törölhetem azokat, vagy akár azonnal válaszolhatok is rájuk.
Szükséges: Perl.
➲ http://home.sprynet.com/~cbagwell/projects.html

Apache Toolbox

Szükségünk lenne egy kibővített Apache-ra, amely gyors és megbízhatóan működik? Még soha nem fordítottunk Apache-ot? Ha mindkét kérdésre igennel válaszoltunk, akkor bizony bajban vagyunk. De nyugodjunk meg, az Apache Toolbox megoldást kínál a gondunkra. Még azt a mod_perl/php4 ütközést is ismeri (és figyelmeztet is rá), amelyik kifagyáshoz vezethet. Én magam számtalan alkalommal fordítottam és telepítettem már egyedi Apache-rendszereket, ennél egyszerűbben azonban még soha. A program nem tökéletes ugyan, teljesítményét azonban kizárólag a legtapasztaltabb Apache-szakik múlhatják felül. A program nem teszi lehetővé, hogy a php3-at és a php4-et egyaránt telepítsük. Ha mindkettőre szükségünk van, akkor az egyiket magunknak kell a rendszerhez illesztenünk a későbbiekben.
Szükséges: sh, wget.
➲ http://www.apachetoolbox.com/

Indexpage

Egy halom képet (jpeg), szeretnénk egy weblapon elhelyezni? Van egy könyvtáram tele saját készítésű képekkel. Egyszerűen bemásoltam ezt a Perl programot a könyvtárba, gyorsan összehoztam a leírásokat tartalmazó fájlt, lefuttattam a programot, és máris létrejött négy html oldal a képek kicsinyített változataival. A képméret nem számít: a program úgy nyújtja meg, illetve nyomja össze a képeket, hogy azok éppen illeszkedjenek a számukra létrehozott dobozokban. Ha megváltoztatunk valamit (például új képek kerülnek a könyvtárba), akkor a html oldalak frissítéséhez elég újra lefuttatni a programot.
Szükséges: ImageMagick, Perl és az Image::Size Perl-modul.
➲ http://www.lysator.liu.se/~unicorn/hacks/indexpage/

tcpspy

Szeretnénk utánajárni, hogy ki hol, mikor és hogyan kapcsolódott a rendszerünkhöz? Ez a program naplózza a kapcsolatok létrehozását, bontását, a felhasználókat, a helyi IP-cím:kapu párokat, a távoli IP-cím:kapu párokat, sőt még a programokat is. Természetesen lehet, hogy egyszerűen nem akarunk tudni ezekről a dolgokról. Mindenesetre érdekes lehet megtudni, hogy ki, mikor és milyen célpontokra futtatta rendszerünkből az nmapot. Mivel a program a syslogra épül, az összegyűjtött adatokat továbbíthatjuk egy központi naplózó kiszolgáló felé. Alapértelmezés szerint a tcpspy a LOCAL1 naplózó szolgáltatást használja, ezt azonban a Makefile-ban bármi másra megváltoztathatjuk.
Szükséges: glibc.
➲ http://box3n.gumbynet.org/~fyre/software/

David A. Bandel