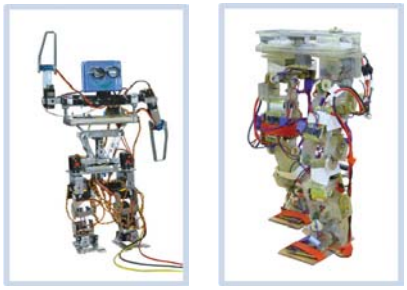
Hírek

Mikrobi



Nemcsak *Japán*, de *Franciaország* is élen jár robotikában. Az *Aldebaran Robotics* 2007-re ígéri az első emberszabású háztartási robotukat, *Nao*-t. *Nao* autonóm és könnyen programozható robot lesz. A robotot gazdája *WIFI*-n keresztül tudja vezérelni, de képes lesz a robot szkriptek segítségével *autonóm működésre* is. A robot létrejöttét a *nyílt forrású URBI (Universal Real-time Behaviour Interface)* eszközkészlet teszi lehetővé.

➲ http://www.linuxdevices.com/news/NS6263763539.html

Motorola zenefon

A *Motorola* válaszul a *Sony-Ericsson* zenetelefonjaira kiadta saját telefonját *Rokr E2* néven. A telefonban – azontúl, hogy *Linux*ot futtat – megtalálhatóak a ma már megszokott funkciók: *USB2*-es adatátvitel, *FM rádió*, *Bluetooth* és vezetékes headset csatlakozás, valamint *1.3 megapixeles kamera* amely egyedinek mondható *15 képkocka per másodperces* videórögzítési képességgel bír. Webböngészőnek *Operát* használhatunk a *2 hüvelykes 240x320 képpont* felbontású kijelzőn.

➲ http://www.linuxdevices.com/articles/AT8243331060.html

Feltörték a Debian egy gépét

Feltörték a *Debian* projekt egyik gépét, a *gluck.debian.org*-ot. Ez sajnos nem az első eset, *2003 novemberé*ben ennél súlyosabb volt a helyzet. A karbantartók amint észrevették a törést, lekapcsolták a hálózatról a gépet és elkezdték újratelepíteni. A bejelentés szerint egy *friss exploit*ot használhattak ki a támadók.

➲ http://www.zdnet.com.au/news/security/soa/Debian_server_hacked/0,2000061744,39263270,00.htm

Nyílt szabványú grafikus kártya

Timothy Miller – talán a *Sun* hasonló lépése nyomán – elkezdte az *Open Graphics Project*-et. Ennek a célja nem kevesebb, mint megalkotni egy mai mértékkel is *komoly grafikus kártyát*, melynek mind a *nyomtatott áramkörei*, mind a *meghajtóprogramja Gnu GPL* licensszel bír majd. A kártya kezdeti ára 1000 dollár körül alakul majd, de amint beindul a *tömegtermelés*, *200 dollár körülire* csökkenhet, mely nagyjából a mai kártyák árainak felel meg.

➲ http://www.freesoftwaremagazine.com/node/1665

8 gigabites flash memória

A *Samsung* elkészítette az első *8 gigabites* (1 gigabájtos) flash memóriát – jelenleg a legkisebbet –, melyet elsősorban mobiltelefonokba és mp3 lejátszókba szánnak. A chip az eddigi 70 nanométeres helyett *60 nanométeres technológiával készült*, melynek köszönhetően az ára csökkenni, a termelékenység pedig nőni fog. A cég továbbá az év végére ígéri 8 gigabájtos flash memóriáját, mely nyolc darab egymásra helyezett rétegből áll majd össze.

➲ http://www.linuxdevices.com/news/NS4610471086.html

32 mag 2010-re?

Az *Intel 2010-re 32 magos processzorral* jelentkezik majd, amelynek révén a mostanihoz képest akár *15-szörös teljesítménynövekedés* várható. Az *Intel feladni* látszik saját – korábbi – *20 Ghz-es* processzorokról szóló *elképzeléseit*.

Hatkerekű autonóm jármű

A francia *Robosoft* cég – amely alapvetően háztartási takarító robotokat fejleszt – bemutatta hatkerekű robotját, mely a majdnem *50 centiméteres kerékátmérőjével* szinte bármilyen akadályt képes legyőzni, legyen az lépcső vagy szikla. Maga a robot körülbelül *150 kilogramm* és mintegy *100 kilogramm hasznos tömeget* képes szállítani akár *négy órán keresztül*. A moduláris rendszer *Linuxot futtat*.

➲ http://www.linuxdevices.com/articles/AT8656331847.html

Mütyűrmánia

Mitől függ vajon, hogy egyesek rajonganak az új technikai dolgokért, míg mások kevésbé? Nos a tudósok szerint ez genetikailag van kódolva bennünk. Ennek neve is van: *neophilia*. Az egészet egy *enzim* okozza, mely magasabb arányban van jelen, és az új dolgok ennek kapcsán jobban *stimulálják* ezeket az embereket, mint másokat.

➲ http://slashdot.org/article.pl?sid=06/07/12/1245254

Biztonsági videórendszer

Az ***Amptronix*** 4 csatornás ***videórendszere – DVR-354 – Linuxot*** futtat, és elsősorban biztonsági figyelő szolgálatok számára készült. Az eszköz egyszerre ***négy kamera*** képét képes ***MPEG4*** adatfolyammá alakítani ***640x480*** képpontos felbontásban. A videó ***2.5 hüvelykes merevlemezre*** vagy ***CompactFlash*** memóriakártyára rögzíthető. A rendszer képes észlelni a kamerák képein a mozgást, illetve ennek megfelelően riasztani is SMS-ben. ***Opcionálisan wifi*** is rendelhető hozzá.

➲ http://www.linuxdevices.com/news/NS8437427336.html

Japán robot

Négy japán cég – ***General Robotix, Moving Eye, Pirkus Robotix, Dai Nippon Technical Research Institute*** - összefogásával elkészült egy alacsony előállítási költségű emberszabású linuxos robot – ***HRP-2m Choromet*** –, melyet elsősorban oktatási és kutatási célokra ajánlanak. A robot lelke egy névjegykártya méretű számítógép ***240 Mhz-es SH4***-es processzorral és ***32 megabájt memóriával***. A robot egyediségét az adja, hogy képes a ***testének*** egyes részeire ***ható erőt érzékelni***, és ezáltal akár magától, emberi beavatkozás nélkül is felállni.
A robot előreláthatóan ***szeptembertől*** lesz kapható kb. ***4250 dolláros áron***.

➲ http://www.linuxdevices.com/news/NS8377820601.html

Profi hangfeldolgozó már ezer dollárért

Októberben szállítják a ***Trinity Audio Group*** legújabb ***Linux*** alapú hangrögzítő és feldolgozó eszközét (***DAW***), mely nem más, mint egy erre a célra kihegyezett ***PDA***. A rendszer lelke egy ***Intel PXA270***-es processzor.
A ***PDA*** kialakításnak köszönhetően hosszabb az akkumulátoros üzemidő (mintegy 4 óra) és egyszerűbb a használata is. A hangfelvétel minőségét jelzi, hogy ***16*** vagy ***24 bites*** felvételeket készíthetünk legfeljebb ***96 kHz***-es mintavételezéssel (192 kHz csak később várható), és felvétel során a két bemenetet egyszerre is használhatjuk.
Az eszköz ***1.8 hüvelykes 20 gigabájtos Hitachi*** merevlemezre rögzíti a felvett anyagot, amely a beépített ***wifi***vel vagy ***USB***-vel tölthető át ***PC***-re.
A hangrögzítést egy ***ARM*** processzorra portolt ***1.3-as*** verziójú ***Audacity*** végzi, mely képes többcsatornás hangrögzítésre is.

➲ http://www.linuxdevices.com/articles/AT6071673793.html

Google Map Mobile

A ***Google,*** haladva a korral elkészítette a ***Google Map*** mobiltelefonos változatát is. A használatához a mobiltelefonra optimalizált ***Google Map java-s kliensét*** kell letöltenünk.

➲ http://www.google.com/gmm

Medve Zoltán
(e-medve@e-medve.hu)

2001-ben kezdett „Linuxolni”, de már korábban is ismerkedett a szabad szoftverek világával.
Ha éppen nem a gép előtt ül, akkor fotózgat, olvasgat vagy bicajozik.

Mi újság a rendszermag fejlesztése körül

Folytatódik a *git* változatkövető rendszerrel kapcsolatos munka. A közelmúltban *Linus Torvalds* kiküszöbölte a külső *diff* programtól való függést. Ennek eredményeként a végrehajtási idő *Linux* alatt hatodára, *Cygwin* alatt pedig tizenötödére csökkent. Mások eközben azon dolgoztak, hogy színes kimenetet varázsoljanak elő a *diff*-ből. Szintén a gittel kapcsolatos újdonság *Petr Baudis* munkája, aki az átnevezés támogatásán dolgozott. A *git* egyik nagy újítása az, hogy a névmódosulások nem a kérdéses tartalom módosításakor, hanem csak a fájl olvasásakor jutnak érvényre, amikor valaki az érintett kódrészlet történetét kéri a le a rendszertől. Bár ennek az olvasás közbeni névváltozásnak az érzékelése meglehetősen összetett feladat, *Petr*, és sokan mások, köztük maga *Linus* nagy buzgalommal álltak neki a megvalósításnak, és erőfeszítéseiket mostanra siker koronázta. A rendszermag egyes részei, különösen a *DevFS* lassanként „bárd alá” kerülnek. A *DevFS*-t a 2.6.13-as magváltozat óta nem lehet használni, *Greg Kroah-Hartman* pedig elkészítette azokat a foltokat is, amelyek végleg törlik a *DevFS* kódrészleteit az elkövetkező kiadásokból. A *blkmnt* meghajtó, amely azt teszi lehetővé, hogy memóriaeszközök blokkeszközökként jelenjenek meg a rendszerben szintén rohamléptekkel halad a nirvána felé. Az ugyanezt a szerepet betöltő *block2mtd* meghajtóval kapcsolatban körülbelül egy éve semmiféle hibát nem tapasztaltak, ráadásul a *blkmtd* összeférhetetlen a *H. Peter Anvin* által vezetett *klibc* projekttel is, amely immár a hivatalos kernelfába való bebocsátásra vár. *Andrew Morton* kifejezetten boldog volt, amikor átadhatta *Linusnak* a *blkmtd* meghajtó eltávolítását végrehajtó foltot jóváhagyásra. Még csak további érlelődési időt sem kért neki a *-mm* fában. Mindeközben *H. Peter Anvin* a *Klibc*-n dolgozik, amely nem más, mint egy kis méretű, a kernelen belül megvalósított *libc*. A dolog értelmét az adja, hogy a *Klibc*-re támaszkodva számos kernelfejlesztési projekt a felhasználói térben valósítható meg, hiszen a fejlesztők biztosak lehetnek benne, hogy ha szükségük van egy interfészre, akkor az a rendelkezésükre fog állni. Ez a fajta fejlődési irány egyébként idővel várhatóan azt eredményezi majd, hogy a *Linux* egyre inkább elkezd hasonlítani a mikrokerneles architektúrákra, azzal az – el nem hanyagolható – eltéréssel, hogy nem lesz ettől lassúbb. Évekkel ezelőtt a mikrokernel-alapú rendszerek ezen a problémán buktak meg és sodródtak a perifériára. *Linus* már évekkel ezelőtt megfogalmazta azt az elképzelését, hogy a rendszermagnak idővel a lehető legteljesebb mértékben modulárissá kell válnia. Ennek megfelelően régóta érzékelhető az a törekvés, hogy amit ki lehet venni a magból, azt ki is kell. És bár a fejlesztők nem egy szuszra oldják meg a dolgot, hanem sok apró lépésben, a *Klibc* egyértelműen olyasvalami, ami ezt a folyamatot katalizálja. A dokumentáció amolyan ritka és értékes drágakőnek számít a nyílt forrású programok világában. Nemrég *Chuck Ebbert* összeállított egy jelentős kiegészítést a *ptrace(2)* súgóoldalához, ami a kernelfejlesztők levelezési listája, a forráskód, illetve a szerző saját tapasztalatai alapján készült. Többen, közöttük *Daniel Jacobowitz*, aki a *Chuck* által dokumentált funkciók túlnyomó többségét megvalósította örömmel üdvözölte a kezdeményezést. Danielnek mellesleg számos javaslata is volt az anyag javításával kapcsolatban. A 2.4-es kernelfa *Will Tarrear* megfogalmazás szerint továbbra is „mélykarbantartási üzemmódban” leledzik. *Herbert Rosmanith* rákérdezett, hogy a *TPM*-et átemelik-e a 2.6-osból a 2.4-es fába, de minden jel arra mutat, hogy ez nem fog megtörténni. *Willy* továbbra is fönntart számos, a legkülönbözőbb helyekről összegyűjtött 2.4-es foltot, de nem föltétlenül abban a reményben, hogy azok valaha is bekerülhetnek a *Marcello Tosatti* által felügyelt hivatalos 2.4-es fába. A cél inkább csak az, hogy a régebbi kernelt használóknak meglegyen a lehetősége arra, hogy a legújabb meghajtókat is beüzemelhessék rendszereiken anélkül, hogy teljesen átállnának egy a 2.6-os magon alapulóra. Tekintettel arra, hogy a hivatalos 2.4-es mag mostanra megállapodottnak tekinthető, a 2.6-os változat fejlesztőire elég nagy nyomás nehezedik a stabilitás növelésével kapcsolatban. Nemrég maga *Linus Torvalds* sürgette egy olyan szabály bevezetését, amely szerint új kódot legfeljebb csak egy adott hivatalos kernelváltozat kibocsátása utáni két hétben lehet benyújtani. Ezt követően már csak javítások küldhetők be. Egyesek persze tiltakoztak ez ellen, de összességében úgy tűnik, a közösség respektálta a javaslatot. És bár mindez azt jelenti, hogy a 2.6-os mag a folyamatos üzemidő szempontjából kifejezetten stabil lesz, az interfészek és a viselkedésformák különböző változatok közti egyeztetése továbbra sem megoldott. Ehhez valamilyen más ötletre lesz majd szükség. *Molnár Ingo* és néhány más fejlesztő megvalósította az úgynevezett *„pehelysúlyú, felhasználói térben működő prioritásöröklődést”*. Elmondásuk szerint ez fontos mérföldkő a valós időben működni képes, felhasználói térben futó alkalmazások felé. Ez egy meglehetősen sokat vitatott témakör, amire maga *Molnár Ingo* is utal a bejelentés szövegében. Meglátása szerint ez a szolgáltatás már „régóta ott keringett” a *Linux* körül, de ami a megvalósítását illeti, az mindig teljesítményproblémákkal küzdött, számos ponton hibás volt, és úgy összességében egy meglehetősen rendetlen kódrészlet benyomását keltette. És ez nyilván így is volt, hiszen általában is elmondható, hogy a valós idejű működés és a *Linux* összeházasítása igen ellentmondásos terület, elsősorban a rendszermagban jelentkező bonyolult kiegészítések miatt.

Zack Brown

Kicsi-kék és makaróni

A címre rácáfolva nem hupikék törpikék csőtészta-vacsorájáról szól ez írás, nem is Maradona túlsúlyáról, hanem a Sphenisciformes-béli Spheniscidae-családnak formabontásáról. Magyarán szólva: lássuk, milyenfajta pingvinek éldegélnek a Déli-sarktól egészen Afrikáig.

Azt mondják, kifordítva mindenki rózsaszín, ám maradjunk egyelőre a felszínen, és vizsgáljuk a madarat tolláról. Liliputi méretekkel bírnak, de igen színes egyéniségek az új-zélandi kék pingvinek, kik legfeljebb 1 kilogrammot nyomnak és maximum 40–45 centisre nőnek. A színeknél időzve, a legkisebb kékek szomszédságában találjuk a legritkább sárgákat: a sárgaszemű pingvineket, kiknek teljes populációját mindössze ötszáz darabra becsülik. Azt csiripelik a madarak, az oroszok azért vannak úgy oda *Brooke Shields*-ért, mert szemöldöke Brezsnyevre emlékezteti őket – talán az új-zélandiaknak is kedvencük e színésznő, hisz kedvencük a bóbitás pingvin is, jellegzetes sötétsárga, bóbita-szemöldökével. Egy (vagy több) hajszál (no meg az óceán) választja el előbbi fajtól a makaróni pingvint, s átfésülve a környéket, hozzá csatlakozik frizuradivatban az aranytollú, avagy sziklamászó rokon, kiről ha testessége okán nem is feltételeznénk akrobatikus elemekkel tarkított kötéltáncot a magasban, tarkított tincseivel dizájnos jelenség.

A madarak királya, vagyis a király-madár császári párjával együtt kimagaslik a pingvinalakúak közül: a métert is meghaladva, csőrük is meghaladja egy átlagos csőrhossz méretét. Ott tanyáznak ők, ahol a madár se jár: az antarktiszi jégpályákon korcsolyáznak. Vannak, akik nemcsak télre húznak délre, Tóbiás és társai, a pápaszemesek, Afrikából érkeztek. Bár sokan élcelődnek azon, ha az úszás olyan jót tesz az alaknak, miért a vastag zsírréteg és zsákszerű tor, no meg a potroh – e zsírréteget tápláló halvadászat során egy pingvin a vacsorájáért akár 110 kilométert is elúszkál, akár 20 km/h-s sebességgel. A kékségtől a szőkeségig 18-féle pingvint lelni, de csak egyetlen igazán veszedelmes ellenségüket: az embert...

Halusz Léna

Szabad Szoftverek Világnapja

Idén immár harmadjára kerül megrendezésre a Szabad Szoftverek Világnapja. Henrik Nilsen Omma ötlete nyomán legelőször 2004-ben került megrendezésre, akkor a világban 60 helyszínen, többek közt Szegeden is.

Az esemény fő célja a közösségépítés, bemutatkoznak szabad szoftveres projektek, és nem maradhatnak el a szabad szoftverekkel kapcsolatos technikai, gazdasági és jogi előadások sem. A részvétel a szponzoroknak hála – *SZTE Informatikai Tanszékcsoport*, illetve a *Neumann János Számítógéptudományi Társaság* – *ingyenes*, az előadások között pedig bárki találhat kedvére valót.

A *Kiskapu Kiadó* jóvoltából idén kedvezményes könyvárusítás is lesz, amelyen nemcsak szakmai, de közösségépítő könyvek is kaphatóak lesznek, mint például a *Szabad kultúra* vagy a *Katedrális és bazár*.
A program még jelenpillanatban is folyamatosan *változik*, érdemes tehát a *rendezvény honlapját* figyelemmel kísérni.
Az idei rendezvény kuriózuma várhatóan a *Magyar Informatika Történeti Múzeum meglátogatása* lesz, ahol rengeteg régi számítógé-pet tekinthet meg a kedves Látoga-tó. Ennek miéntjéről a honlapon található majd részletes információ.

Medve Zoltán
(e-medve@e-medve.hu)

Free = szabad vagy ingyenes?

A magyar nyelv árnyaltabb az angolnál, hiszen az angol *free* szónak nemcsak *szabad* a jelentése, de az *ingyenes* is. Azonban *Richard Stallman* szerint mégsem mindegy, hogy a programunk ingyenes, vagy szabad, azaz kiadjuk a forráskódját. Ugyanis a kiadott forráskód nem akadályozhat abban, hogy pénzt kérjünk a bináris termékért, vagy a beüzemeléséért nyújtott segítségért. Mindazonáltal a kiadott szabad forráskód lehetővé teszi a szoftver evolúcióját, hiszen a felhasználó belejavíthat és kiadhatja újra, természetesen a megfelelő induló licensz (*GPL*, *BSD*, stb.) figyelembevételével.

KAPCSOLÓDÓ CÍMEK

Szabad Szoftverek Világnapja (Szeged)
➲ http://www.inf.u-szeged.hu/~opensource/sfd/
Szabad Szoftverek Világnapja (nemzetközi)
➲ http://softwarefreedomday.org/
Magyar Informatikatörténeti Múzeum
➲ http://www2.u-szeged.hu/infmuz/


Linuxvilág
www.linuxvilag.hu

Új vállalati platform született – megjelent a Novell SUSE Linux Enterprise 10

A Novell SUSE Linux Enterprise 10 egy átfogó, nyílt vállalati platform, amely magában foglalja mind a szerver, mind a desktop oldali megoldásokat: a SUSE Linux Enterprise Server 10-et és a SUSE Linux Enterprise Desktop 10-et, valamint a Novell Customer Centert, egy on-line portált, ahol a műszaki támogatás mellett a szoftverfrissítések, hibajavítások is egy helyen elérhetőek. Jelenleg a Novell az egyetlen Linux szállító, aki ilyen szintű vállalati megoldást kínál.

A *Novell* 2006. július 17-e óta elérhető következő generációs nyílt vállalati platformja kiemelkedő fogadtatásban részesült a piacon és a médiában egyaránt, hiszen a kiadását követő tíz napban több mint 165000 példányban töltötték le a komponenseit. A *Novell.com* weboldalon a napi szoftverletöltések száma átlagosan 800 százalékkal emelkedett, azaz a bejelentés óta 5 másodpercenként töltenek le egy *SUSE Linux Enterprise* terméket.

„Az IDC prognózisa szerint 2008-ra a teljes Linux-piac értéke - beleértve a szervereket, asztali gépeket és a szoftvereket is - meghaladja a 35 milliárd dollárt, amelyből tisztán a szoftverek várhatóan több mint 14 milliárd dollárt tesznek majd ki. Az elemzői várakozások ismeretében büszkén mutatjuk be ügyfeleinknek új Linux munkaállomás- és szervertechnológiánkat. Biztosak vagyunk abban, hogy Magyarországon is népszerűek lesznek a SUSE Linux Enterprise 10 platform termékei és a Novell Customer Center szolgáltatásai, hiszen a hazai szakmai közönség már a béta változatokat is nagy érdeklődéssel fogadta. A most beépítésre került technológiai újdonságaink várhatóan Magyarországon is jelentősen előremozdítják a Linux vállalati bevezetését és alkalmazását” – nyilatkozta *Szittya Tamás*, a Novell Magyarország ügyvezető igazgatója.

„Mérettől függetlenül számítunk a kis-, közép- és nagyvállalatok valamint intézmények érdeklődésére is, mivel a termékek mind a felvonultatott technológiát, mind a termékek árát illetően versenyképes alternatívát jelentenek a hazai és nemzetközi piacon: a SUSE Linux Enterprise Desktop 47 Euró, a SUSE Linux Enterprise Server pedig 290 Euró ajánlott végfelhasználói áron rendelhető viszonteladói hálózatunkon keresztül” – tette hozzá *Szittya*.

SUSE Linux Enterprise Server

A különböző szervezeteknél számos eltérő típusú hardveren történő telepítéssel – többek között az *IBM* zSeries* mainframe Linux**-piac 80 százalékán – a *SUSE Linux Enterprise Server* a szerverfeladatok széles skálájához biztosít nyílt alapokat. Használható adatközpontokban és fiókirodákban egyaránt, és kiváló adatbázisok, az üzletmenetet segítő alkalmazások és a vállalat működése szempontjából kritikus fontosságú szoftverek futtatásához. A termékhez 7 éves terméktámogatás jár.

A *SUSE Linux Enterprise Server 10* integráltan és támogatottan tartalmazza a nyílt forrású *Xen* virtualizációs technológiát, ami lehetővé teszi különböző platformokon futó alkalmazások konszolidációját egy szerveren, ezáltal javítva az erőforrások kihasználtságát, növelve a hatékonyságot és csökkentve a költségeket. A *Xen* technológiát a *Novell YaST*-alapú (*Yet Another Setup Tool*, telepítő- és konfigurációs eszköz) menedzsment eszközei is támogatják, leegyszerűsítve a virtualizált megoldások bevezetését a vállalati infrastruktúrákon. A *SUSE Linux Enterprise Server 10*-ben található *Xen 3.0* támogatja az *AMD Pacifica* és az *Intel VT* hardveres virtualizá-

Magyarországon a jelenlegi legnagyobb szuperszámítógép is *SUSE Linux Enterprise Serveren* fut. Az *Országos Meteorológiai Szolgálat* számára kifejezetten szuperszámítógépes feladatok ellátására fejlesztett gép 144 darab, 1,5 gigahertzes és 6 megabájt L3 cache-sel rendelkező Itanium 2 processzorral, valamint 288 gigabájt memóriával szerelt *SGI Altix 3700 Bx2* számítógép.

A világ 10 legnagyobb szuperszámítógépe közül három fut *SUSE Linux Enterprise Serveren*:

- #1: A *Lawrence Livermore Nemzeti Laboratórium IBM Blue Gene/L-eServer* számítógépe (65,536 processzor)
- #4: A *NASA SGI Altix* 1.5 GHz *Infiniband* számítógépe (60,960 processzor)
- #8: A *Mare Nostrum IBM JS20 Cluster* (4,800 processzor)

Árak, elérhetőség

A *SUSE Linux Enterprise 10* platform megjelenésével a *Novell* egy egyszerűsített árképzést is bevezet a vállalati ügyfelek számára.

A *SUSE Linux Enterprise Server* előfizetések kétféle formában érhetők el: a mainframe, illetve a nem mainframe szerverekhez. Az előfizetések szerverenként 290 Eurótól érhetőek el, és nem jelent további költséget az esetleges virtuális szerverek futtatása sem.

A *SUSE Linux Enterprise Desktop* használata évente eszközönként 47 Euró, illetve a három éves előfizetés 116 Euróba kerül. Az előfizetéshez 90 napos telepítési támogatást, a szerződés időtartama alatt pedig folyamatos termékfrissítéseket biztosít a *Novell*. *Magyarországon* a *SUSE Linux Enterprise Server* és a *SUSE Linux Enterprise Desktop* a *Novell* viszonteladói és partnerhálózatán keresztül vásárolható meg.

ciós technológiáit. A *YaST* fejlődésének köszönhetően egyszerűsödött a telepítés és konfiguráció. A *SUSE Linux Enterprise Server 10* architektúrája teljesen *CIM*-kompatibilis (*Common Information Model*) ezáltal zökkenőmentesen együttműködik más monitoring és menedzsment-eszközökkel is.

A termékben található *AppArmor* előre meghatározott biztonsági házirend csomagokat kínál népszerű alkalmazásokhoz, valamint olyan varázslókat, amelyek leegyszerűsítik az egyéni biztonsági házirendek létrehozását és telepítését. A beépített jelentéskészítési és figyelmeztető szolgáltatások lehetővé teszik, hogy az informatikai szakemberek gyorsan és könnyen biztosítsák a biztonsági szabályozásoknak való megfelelést, valamint behatolás esetén azonnali értesítést kapjanak. A termék részét képezi számos hálózatdetektáló és -figyelő eszköz, beleértve a nyílt forrású *SNORT* megoldást. Tartalmazza továbbá az *Oracle Cluster File System* 2-t, az *Enterprise Volume Manager Servert*, valamint a *Heartbeat* 2-t, amelyek segítségével nagy megbízhatóságú, hibatűrő fürtözött rendszerek hozhatók létre.

A *SUSE Linux Enterprise Server 10* akár 1024 processzorig és 10 terabájt fizikai memóriáig is skálázható, így kiválthat nagyobb *UNIX*-os installációkat.

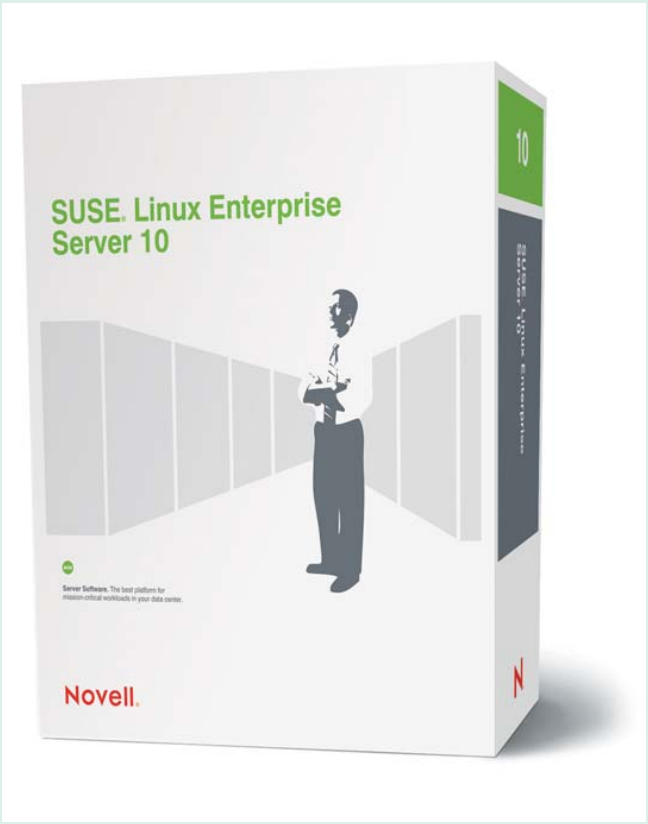
Az adminisztrátorok az eddigieknél könnyebben telepíthetik az új meghajtóprogramokat is.

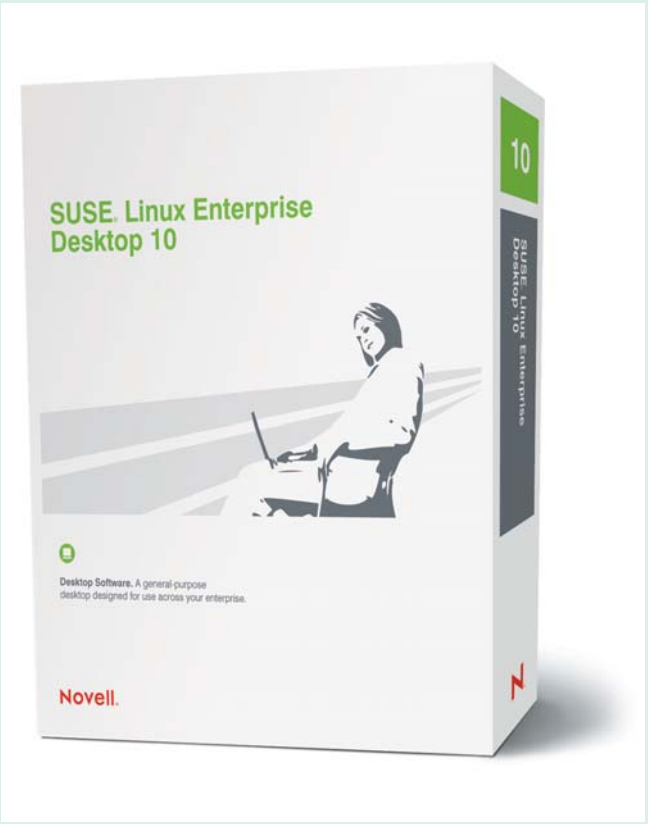
SUSE Linux Enterprise Desktop

A *SUSE Linux Enterprise Desktop* a könnyű használhatóság mellett számos újdonságot is tartalmaz, beleértve az *XGL*-grafikai megoldásokat, a beépített *Beagle* keresőmotort, és az *OpenOffice.org 2.0* irodai csomag *Novell* változatát - a jelenleg piacon lévő operációs rendszerek és irodai programcsomagok árának töredékéért. A *SUSE Linux Enterprise Desktop* teljes asztali megoldásként kiváló alternatívája a jelenlegi asztali operációs rendszereknek, vékony-kliens alkalmazásként és mérnöki munkaállomásként is használható.

Teljes körű ügyfélmegoldás

A *SUSE Linux Enterprise* ügyfelek az előfizetési időszak alatt hozzáférést kapnak a nemrégiben bejelentett *Novell Customer Centerhez*. A *Customer Center* egy központi on-line portál, amely olyan eszközöket és automatizált szolgáltatásokat kínál, melyek használata leegyszerűsíti a licencelést, a frissítések kezelését, és amely teljes mértékben együttműködik a *Novell ZENworks® Linux Managementtel*.





PHP nyomkövetők (2. rész)

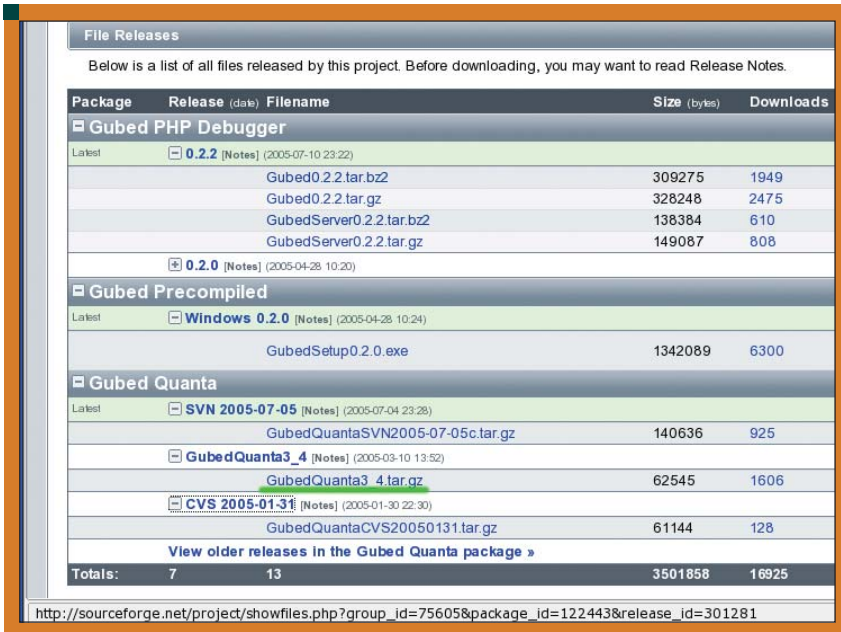
Sorozatunk előző részében végigtekintettünk azon a palettán, amit a Linux alatt elérhető PHP nyomkövetők és profilkészítő alkalmazások jelentenek, és végiggondoltuk, mi indokolhatja ezek használatát. Ebben a részben néhány általános meggondolás után a Quanta Plus fejlesztőkörnyezetről és az ebbe integrált Gubed nyomkövetőről lesz szó.

Általában a webes alkalmazások nyomkövetéséről

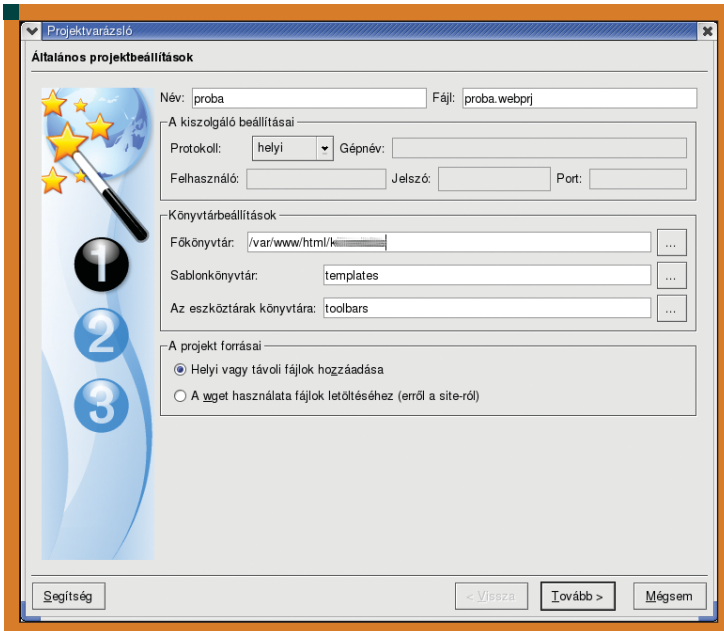
Gondoljuk át, mi történik egy (például *PHP* nyelvű) dinamikus weboldal meglátogatásakor: a felhasználó küld egy *HTTP*-kérést a webszervernek a böngésző segítségével, mire a webszerver feldolgozza a háttérben álló *PHP* szkriptet, és – lehetőség szerint titokban tartva eme szkript szerkezetét – csak a kimenetet küldi vissza a böngészőnek.
Ha viszont nyomkövetni vagy teljesítmény-elemezni szeretnénk egy *PHP* szkriptet, akkor – nem lévén más eszközünk: a kért *URL* szövegében (amikor elküldjük a *HTTP*-kérést a böngészőből) vagy magában a *PHP* kódban – ezt tudomására kell hoznunk a webszervernek. Három megoldást láttam erre:

- a legegyszerűbb az *APD* profilkészítője, melyben „egyszerűen” egy *PHP* parancs jelzi, hogy valami extra igényünk lenne (például `apd_set_pprof_trace();` – ha szeretnénk teljesítményt elemezni). Ennek hatására a *PHP* egy előre kijelölt könyvtárba elmenti a szükséges információkat, mérési adatokat, amit aztán lehet utólag megjeleníteni.
- A másik esetben az oldal *URL*-je *elé* kerül beillesztésre többlet információ (maga a *Gubed* nyomkövető hívódik meg, és a paraméter az oldal URL-je):
`http://szerverneve_gyokerkonyvtar/Gubed/StartSession.php?gbdScript=___` . Előnyeként hirdeti magáról a Gubed, hogy ennek a módszernek köszönhetően nem kell belenyúlni (még modulok révén sem) a futtatandó PHP-be (az-az nem szükséges rendszergazdai jog a lábra állításhoz), és mégis: szinte az összes lényeges funkció megvalósítható.
- Az összes többi változatban az oldal URL-je *mögé* kerülnek be a szükséges információk GET argumentum(ok)ként – legalábbis első alkalommal, mert utána már akár sütiként is tárolódhat a nyomkövetési igény, mint ahogy ez történik például a *Komodo*/*Xdebug* esetében. Tehát úgy fog kinézni a HTTP-kérés, hogy
`http://nyomkovetendo_weboldal?XDEBUG_SESSION_START=...`
vagy
`http://nyomkovetendo_weboldal?DBGSESSID=0@localhost:7869`
stb, nyomkövetőtől függően. Ezt megkapja a webszerver, és ennek megfelelően cselekszik, ha fel van rá készítve megfelelő kiegészítő modul(ok) révén.

Innentől kezdve (a nyomkövetést illetően) eltérő a folytatás. A *Gubed* változatában maguk a *Gubed* szkriptjei tevékenykednek: ők nyitnak meg egy foglalatot (*socket*-et) a nyomkövető kliens (*Quanta* fejlesztőkörnyezet vagy egy *wxWidgets*-es kliensprogram) felé és onnan vissza, és miután kijelzik a vizsgált program sorait, várják a fejlesztő döntését a további lépéseket illetően. S mihelyt végére értünk a vizsgálandó programnak (vagy idejekorán a befejezés mellett döntöttünk), a *PHP* elküldi a böngészőnek a kimenetet.



1. ábra GubedQuanta a SourceForge.net-en



2. ábra Projektvarázsló

A többi nyomkövető a webszerver moduljaként működik, és a webszerver a megfelelő GET argumentumok hatására nyitja meg a *socket*-et a nyomkövető felé. Ezt egy ssh-alagútba is terelhetjük egy terminálablakból kiadott `ssh -R port:localhost:port loginnev@szerverneve` paranccsal, ha távoli webszerverrel dolgozunk. (Pontosabban nem érdemes másként eljárni, mert komoly kockázatot vállalnánk a webszerver biztonságát illetően.) Némi túlzással úgy fogalmazhatok, hogy ezen írás elkészítése közben a számomra legjelentősebb felfedezés az volt, hogy ez az *SSH* alagútba terelés minden nyomkövetőnél működött.

A Gubed/Quanta szerver/kliens bemutatása

Az előbb említett SSH-alagutazást az

```
ssh -R 8016:localhost:8016
loginnev@szerverneve
```

paranccsal tudtam megvalósítani a *Gubed*/*Quanta* esetében, mely az előző részben felsoroltak közül egyedül teljesen szabad nyomkövető/fejlesztőeszköz páros. Ezt a port értéket persze át lehet állítani a *Gubed* szerveren levő (verziótól függően) *localsettings(_dist).php* vagy *GubedGlobals.php* fájlban a `$gbdDebugPort = 8016` részlet megváltoztatásával. A *Quanta* és a *Gubed* lábra állítására és összehangolására jó leírás található a ➲ *www.very-clever.com/quanta-gubed-debugging.php* oldalon. *(Megjegyzés: jelen cikkben a Quanta és a Quanta Plus ugyanazt jelenti.)*

A *Gubed* telepítéséhez kell egy szerver oldali rész, mely több változatban is letölthető, de a leginkább ajánlott a *GubedQuanta* (*1. ábra*). Megtévesztő a letöltési táblázat fejléce, mert ha a *Quantá*t használjuk, akkor a *„Gubed PHP debugger”* részről semmi nem fog kelleni. A *Quantá*ban különös módon az *egyetlen* beépített nyomkövető a *Gubed*. Érdemes a két program (a kliens és a szerver) letöltésekor és telepítésekor komoly figyelmet fordítani arra, hogy a *Quantá*n belüli *Gubed* kliens (fejlesztés alatt álló) protokollverziója megegyezzen a telepítendő *GubedQuanta* protokollverziójával. Több napi keresés után jöttem rá, hogy ez nem volt így esetemben, és ez volt a gyökere sok problémának. A *Quanta* 3.4.2-es verziója együtt tud működni a *GubedQuanta3_4*-gyel. Talán pont emiatt hasonlít a verziószámuk. Ha *Shift+F7*-tel előcsalogatjuk a *Quanta* projektbeállításait, a *Gubed* beállításánál a *Névjegy* fül ezt mutatja: *„A nyomkövetőnek ez a verziója a Gubed protokoll következő verzióját támogatja: 0.0.11.”*

Nagy örömömre a kibontott *GubedQuanta3_4* könyvtárában kiadott `grep -r Protocolversion *` parancs is megmutatta a *0.0.11*-es verziószámot.

A *Quantában*, mint minden rendes fejlesztőkörnyezetben, projekt alapon történik a munkafolyamat. Ennek beállításait kell először végiggondolni (*2. ábra*). Nagy haszonnal használható a *Projekt/„Projektkönyvtár átnézése”* menüpont, ha utólag másoltunk a projektünk könyvtára alá egy-egy könyvtárat, fájlt.

A *Quantában* a *Nyomkövetés* menüpont alatt az alábbiak találhatóak: *Munkafolyamat (elindítása / befejezése),*

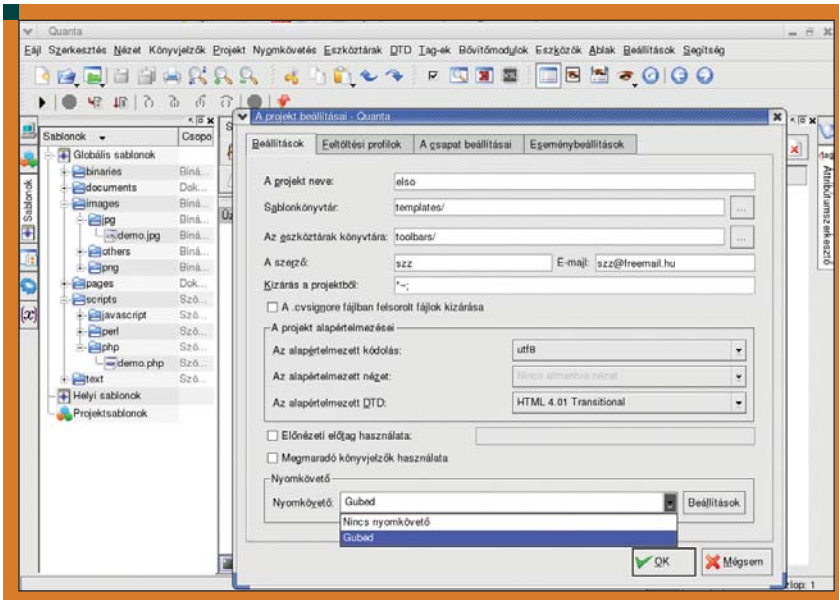
Végrehajtás (HTTP-kérés küldése, Szünet, Futtatás lassan/gyorsan, Lépés, Belépés, Kilépés, Kihagyás, Kilövés), Töréspontok (Feltételes megszakítás, Töréspont átváltása, Töréspontok törlése), Változók (Figyelt változók, Változó értékének beállítása). Ezekből a legtöbb ikonként is látható: egérrel rájuk mutatva részletes magyar nyelvű súgót láthatunk.

A (távoli) webszerveren levő vizsgált *PHP*-fájloknak megtalálhatóaknak kell lenniük a helyi fájlrendszerben is. Ennek sikeréhez először a *Projekt* menüponton belül, a *Projekt tulajdonságai*nál (alul) a *Gubed* mellett jobboldalt, szinte bosszantóan elrejtett *Beállítások* gombot kell megtalálnunk (*3. ábra*), ezen belül pedig a *Könyvtárleképezés* részt. A későbbiekben, az egyes szkriptek beazonosításakor nagy segítségünkre van a *Gubed* beállításánál látható könyvtárleképezési dialógusablak zöld és piros jelzője (*4. ábra*). Ha ez nem sikerül jól, akkor a *Quanta* ezt a dialógusablakot léptennyomon felhozza (például töréspontok kérésekor). Érdemes egy külön helyen, például egy (akár virtuális) „ragadós cetlin" feljegyezni a webszerver és a saját gépünk fájlrendszerének fontosabb útvonalait (s hogy melyik melyik), hogy ne kelljen mindig kézzel beírni (és eltéveszteni) ezeket az elérési utakat.

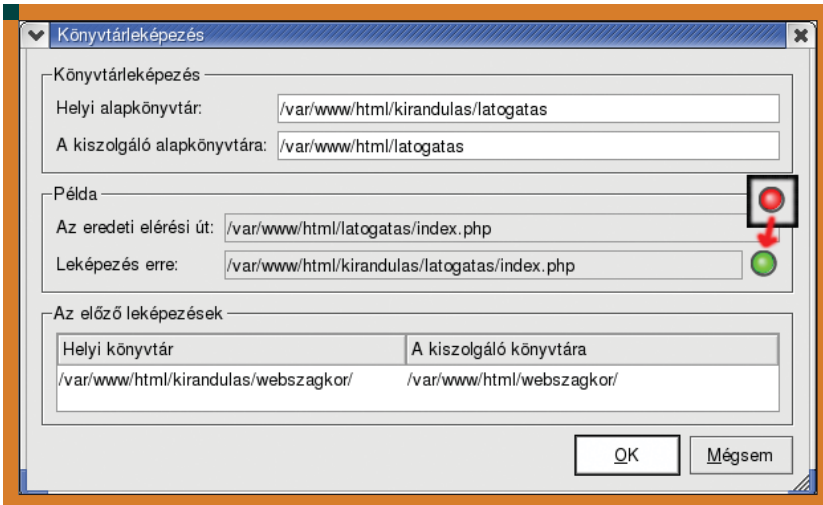
Érdemes tudni, hogy nem lehet jól használni a szimbolikus linkeket a *Quanta* könyvtárleképezésének megadásakor. Tehát azt sajnos nem lehet megtenni, hogy van valahol egy könyvtár, amit láttatni szeretnénk a *Quantával*, de a dokumentumgyökérhez másolás helyett csak egy linket szeretnénk ráállítani. Ilyenkor félrevezetődik a *StartSession.php* (hacsak manuálisan bele nem nyúlunk, de az nagy körültekintést – és az eredeti változat elmentését – igényli).

„HTTP-kérés küldése" után (jól beállított könyvtárleképezés esetén) alul megtekinthető a „Nyomkövetési kimenet" (*5. ábra*), így a *Quanta* más böngésző nélkül, önmagában is használható nyomkövetésre. Rossz beállítások esetén a nyomkövetési kimenetként a *StartSession.php* tartalma, jelenik meg, ami egy űrlap, melynek egyetlen szövegmezőjébe beírható a nyomkövetésre célzott szkript.

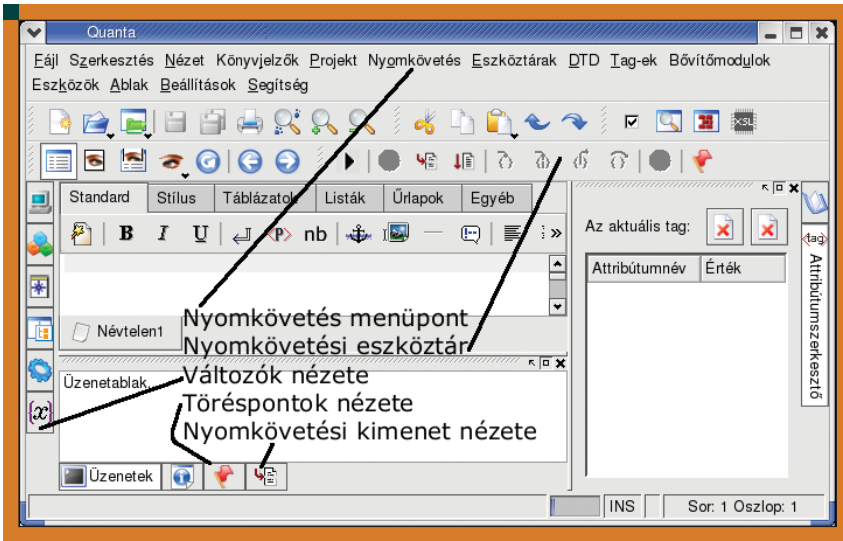
A változók megnézésére van egy



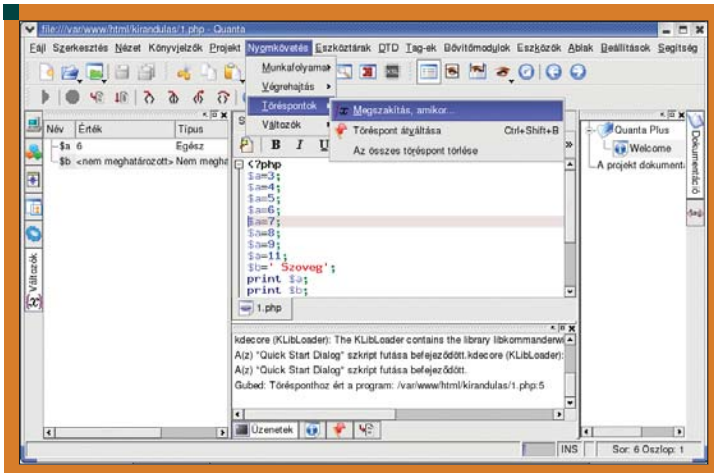
3. ábra Projekt-tulajdonságokon belül, a Gubed beállításai



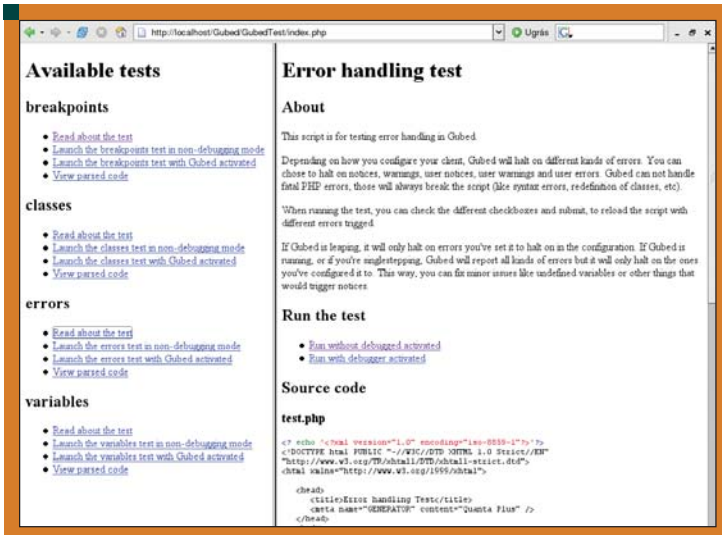
4. ábra Miközben beírjuk a könyvtárleképezést, jelzőlámpa mutatja helyességét



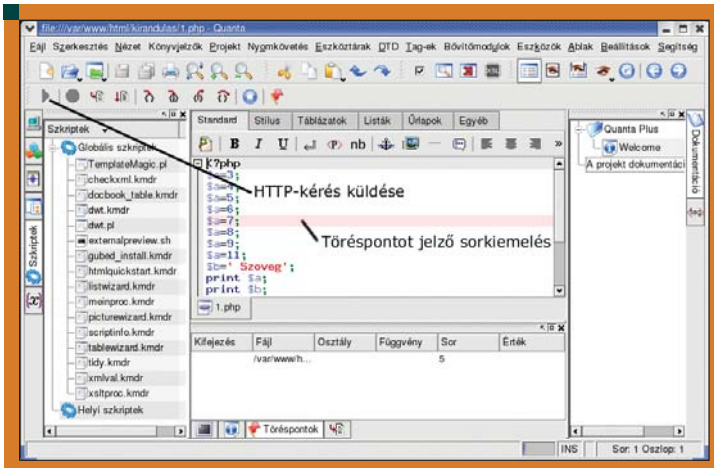
5. ábra Fontosabb nyomkövetési nézetek a Quantában



6. ábra Töréspontok állítása és változók figyelése a Quantában



7. ábra GubedTest



8. ábra Quanta nyomkövetés közben

külön része a fejlesztőkörnyezetnek: {*x*} jelzi (*6. ábra*). Sajnos nem olyan elegáns, hogy ráállunk az egérrel, és már látjuk is. Ezzel szemben a kód szerkesztésekor jó lehetőség, hogy a $ jel lenyomásakor az eddigi változók legördülőmenüje megjelenik. Jól elhelyezett (*Ctrl+Shift+B*) töréspontokkal, melyeket a sorok rózsaszínné válása mutat, érdemi munkára lehet bírni a nyomkövetőt.

A *GubedQuanta*val kapunk egy tesztkörnyezetet is, amit mind a kliens, mind a szerver oldalon tárolnunk kell, hogy a megfelelő könytárleképezés megtörténhessen. A töréspontok, osztályok, hibák és változók megvizsgálásának (és magának a *Gubed*nek) finomságait tudhatjuk meg ezen tesztekből (*7. ábra*).

Ha nem egyezik meg a kliensen és a szerveren tárolt fájl (amit egymáshoz rendeltünk a könyvtárleképezési dialógusablakban), vagy más hiba lép fel, akkor efféle zaj jelenik meg a böngészőben vagy a „Nyomkövetési kimenet”-ben:

```
PHP Bug error: Array ( [0] =>
➥< [1] => ? [2] => = [3] =>
➥Array ( [0] => ...
```

A *Gubed* felhasználói számára *James Humphreys* (*james@mozmarks.com*) készített egy *Firefox*-bővítményt (*gubed.xpi*). Ugyancsak ő volt az elkészítője annak a bővítménynek, ami a ➲ *del.icio.us* előtt szinte az egyetlen lehetőséget kínálta a *Firefox*-könyvjelzők interneten való tárolására. A *gubed.xpi* egy *Gubed* menüpontot hoz létre a *Firefox* menüsorában, és használatával könnyen elérhető, hogy a nyomkövető *URL*-jét beszúrjuk (vagy éppen kivegyük) az aktuális URL elé/elől: *http://localhost(vagy_más_webszerver)/Gubed/StartSession.php?gbdScript=*

A *Quanta Plus* webfejlesztő környezet ügyes eszköz. Ezt hirdeti magáról: „Az a célunk, hogy a *Quanta* legyen a legkönnyebben használható szerkesztőprogram *HTML* fájlok és webes szkriptek készítéséhez”. Van rá esély, hogy ez valóra is váljon a szabad szoftverek fejlődésének erejéből.

Szabó Zoltán
(szz@freemail.hu)
Négy gyermekével és feleségével Pannonhalmán él. Tíz éve kísérletezik a Linuxszal. Matematikát és informatikát tanít, diákotthonban keseríti a rábízottak életét. Szívügye a PHP, a PostgreSQL és a Moodle.

Grafikus programok készítése GTK+-szal

Az egyik legmodernebb grafikus eszközkészlet (toolkit) a GTK+ nevű könyvtár, amelyet eredetileg a jól ismert Gimp rajzolóprogrammal együtt kezdtek el kifejleszteni. A neve, „The Gimp Toolkit”, is innen jön. Ma már rengeteg alkalmazás használja, és a Gnome is teljesen erre épül. Ebben a cikkben ennek a programozását mutatom be, részletezve a kezelésének alapjait, az eseményvezérelt programozás módját, és a konténerek használatát.

Bevezetés

Toolkitnek nevezzük azt a programkönyvtárat, amellyel egyszerűen rajzolhatunk felhasználói elemeket (***widget***), gombokat, menüket, gördítősávokat a programunkban. Az ***Xwindow*** rendszernél, a ***Windowszal*** ellentétben, ez nem része az alaprendszernek. Nem érdemes nekiállnunk az ***X*** rendszert ***toolkit*** nélkül, alacsony szinten programoznunk, ilyenkor ugyanis minden grafikát a legegyszerűbb elemekből, pontokból, vonalakból kell felépítenünk, és egyesével feldolgoznunk a szervertől jövő üzeneteket – egérmozdulatokat, kattintásokat, billentyű lenyomásokat. Az említett ***toolkitek*** többé-kevésbé leveszik a vállunkról ezt a feladatot. Az elavultabbakkal szerencsére a mai felhasználók már nem is találkoznak. Ezek meglehetősen csúnyák és nehézkesek voltak. Egyszerűbb programoknál az ***Xaw***, összetettebbeknél a ***Motif*** volt használatos, amely ráadásul nem is szabad szoftver. Manapság a két leggyakrabban alkalmazott ***toolkit a KDE-ben*** használt ***Qt*** és a ***Gnome*** alapját képező ***GTK+***.

Ha szeretnénk egy adott rendszer alá programot fejleszteni, érdemes megtanulnunk az abban megszokott ***toolkit*** használatát. Léteznek megoldások az egyes készletek hordozhatóvá tételére, de ezek mind a teljesítmény rovására mennek. Valljuk be őszintén, hogy például az ***OpenOffice.org*** vagy a ***Mozilla Firefox*** is jóval lassabbak, mint a ***Windowson*** futó natív társaik, ráadásul nem is illeszkednek teljesen egyik rendszerbe sem.

Helló, világ!

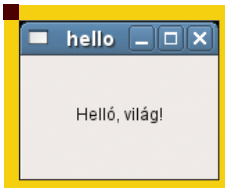
Először készítsünk egy programot, amely egy ablakot jelenít meg a képernyőn, benne a „Helló, világ!” felirattal! Gépeljük be az első listán látható forráskódot. Figyeljünk arra, hogy a szöveget UTF-8 kódolással mentsük el, a ***GTK+*** ugyanis belül mindig ezzel dolgozik. Ha a fájl neve ***hello.c***, akkor ezzel a ***Makefile*** részlettel fordítható:

```
hello: hello.c
    gcc $< -o $@ `pkg-config
    ➥gtk+-2.0 -cflags -libs`
```

Nézzük sorról sorra, mit csinál a program! Először is meghívjuk a `gtk_init()` függvényt. Ez átnézi a programunk argumentumlistáját, és amely paramétereket értelmezni tud (például `--display`), azt el is távolítja a listából, így a programunknak nem kell azokat figyelmen kívül hagynia. Ezután létrehozunk egy ablakot. A `gtk_window_new()` visszatérési értékét elmentjük egy változóba, hogy később hivatkozni tudjunk erre az ablakra. A következő sorban létrehozunk egy címkét (a `GtkLabel` olyan elem, amely csak egy szöveget ír ki). A `gtk_container_add()` függvénnyel el is helyezzük az ablakunkban. Ennek első paramétere az ablak, a második pedig a felületi elem, amelyet belerakjuk: itt az újonnan készített címke. Az első paraméter `GtkContainer*` típusú kell legyen, ezért a `GTK_CONTAINER()` makróval konvertáljuk. (Erre nincs feltétlenül szükség, csak a fordító figyelmeztető üzenetét kerüljük el vele. Írhattuk volna azt is, hogy `(GtkContainer *)window`. A makró azonban típusellenőrzést is végez.) Az ezt követő sorban a szöveg

1. Lista: Helló, világ GTK+ módra

```
#include <gtk/gtk.h>

int
main (int argc, char *argv[])
{
    GtkWidget *window;

    gtk_init (&argc, &argv);

    window = gtk_window_new
➥(GTK_WINDOW_TOPLEVEL);
    gtk_container_add
➥(GTK_CONTAINER (window),
➥gtk_label_new ("Helló,
➥világ!"));
    gtk_container_set_border_
➥width (GTK_CONTAINER
➥(window), 36);

    gtk_widget_show_all
➥(window);
    gtk_main ();
    return 0;
}
```



1. ábra GTK+ Helló, világ!

körüli keretet 36 képpontra állítjuk. A kód utolsó részében megjelenítjük az ablakot, és az azt tartalmazó összes elemet a `gtk_widget_show_all()` függvénnyel. A `gtk_label_new()` függvény visszatérési értékét, amely egy `GtkLabel*` típusú mutató, ezért nem kellett elmentenünk külön változóba. Mi később már nem hivatkozunk rá; a *GTK+* pedig számon tartja az ablak tartalmát. A függvény hatására az egész tartalmat bejárja, minden elemet, láthatóvá téve. Végül átadjuk a vezérlést a `gtk_main()` hívással.
Fordítsuk le a programot (`make`) és indítsuk el (`./hello`)! Megjelenik az 1. ábrán látható ablak. Az első programunk nem tartalmaz aktív elemeket, így nem tudunk sehogyan kilépni belőle. Észrevehetjük azt is, hogy ha az ablak jobb felső sarkában az X-re kattintunk, akkor is csak eltűnik, de a programunk fut tovább. (A terminál ablakban Crtl-C gombbal tudjuk megállítani.)

Aktív elemek, események, szignálok

A *GTK+* használatához meg kell értenünk néhány fogalmat.
Egy grafikus rendszerre írt program alapvetően más felépítésű, mint a konzolos társa. Az ilyet ugyanis ***eseményvezérelt*** (event-driven) módon kell megírnunk. A programnak a bejövő eseményeket kell feldolgoznia, és ha végzett, várni a továbbiakra. Az események lehetnek sokfélék, például lejár egy időzítő, a felhasználó megmozdítja az egeret, egy gombra kattint, vagy adat érkezett a hálózati kapcsolaton. A *GTK+* támogatást nyújt ilyen programok készítéséhez. A beérkező eseményeket maga kezeli: összegyűjti őket az úgynevezett ***event pool***-ban, és a megfelelő sorrendben végrehajtja az általunk hozzájuk rendelt programrészleteket.
Számunkra két fajta esemény fontos, amelyek között apró különbség van csak. ***Event*** a neve annak az eseménynek, amelyet az X szerver továbbít az alkalmazásunk felé. Ilyen lehet egy billentyű megnyomása, vagy egy kattintás valahol. A ***szignálok*** pedig általában már az ***eventeket*** értelmezik. Emiatt gyakran mindkettőre pongyolán szignálként szoktak hivatkozni. Ezek valamilyen objektum működésével kapcsolatos jelzések. Például egy gördítősávot mozgathatunk fogd és vidd módszerrel. Rákattintunk a csúszkára (***button-press-event***), arrébb húzzuk (az X szerver akár képpontonként küld egy ***motion-eventet***), és végül elengedjük (***button-release-event***). A felhasználói elemek szerencsére magasabb szintű szignálokat biztosítanak, például egy *GTK+* gördítősáv a ***value-changed*** jelzést küldi nekünk (emit), ha az értéke bármilyen módon megváltozott. Szignált mi magunk is indíthatunk. Egyes esetekben erre is szükség lehet: ha egy gördítősávot elmozdítunk a programból, hogy felhívjuk valamire a felhasználó figyelmét.
Egészítsük ki a programunkat, hogy kezelje az eseményeket (*2. lista*)!
A program hasonló az előzőhöz, viszont most a címke helyett egy gombot hozunk létre „Klikk ide!” felirattal, és ezt helyezzük el az ablakban. Ha a felhasználó erre kattint, akkor ez az elem kibocsájt egy *„**clicked**”* szignált, amelyhez mi a `klikk_cb()` függvényt kapcsoljuk. Akárhányszor a felhasználó a gombra kattint (vagy a szóköz, enter billentyűk valamelyikét megnyomja), a *GTK+* elindítja nekünk ezt a kezelőt (***callback***). A kezelő tartalmaz egy értékét megtartó (***static***) változót, amelyik számolja, hogy hányszor aktiváltuk a gombot, és ennek alapján megváltoztatja a feliratát. (A gomb az ablakhoz hasonlóan egy konténer elem, amely egy további elemet tartalmazhat.
A `gtk_button_new_with_label()` függvény tulajdonképpen egy kényelmi funkció, amely létrehoz egy gombot, és benne elhelyez egy adott címkét is.) A kezelőnek nincsen visszatérési értéke (***void***).
Az ablak *„**destroy**”* szignáljához is kapcsolunk egy kezelőt, ez a `gtk_main_quit()`. Ez a függvény kilép a programból, hatására a vezérlés a `gtk_main()` utáni sornál folytatódik. Amikor az ablak jobb felső sarkán az x-re kattintunk, vagy ***Alt-F4***-gyel bezárjuk azt, a szignál akkor jön létre. Látható, hogy nem feltétlenül kell minden szignálhoz kezelőt írnunk, használhatunk egyet a néhány előre definiált közül is.
Nézzük meg kicsit pontosabban, mi is történik az ablak bezárásakor!

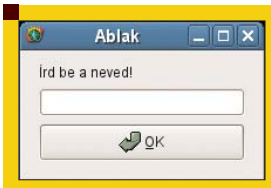
2. Lista: Eseményvezérelt Helló, világ

```
#include <gtk/gtk.h>

void
klikk_cb (GtkWidget * widget,
➥gpointer data)
{
   static int klikk = 0;
   char *text;

   text = g_strdup_printf
➥("Kattintások száma: %d",
➥++klikk);
   gtk_button_set_label
➥(GTK_BUTTON (widget),
➥text);
   g_free (text);
}

int
main (int argc, char *argv[])
{
   GtkWidget *window,
➥*button;

   gtk_init (&argc, &argv);

   window = gtk_window_new
➥(GTK_WINDOW_TOPLEVEL);
   gtk_container_set_border_
➥width (GTK_CONTAINER
➥(window), 36);
   g_signal_connect (G_OBJECT
➥(window), "destroy",
➥G_CALLBACK (gtk_main_quit),
➥NULL);

   button =
➥gtk_button_new_with_label
➥("Klikk ide!");
   gtk_container_add
➥(GTK_CONTAINER (window),
➥button);
   g_signal_connect (G_OBJECT
➥(button), "clicked",
➥G_CALLBACK (klikk_cb),
➥NULL);

   gtk_widget_show_all
➥(window);
   gtk_main ();
   return 0;
}
```

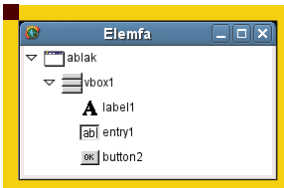
3. Lista A delete-event kezelője

```
gboolean
delete_event (GtkWidget *
➥widget, GdkEvent * event,
➥gpointer delete)
{
   GtkWidget *dialog =
➥gtk_message_dialog_new
➥(GTK_WINDOW (widget), 0,
      GTK_MESSAGE_QUESTION,
➥GTK_BUTTONS_YES_NO,
➥"Biztosan kilépsz?");
   int result;

   result = gtk_dialog_run
➥(GTK_DIALOG (dialog));
   gtk_widget_destroy
➥(dialog);
   if (result ==
➥GTK_RESPONSE_YES)
      return FALSE;

   return TRUE;
}
. . . . .
   gtk_container_set_border_
➥width (GTK_CONTAINER
➥(window), 36);
   g_signal_connect (G_OBJECT
➥(window), "delete-event",
➥G_CALLBACK (delete_event),
➥NULL);
   g_signal_connect (G_OBJECT
➥(window), "destroy",
➥G_CALLBACK (gtk_main_quit),
➥NULL);
```

Az X rendszer küld az alkalmazásunknak egy *„delete-event”* jelzést. Mivel ehhez mi nem adtunk meg kezelőt, a *GTK+* átadja az eredeti kezelőnek a vezérlést, amely bezárja az ablakot, az összes benne lévő elemmel együtt, és felszabadítja a hozzájuk tartozó erőforrásokat. A bezáráskor létrejön a *„destroy”* szignál, amelyhez viszont már definiáltunk egy kezelőt, amely pedig kilép a programból.
Az *eventek* kezelői, a szignálokéitól eltérően rendelkeznek visszatérési értékkel, mégpedig egy *gboolean* típusúval. Ha írunk egy ilyen függvényt, ezzel adhatjuk meg, hogy elvégeztük-e az eseményhez tartozó műveleteket, vagy nem. Ez nagyon hasznos dolog. Például készíthetünk a *„delete-event”* jelzéshez egy kezelőt, amelynek a hatására felugrik egy egyszerű dialógus, benne egy kérdéssel: „Biztos ki akarsz lépni a programból?”, alatta pedig Igen-Nem gombokkal. Ha a felhasználó a nemre kattint, akkor *TRUE* értékkel térünk vissza a kezelőből, jelezvén a *GTK+*-nak, hogy nincs további teendő. Ha pedig az igenre, akkor *FALSE* értékkel, aminek hatására a vezérlés az alapértelmezés szerinti kezelőre adódik tovább (*propagate the event further*), amely bezárja az ablakot. Az ablak bezárása pedig egy *„destroy”* szignált generál, amelyik kilép a programból.
A program kiegészítését lásd a *3. listán*. Ebben a `gtk_message_dialog_new()` egy olyan kényelmi funkció, amelyik rengeteg feladatot elvégez: létrehoz egy ablakot, rajzot, gombokat, szöveget tesz bele helyettünk.
A `gtk_dialog_run()` függvény visszatérési értéke adja meg, hogy melyik gombra kattintott a felhasználó. A dialógus ablak bezárása után döntjük el, hogy az alkalmazás ablakát engedjük-e bezárni az eredeti kezelőnek (`return FALSE`), vagy sem (`return TRUE`).



2. ábra A GTK+ konténerek



3. ábra Az ablak elemfája

A GTK+ tárolóelemek

A *GTK+-ban* vannak olyan elemek, amelyek továbbiakat tartalmazhatnak. Ilyen például a fenti gomb, amely egy címke elemet foglal magába. Lehetséges az is, hogy egy ikont és mellette egy szöveget mutat. Az ilyen elemeket nevezzük konténereknek. Aki készített már felhasználói felületeket *Windowsban*, tudja, hogy ott minden elemnek meg kell határoznunk a koordinátáját és a méretét, ahol az ablakon belül látszik. Ezzel ellentétben itt minden elem helyét és méretét az jelöli ki, hogy milyen konténerben helyeztük el és hogyan. Az elhelyezést akár egy fa gráfban, az elemfában le is rajzolhatjuk.
A konténer általános fogalom, rengeteg fajtája van. Vannak konténerek, amelyek egyetlen elemet tartalmaznak. Ilyen például egy ablak vagy egy gomb. Vannak olyanok is, amelyeknek több gyermekük (*child*) lehet, és azok között a rendelkezésre álló helyet az általunk meghatározott módon elosztják. Ilyenek a dobozok (vízszintes és függőleges, *GtkHBox*, *GtkVBox*) és a táblázat (*GtkTable*). A konténereket tetszés szerint egymásba ágyazhatjuk. A koncepciót a *2.* és *3. ábra* mutatja be. A legfelső szintű elem az ablak. Az tartalmaz egy függőleges dobozt, amely három részre osztja a területét. Az első elem egy címke, a középső egy szövegbeviteli doboz, a legalsó pedig a gomb. Az utóbbi is egyébként különálló gyermekekként foglalja magába a képet és az *OK* feliratot.
A *GTK+-ban* a felhasználói elemeket a `gtk_button_new()` és hasonló függvényekkel hozzuk létre, aztán egy konténerbe csomagoljuk (pack) őket. Léteznek kényelmi függvények is, amelyek több elemet hoznak létre egyszerre. Ilyen a már fent bemutatott `gtk_button_new_with_label()` is. Ez létrehoz egy gombot és egy címkét, majd a címkét elhelyezi a gomb (mint konténer) belsejében.
Több elem is van, amely képes konténerként működni. A *GTK+-ban* az egyes elemeket egymásból származtatják. Sok helyen ezért van szükség a típuskonverziós makrókra: a `gtk_container_add()` függvény, ahogy fent is láttuk, `GtkContainer*` típusú első argumentumot vár, de mi adhatunk neki egy gombot vagy egy ablakot is. Ezeket mind a `GtkContainer` típus leszármazottjai.
Módosítsuk programunkat úgy, hogy több gombot tartalmazzon (*4. lista*)!
A program sokban hasonlít az előzőre. A gombok létrehozása előtt kérünk egy *GtkVBox*-ot, amely az ablak belsejét függőlegesen több részre osztja. Ezt a dobozt tesszük az ablakba, majd a gombokat pedig egymás után ebbe a dobozba.
Létrehozunk egy egész számokat tároló `klikk[]` tömböt, ebben tároljuk majd, melyik gombunkra hányszor kattintott a felhasználó. Egy ciklussal létrehozzuk a tíz gombot, és egymás alá csomagoljuk őket a dobozba

4. Lista: Elemek egymásba ágyazása

```
#include <gtk/gtk.h>

void
klikk_cb (GtkWidget *widget,
int *data)
{
  char *text = g_strdup_printf
("Kattintások száma: %d",
++(*data));
  gtk_button_set_label
(GTK_BUTTON (widget), text);
  g_free (text);
}

int
main (int argc, char *argv[])
{
  GtkWidget *window, *button,
*vbox;
  const int gombok = 10;
  int klikk[gombok];
  int i;

  gtk_init (&argc, &argv);

  window = gtk_window_new
(GTK_WINDOW_TOPLEVEL);
  g_signal_connect (G_OBJECT
(window), "destroy",
G_CALLBACK (gtk_main_quit),
NULL);

  vbox = gtk_vbox_new (FALSE,
6);
  gtk_container_add
(GTK_CONTAINER (window),
vbox);
  for (i = 0; i < gombok; i++)
{
     klikk[i] = 0;
     button =
gtk_button_new_with_label
("Ide még nem
kattintottak!");
     g_signal_connect
(G_OBJECT (button),
"clicked", G_CALLBACK
(klikk_cb), klikk + i);

gtk_box_pack_start_defaults
(GTK_BOX (vbox), button);
  }

  gtk_widget_show_all
(window);
  gtk_main ();
  return 0;
}
```

(`gtk_box_pack_start_defaults()`). A gombok mindegyike ugyanúgy működik, mint az előző programban: a feliratán számolja az eddigi kattintásokat. A *GTK+* lehetőséget ad arra, hogy több különböző elemhez ugyanazt a kezelő függvényt kössük, az ugyanis megkapja az elemre mutató pointert (`GtkWidget *widget`). A `g_signal_connect()` függvény utolsó paraméterével minden kezelőhöz társíthatunk egy további mutatót is, amelyet saját célunkra használhatunk. Itt ez a `klikk[]` tömb arra elemére mutat, amelyben az adott gomb aktiválásait számoljuk; ez a `klikk+i` mutató. Ezt a mutatót a `klikk_cb()` függvény deklarációban bármilyen típusúnak definiálhatjuk, ugyanis a mutatók mérete mindig egyforma. Figyeljük meg, hogy ebben a programban sem használtunk egyetlen globális változót sem!

Ezen a programon a konténerek működését is kipróbálhatjuk. Futás közben az ablakot szabadon átméretezhetjük; a *GTK+* ekkor automatikusan újraszámolja a gombok méretét, és kirajzolja őket.

Tanácsok a fejlesztéshez

A ➲ *http://gtk.org/* oldalon megtaláljuk a *GTK+* dokumentációját, illetve egy tankönyvet is a ➲ *http://gtk.org/tutorial/* címen. Ez ehhez a cikkhez hasonló bevezetést nyújt (angolul). Itt az összes függvény leírását megtaláljuk; a dokumentációt érdemes a saját gépünkre is telepíteni (pl. *gtk-doc* csomag). A *Gnome* rendszer tartalmaz egy *Devhelp* nevű programot, amely a telepített dokumentációk böngészését teszi lehetővé.

A *GTK+-hoz* jár egy példaprogram gyűjtemény is. Ezt több terjesztésben a *gtk-examples* csomag tartalmazza, és a `gtk-demo` paranccsal indítható. Ez sok elem kezeléséhez mutat példákat, és egyben a forráskódot is megnézhetjük. Én magam is jobban szeretem, ha kezembe adják a példaprogramokat, amelyek alapján felépíthetem a saját alkalmazásomat. A *GTK+* viszont kitűnő minta arra, hogyan kellene felépíteni a programjainkat. Az egész könyvtár egy meglepően konzisztens, átlátható rendszert alkot; ha az ember kezd belejönni a használatába, már szinte automatikusan kitalálja a függvények neveit, amelyekkel az egyes elemeket kezelni lehet. Például egy beviteli dobozból a szöveget a `gtk_entry_get_text()` függvénnyel kérdezhetjük le, egy ablak címkéjét pedig a `gtk_window_set_title()` függvénnyel állíthatjuk be. A *Devhelp* programban az egyes függvényeket és szignálokat áttekintve percek alatt képet kaphatunk egy adott elem működéséről. Ha valamilyen műveletet nem találunk, amelyről úgy gondoljuk, kellene lennie, általában valamelyik szülő objektum rendelkezik olyannal: nincsen `gtk_window_show()`, de van `gtk_widget_show()`. Az utóbbi viszont bármilyen elemre alkalmazható.

Ha összetettebb felhasználói felületeket készítünk, akkor érdemesebb az egyes ablakokhoz létrehozni egy struktúrát, amelyben tároljuk a benne létrehozott elemeket (pontosabban azok `GtkWidget*` mutató reprezentációit). Ennek több előnye is van, azon kívül, hogy áttekinthetőbbé teszi a programunkat. Egy adott ablakból így könnyen több példányt készíthetünk. Gyakran előfordul az is, hogy a szignálok `user_data` paraméterében egy felületi elemet adunk át; az egész struktúrára mutató *pointer* átadásával minden elemre tud hivatkozni a kezelő.

Bonyolultabb felhasználói felületek létrehozásához mindenképpen érdemes a *Glade* programot kipróbálnunk, amely a ➲ *http://glade.gnome.org/* címen található meg. Ez a megtervezett ablakokat XML fájlokban is el tudja menteni, hogy futás közben töltsük be őket, de képes *C* forráskódot is generálni, amelyhez már csak a kezelőket kell megírnunk. Az áttekinthető felületek készítéséhez a *GNOME Human Interface Guidelines* dokumentum ad segítséget.

Ez a ➲ *http://developer.gnome.org/projects/gup/hig/2.0/* helyen található.

Czirkos Zoltán

Jelenleg diplomatervező a Budapesti Műszaki Egyetem Elektronikus Eszközök Tanszékén. Kutatási területe az operációs rendszerek betörésvédelme és a P2P kommunikáció. 2005-ben a Tudományos Diákköri Konferencián II. helyezést ért el. Kedvencei a boszorkányos és a rózsaszín párducos filmek.

Wine

Egy korty Windows

Minden bizonnyal a Microsoft operációs rendszerére több alkalmazás létezik, mint a szabadszoftveres közösség éllovasára, a GNU/Linuxra. Ennek egyszerű az oka, a Windows elterjedtebb. Természetesen számtalan hasznos és kevésbe hasznos program létezik Linux alá is, de néha kedvünk szottyanhat kifejezetten windowsos programokat futtatni. Ebben lehet a segítségünkre a Wine.

Háttértörténet

Essen néhány szó a projekt kezdetéről. 1993-ban ***Bob Amstadt*** irányításával elindult egy fejlesztés, mely azt a célt tűzte ki maga elé, hogy ***Windows 3.1*** alkalmazásokat lehessen futtatni ***Linux*** alatt. Ezt követően hamarosan ***Alexandre Julliard*** vette át a stafétabotot, aki a mai napig is a fejlesztői gárda élén áll.
Az évek során a ***Linux*** mellett egyéb ***Unix*** rendszerekre is elérhetővé vált a program, mely időközben a ***Win32*** alkalmazások futtatására is képes lett.

A ***Wine*** még mindig ***in development***, azaz fejlesztési stádiumban van, ez azonban nem zavarja a szabad közösség lelkes felhasználóit abban, hogy kedvükre használhassák ezt a kiváló dolgot. Bővebb információ található a ➲ http://winehq.org/site/history oldalon.

A Wine

A ***Wine***-t ➲ http://winehq.org sokan tévesen ***Windows***-emulátornak gondolják, holott ez az alkalmazás egy ***wrapper***, ami futtatási környezetet jelent. Még a rövidítése is a ***Wine In Not an Emulator*** (a ***Wine*** nem egy emulátor) szavakból jött. Tehát magát a ***Windowst*** nem emulálja, hanem elősegíti a w***indows***os alkalmazásnak futását. Ez annyit jelent, hogy megpróbálja (több-kevesebb sikerrel) helyesen megjeleníteni azokat a képernyőn, illetve a rendeltetésszerű működésüket biztosítja.
Felmerülhet a kérdés, hogy miért is van szükség arra, hogy w***indows***os programokat futtassunk ***Linux*** alatt. A válasz rendkívül egyszerű. Vagy nincs az adott alkalmazásnak ***linux***os megfelelője, vagy az nem éri el a ***windows***os alkalmazás szintjét.
A ***Wine*** jelenlegi verziója a ***0.9.16***, amely letölthető a ➲ ***http://winehq.org/site/download*** oldalról.
A forráskód mellett a számos ***Linux*** terjesztés bináris csomagként is tartalmazza. Ezen felül a következő ***Unix*** rendszerekre: ***FreeBSD 5.3 és feljebb, PC-BSD, Solaris (9, 10)***. Mulatságos módon ***Windows***ra is található bináris ***Wine*** csomag ***zip***-pel csomagolva...

Telepítés

Manapság már szinte mindegyik (elterjedt) ***Linux*** disztribúció szállítja a ***Wine***-t, azaz tartalmazza. Ezen ***Wine*** frissessége az adott disztribúció frissességétől függ. Tehát ha a legfrissebb ***Linux*** terjesztést használjuk, akkor nincs szükségünk telepítésre. Azonban előfordulhat, hogy egy régebbi disztribúciót használunk, amelyben a ***Wine*** már elavult. Ebben az esetben frissítenünk kell a programot, azaz újra kell telepítenünk. Ez az előző verzió eltávolításával kezdődik. Például ***Mandriva Linux*** alatt a ***Mandrake Control Center*** szoftvereltávolítás funkciójával megtehetjük ezt. ***Debian GNU/Linux*** alatt pedig az

```
apt-get remove wine
```

parancsot kell kiadnunk.

Ezután letöltjük a legfrissebb csomagot az előbb említett oldalról, vagy disztribúciónk csomagkezelőjével tesszük ezt meg. Amennyiben ***dpkg***-alapú terjesztésünk van (például ***Debian*** vagy ***UHU-Linux***), akkor a

```
dpkg -i wine_verzio.deb
```

vagy a

```
dpkg -i wine_verzio.uhu
```

paranccsal telepíthetjük.
Ha ***rpm***-alapú terjesztést (***SuSE***, ***Mandriva***, stb.) használunk, akkor az

```
rpm -ivh wine_verzio.rpm
```

paranccsal varázsolhatjuk a ***Wine***-t a gépünkre.
Slackware és egyéb ***tgz***-alapú disztribúciókban az

```
installpkg wine_verzio.tgz
```

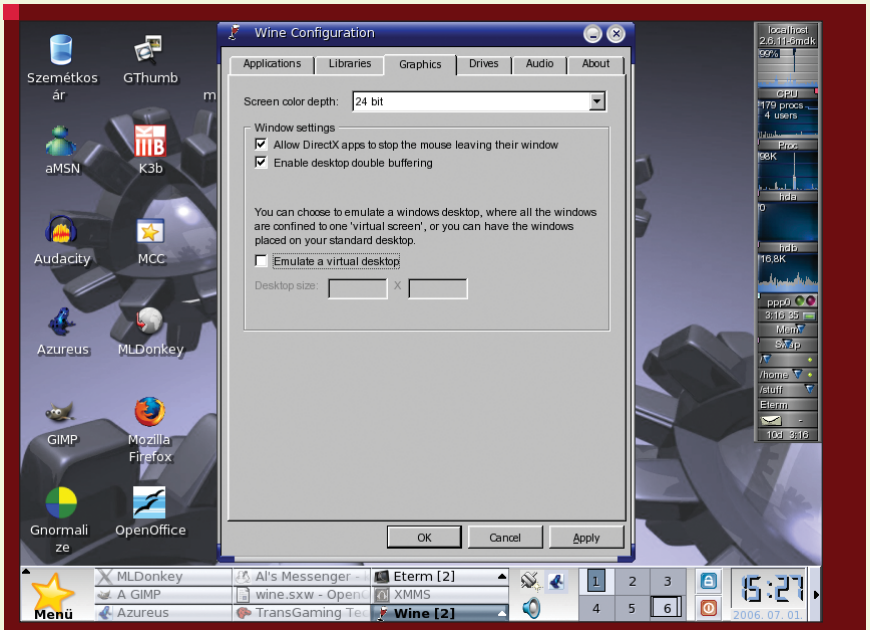
paranccsal telepíthető a program.
Amennyiben teljesen ***hardcore*** felhasználók vagyunk, telepíthetjük a ***Wine***-t forrásból is. Ez különösen akkor hasznos, ha speciális opciókat szeretnénk megadni a fordítás során, amelyek a `README` vagy az `INSTALL` állományokban olvashatóak.
Amint kicsomagoltuk a tömörített állományt a

```
tar xvjf wine_verzio.tar.bz2
```

paranccsal és beléptünk a keletkezett könyvtárba, adjuk ki sorrendben a

```
$ ./configure
# make depend
# make
# make install
```

parancsokat.
Ezen felül telepíthetjük ***CVS***-ből is az éppen aktuális ***snapshot***-ot, így naprakészek lehetünk. Erről bővebben a ➲ http://winehq.org/site/cvs oldalon találhatunk információt. Természetesen ezekre a parancsokra csak végső esetben lehet szükségünk, mivel az elterjedt és felhasználóbarát disztribúciók mind rendelkeznek grafikus ***csomagtelepítő***vel, illetve ***Vezérlőpult***tal, amelyekben pár kattintással elvégezhetjük a telepítést, illetve a frissítést.



1. ábra Akár így is konfigurálhatjuk a Wine-t

Konfigurálás

Ha a ***Wine*** feltelepült és először elindítjuk az opciók és argumentumok nélküli

```
wine
```

paranccsal, létre fog hozni egy rejtett könyvtárat a ***home*** könyvtárunkban. Ennek az elérési útja a `$HOME/.wine`. Célszerű nem elfelejteni, hogy a ponttal kezdődő könyvtárnevek és fájlnevek rejtett könyvtárakat, illetve fájlokat takarnak. Továbbá -ha még nincs beállítva- érdemes a fájlkezelőnkben beállítani a rejtett fájlok és könyvtárak megjelenítése opciót.
Amennyiben ***Midnight Commander***-t (***mc***) használunk, ezt a ***Beállítások/Alapbeállítások*** menüben tehetjük meg.
Az imént létrejött rejtett könyvtárban található a ***Wine*** alapbeállításait tartalmazó konfigfájl is (`$HOME/.wine/config`). Most ezzel fogunk foglalkozni egy kicsit, tudniillik ez a lelke mindennek.
Grafikus beállításra is van lehetőségünk, ha kiadjuk a

```
winecfg
```

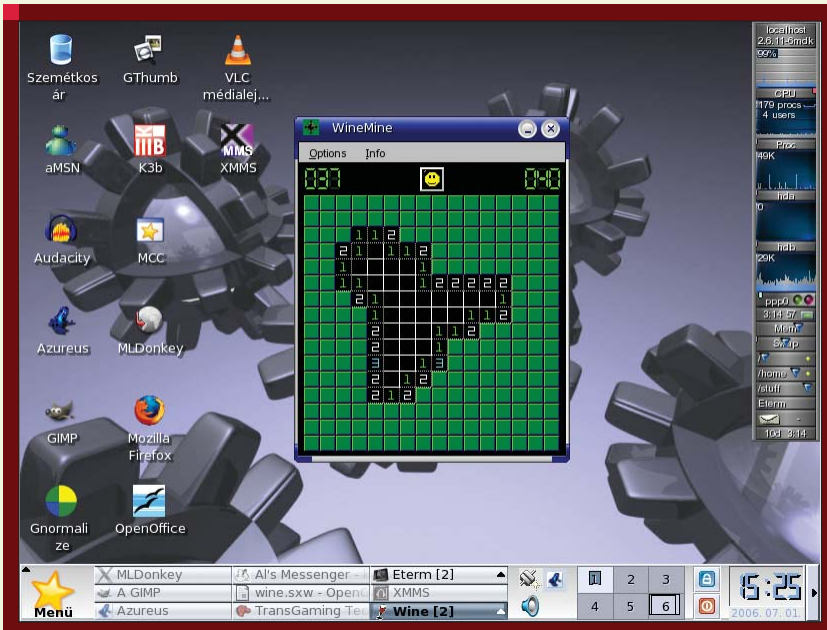
parancsot.
Nos, akkor lássuk apránként, hogy a különböző bejegyzések mire is szolgálnak. Fontos, hogy ebben az állományban a ***komment*** jele nem a megszokott kettőskereszt (#), hanem a pontosvessző (;). Természetesen nincs elég hely arra, hogy minden apró részletre kitérjek, de a legfontosabbakat megpróbálom kiemelni.

Meghajtók

Lássunk erre egy példát:

```
[Drive A]
"Path" = "/mnt/floppy"
"Type" = "floppy"
"Label" = "Floppy"
"Device" = "/dev/fd0"

[Drive C]
"Path" = "/home/killall/
➥.wine/drive_c"
"Type" = "hd"
"Label" = "/home/killall/
➥.wine/drive_c"
"Filesystem" = "win95"
```

A beállításokat tartalmazó fájlban `[drive A]`-tól `[drive Z]`-ig szereplő bejegyzések. Rögtön láthatjuk, hogy A-Z-ig bármilyen betűt használhatunk meghajtónévnek. Persze értelemszerűen érdemes betartani a szabályokat és a meghajtók hierarchiáját. Értem ezalatt, hogy A: ***floppy-meghajtó***, C: ***merevlemez***, és így tovább.
Láthatjuk, hogy logikusan van felépítve a konfig állomány: először a csatlakoztatási pontot kell megadni (`/mnt/csatlakoztatasi_pont`), utána a meghajtó típusát, ami lehet `hd` (***hard***



2. ábra Aknakeresőnk Wine alatt is alapból van

disk, azaz merevlemez), `floppy` (floppy-meghajtó), `cdrom` (*CD* vagy *DVD*) és `network` (távoli meghajtó). Ezek után `Label`-nek, azaz címkének bármit megadhatunk, `Device`-nak pedig a meghajtó tényleges (*Linux* alatt értendő) elérési útját kell írnunk (`/dev/meghajto`).
`Filesystem`-nek (fájlrendszer), -ahogy a rögtönzött „súgó” is mutatja a konfigurációs állományban-, érdemes `win95` értéket adni (ez az összes ismert és elterjedt l*inux*os és *windowsos* fájlrendszert lefedi). Viszont ha véletlenül egy olyan régi géppel rendelkezünk, amin van *FAT16*-os *MS-DOS* partíció és ezt szeretnénk *Wine*-beli meghajtóként kezelni, `win95` helyett `msdos`-t adjunk meg.
Nagyon fontos dolog a *C:* meghajtó, amit külön ki kell emelnem. Azt tudjuk, hogy *Windows* alatt a *C:* az elsődleges merevlemez (mint *Linux* alatt a `/dev/hda1`), amire a rendszer települ. *Wine* alatt két lehetőségünk van a *C:* meghajtó használatára.
A javasolt lehetőség az úgynevezett *Fake-Windows* (puritánul fogalmazva kamu-*Windows*), ami annyit jelent, hogy a *Wine* létrehozza a *windows*os programok futtatásához szükséges alapvető könyvtárakat (`C:\windows`, `C:\system`, `C:\system32`, `C:\Program Files` stb.) és *.dll* fájlokat, ezzel tökéletes szimulációt teremtve. Amennyiben ezt választjuk (alapértelmezett), akkor a hamis-*Windows*unk C: meghajtója a `$HOME/.wine/drive_c` alatt érhető el.
Választhatjuk azt a lehetőséget is, hogy egy meglévő *Windows*-partíciót (telepített *Windows*-zal) használunk. Ekkor természetesen a tényleges elérési utat kell megadnunk a C: meghajtóhoz (például `/mnt/windows`).

Wine

```
[wine]
"Windows" = "C:\\Windows"
"System" = "C:\\Windows\\
➥System"
"Temp" = "X:\\"
"Path" = "C:\\Windows;C:\\
➥Windows\\System;X:\\;X:\\
➥test;Y:\\"
"GraphicsDriver" = "x11drv"
```

A *Wine* alapbeállításai. Általában ezen nem kell kézzel változtatni semmit. Ettől függetlenül itt megjegyezném a `GraphicsDriver` funkciót. Ez nagyon fontos, mivel a *windowsos* programok grafikusak. Így helyes beállítást igényelnek ahhoz, hogy ne fagyjanak (sajnos ez *Wine* alatt is előfordulhat). Győződjünk meg arról, hogy a `GraphicsDriver` `x11drv` legyen!

Wineconf

Az első érdeke szakasz a következő:

```
[Version]
; Windows version to imitate
➥(win95,win98,winme,nt351,nt40,
➥win2k,winxp,win2k3,win20,
➥win30,win31)
"Windows" = "win98"
; DOS version to imitate
;"DOS" = "6.22"
```

Ebben a szekcióban többféle dolgot is beállíthatunk. Először is meg kell adnunk a *Wine*-beli *Windows* típusát (`[Version]`). Ez lehet `"win95"`, `"win98"`, `"winme"`, `"nt351"`, `"nt40"`, `"win2k"`, `"winxp"`, `"win2k3"`, `"win20"`, `"win30"`, `"win31"`. Láthatjuk, hogy egészen a kezdetektől mostanáig bármilyen futtatási környezetet teremthetünk. Én javaslom a `win98` használatát, mivel ezzel az opcióval szinte mindegyik program hibátlanul fut. A *Wine DOS*-t is képes imitálni.
Lehetőségünk nyílik beépített *.dll* fájlok használatára (`[DllOverrides]`), tudniillik ezek nélkül *Windows* sem lenne. Rengeteg *.dll*-t tartalmaz a *Wine* alapból, ellenben mindig akadhatnak olyan hibaüzenetek egy program futtatása során, melyek *.dll* fájlokat hiányolnak. Ezeket a ➲ http://www.dll-downloads.com/ oldalon szerezhetjük be.
A *.dll*-ek kezelését a következő szakasz vezérli:

```
[DllOverrides]
; some dlls you may want to
➥change
"oleaut32" = "builtin, native"
"ole32" = "builtin, native"
"commdlg" = "builtin, native"
"comdlg32" = "builtin, native"
"shell" = "builtin, native"
```

x11drv

A egyik legfontosabb szakaszába érkeztünk a beállításoknak: a *Wine* alkalmazások megjelenítésének opciói. A legeslegfontosabb a *Display*. Enélkül nem fogunk látni semmit. A rendes *linuxos* (*X11*) *Display* nevét kell beírnunk. Ez majdnem minden esetben *:0.0*. A *„Managed”* funkció annyit tesz, hogy az *X* tekintse-e úgy a w*ine*-os alkalmazást, mintha az *linuxos* lenne. Konyhanyelven ez annyit jelent, hogy ha *Yes*-re állítjuk a *Managed* (menedzselt) opciót, akkor az alkalmazás „letehető” lesz a tálcára. Ellenkező esetben ha az értéket *No*-ra állítjuk, akkor a *Wine*-alkalmazás az

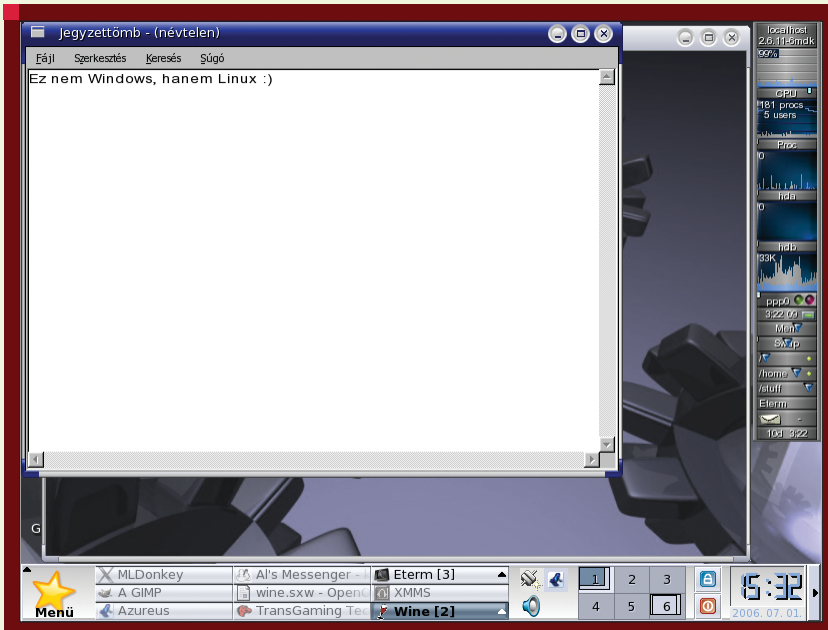
összes munkaterületen futásának végéig meg fog jelenni, ráadásul mindig felül.
Nézzük az ehhez kapcsolódó kódrészletet:

```
[x11drv]
; Number of colors to allocate
➥from the system palette
"AllocSystemColors" = "100"
; Use a private color map
"PrivateColorMap" = "N"
; Favor correctness over speed
➥in some graphics operations
"PerfectGraphics" = "Y"
; Color depth to use on
➥multi-depth screens
"ScreenDepth" = "16"
```

A többi beállítás itt opcionális. Különösebb boncolgatás nélkül (mindenki tapasztalja meg saját maga) elmondhatom, hogy az ***x11***-es rész minden opciójának érdemes igaz értéket adni, magyarul „Y” (azaz igen) értéket. Különösen a játékoknál lesz ez hasznos.
Ezen felül kiemelném a *Wine* kiváló portkezelését (soros, párhuzamos), mely lehetővé teszi akár a nyomtatást is.
A *Wine* portkezelése:

```
[serialports]
"Com1" = "/dev/ttyS0"
"Com2" = "/dev/ttyS1"
"Com3" = "/dev/ttyS2"
"Com4" = "/dev/modem"

[parallelports]
"Lpt1" = "/dev/lp0"
```

Wine-alkalmazásokkal igénybe vehetjük a világháló szolgáltatásait is, például futtathatunk ***mIRC***-et is, mellyel csatlakozhatunk az ***irc***-szerverekhez.
A legfontosabb részeket végigvettük, a többi beállítás áttanulmányozását a kedves olvasó kitartására és leleményességére bízom. Érdemes ezt a konfigurációs fájlt sokat böngészni, állítgatni, aztán kipróbálni, hiszen gyakorlat teszi a mestert. Ne felejtsünk el biztonsági mentést készíteni egy már működő konfigurációs állományról! Ez nagyban megkönnyíti a munkánkat, ráadásul a felesleges idegeskedésektől is megóv.
Bővebb információ a *Wine* beállításaival kapcsolatban a ➲ http://winehq.org/site/docs/wineusr-guide/config-wine-main oldalon található.



3. ábra Ez a jegyzettömb bizony nem Windows alatt fut

Használat

Ha végre túlestünk a beállításokon, akkor használatba vehetjük magát a *Wine*-t. De még mielőtt belemerülnénk, van itt néhány dolog, amit érdemes tisztázni. Először is, mint már említettem, a *Wine* nem *Windows*-emulátor. Tehát senki se számítson zöld asztalfelületre és startmenüre (aki erre vágyik, annak tudom ajánlani a *VMware*-t, amiről a ➲ http://vmware.com oldalon lehet információt találni). Másrészt ez a program egy parancssoros alkalmazás, tehát vérbeli ***linux***os alkalmazás. Ettől természetesen nem kell megijedni, mert pofonegyszerűen használható.
Volt róla szó, hogy a *Wine* egy ***wrapper***, ami futtatási környezetet jelent. A beállító állományban megadott adatok érvényesek erre a környezetre. Értelem szerűen, ha egy olyan ***windows***os alkalmazást szeretnénk elindítani *Wine* segítségével, ami egy, a beállító fájlban nem szereplő meghajtón van, akkor ezt nem fogjuk tudni megtenni. Magyarul a *Wine* csak azon a meghajtókat „látja”, amelyek helyesen szerepelnek a beállító fájlban. Ezért érdemes minden *.exe* fájlt rögtön a `$HOME/.wine/drive_c` könyvtárba menteni/helyezni.
Ezek után belépünk a könyvtárba a

```
cd $HOME/.wine/drive_c
```

paranccsal.

Tegyük fel, hogy egy programot szeretnénk telepíteni, melynek a telepítőállományát (***setup.exe***) már a virtuális C: meghajtóra helyeztük. Nincs más dolgunk, mint kiadni a

```
wine setup.exe
```

parancsot és máris kezdődhet a telepítés. A telepítés ugyanúgy zajlik, mint *Windows* alatt. Ha sikeresen (hibáktól mentesen) feltelepült a kívánt alkalmazás, akkor lépjünk be a ***Program Files***-ban lévő könyvtárba, melyet a telepítő hozott létre a telepített alkalmazásnak (a *Windows* a telepített programokat itt helyezi el) a

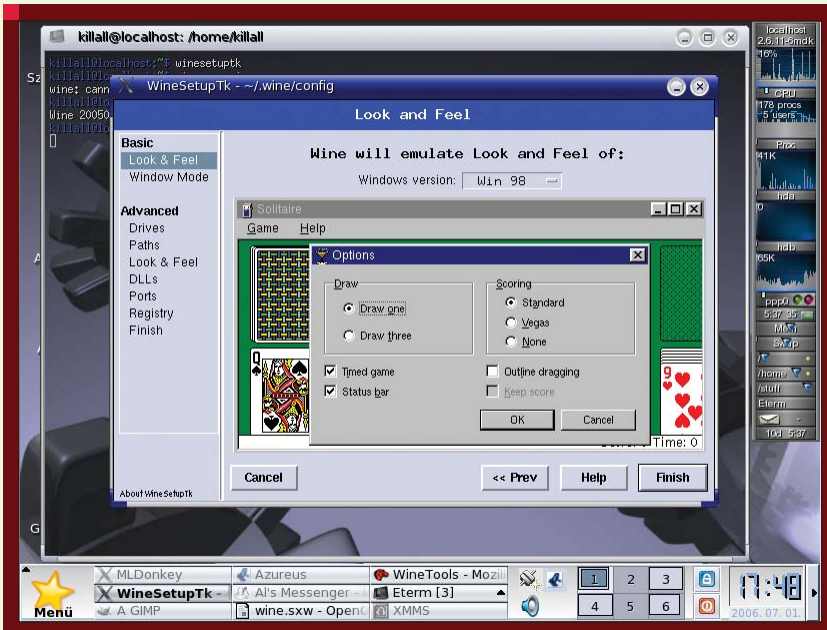
```
cd "$HOME/.wine/drive_c/Program
➥Files/Program_Neve"
```

paranccsal. Kis emlékeztető: az elérési út azért van kétszeres idézőjel között mert nélkülük a ***bash*** (Bourne Again Shell) csak a ‘`cd $HOME/.wine/drive_c/Program`’ parancsot értelmezné.
Ha beléptünk a könyvtárba, adjuk ki a

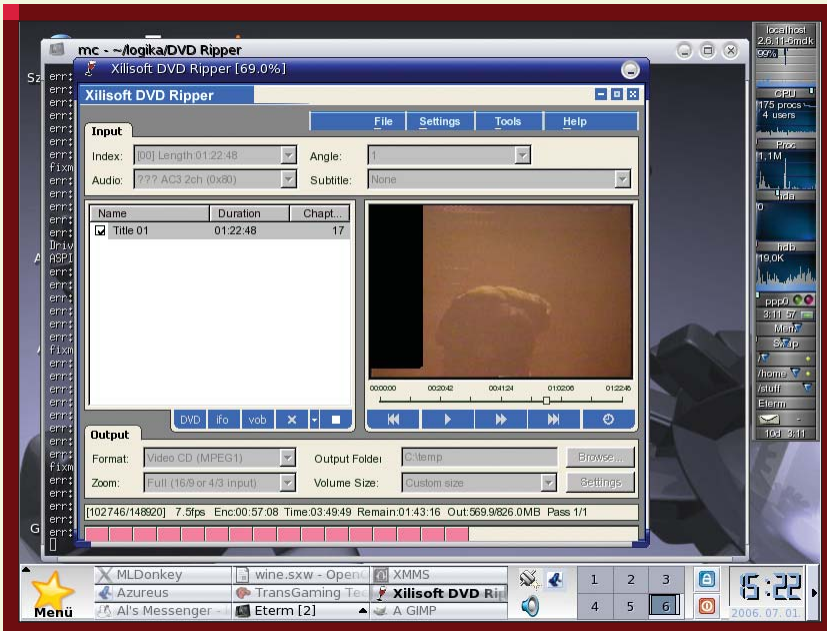
```
wine progam_neve.exe
```

parancsot, mellyel indítható az imént telepített program. Észrevehetjük, hogy minden egyes programindítás a

```
wine program_neve.exe
```



■ *4. ábra* A legkellemesebb Wine-beállító felület



■ *5. ábra* Én a DVD-k rippeléséhez is Wine-t használok

paranccsal történik. Jó esetben, ha mindent helyesen állítottunk be a beállítófájlban, más opciókra nem is lesz szükségünk. Azonban előfordulhat hogy egy programnak speciális opciókra (például extra *.dll*-ek betöltésére vagy más videó üzemmód megadására) van szüksége. Ezekre a legtöbb esetben konkrét megoldásokat találunk internetes fórumokon.
Aki szeretné minden egyes *Wine*-beli alkalmazásának indítását finomhangolni különféle opciókkal és extra argumentumokkal, annak nagy segítségére lehet a

```
man wine
```

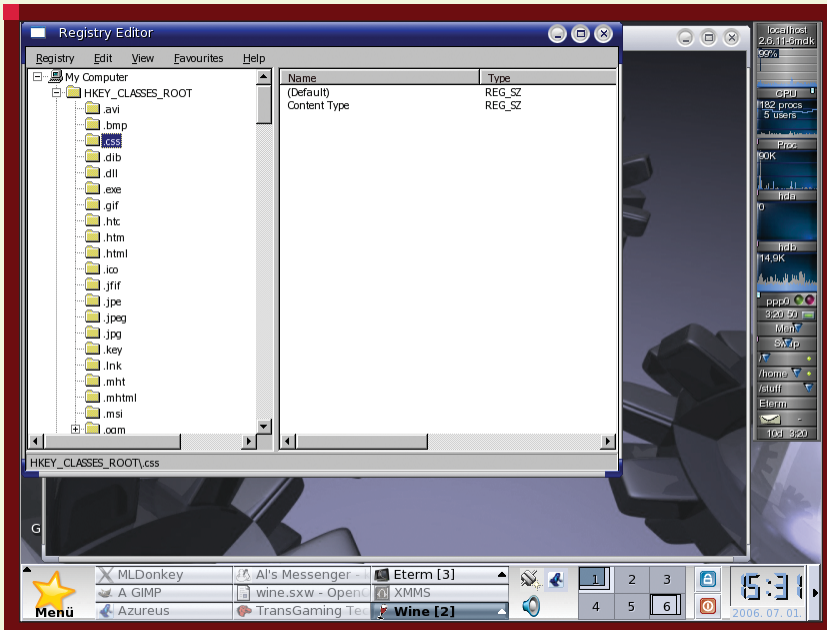
parancs, melyből minden indítási trükköt megtanulhatunk.
Természetesen a *Wine* alkalmazásoknak is készíthetünk ikonokat, illetve menübejegyzéseket, ezzel megkönnyítve a használatukat.

Támogatott Windows-alkalmazások

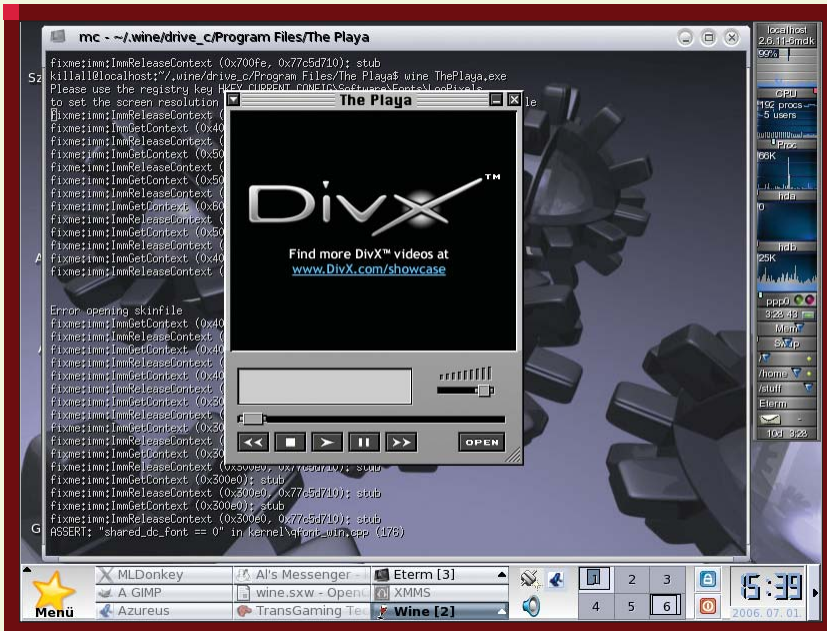
Akármennyire is szeretjük a *Wine*-t, néha bosszankodhatunk, mert sajnos nem támogatja az összes *windows*os alkalmazást.
A leleményes *Wine*-fejlesztők illetve a fanatikus felhasználók viszont megkönnyítették a helyzetünket: adatbázisba szedték a garantáltan működő programokat leírásokkal, képekkel.
Ez az adatbázis megtalálható a ➲ http://appdb.winehq.org oldalon, ahol többféleképpen is kereshetünk az alkalmázok listájában. Képernyőképek alapján (➲ http://appdb.winehq.org/viewScreenshots.php), tehát ha megtetszik egy alkalmazás, a képre kattintva rögtön mindent megtudhatunk róla és a *Wine*-beli működéséről. Kereshetünk Linux-disztribúció specifikusan is alkalmazásokat (➲ http://appdb.winehq.org/distributionView.php), *AGNULA/DeMuDi 1.2.0*-tól *Zenwalk 2.4*-ig.
Lehetőségünk nyílik terjesztőcég szerint (➲ http://appdb.winehq.org/vendorview.php), illetve programkategória szerint (➲ http://appdb.winehq.org/appbrowse.php) is böngészni.
Hogy néhány példát is nézzünk: a *FarCry* és az *Aliens vs. Predtor 2* játékok, illetve a *Windows Media Player 6.4* tökéletesen működik *Wine* alatt.

Cedega

Létezik egy pár éve alakult projekt, mely a *Wine*-on alapul, de kifejezetten a *windows*os játékok futtatásával foglalkozik. A vélemények megoszlanak arról, hogy a korábban *Winex* néven ismert program jobb-e, mint a *Wine*. Egyesek szerint igen, mások szerint nem. Saját véleményem szerint az ingyenes *Wine* van annyira jó, mint a fizetős *Cedega*. Természetesen létezik belőle ingyenes *CVS* verzió is, igaz, ez jóval "lebutítottabb" a megvásárolható változatnál. Mindenki maga döntheti el, hogy melyik verziót preferálja.
Nem akarok elfogult lenni, de én már futtattam sima *Wine*-on a *Red Faction* névre hallgató, erősen *DirectX*-re épülő játékot, holott a *Cedega* reklámkampánya pont a *DirctX* támogatásra volt/van kihegyezve...
További információk a projektről a ➲ http://transgaming.com oldalon találhatók.



6. ábra Wine alatt is szerkeszthetjük a Registry-t



7. ábra The Playa DivX lejátszó Wine alatt

Crossover Office

Lényegesen bizalomgerjesztőbb és szimpatikusabb projekt a szintén *Wine*-klón c*xOffice*. Ez az alkalmazás szintén nem ingyenes (*trial*-verzió viszont elérhető), akárcsak a *Cedega*. Ez a társulat azt a célt tűzte ki maga elé, hogy addig finomítják a *Wine* kódját, amíg el nem érik, hogy bizonyos programok tökéletesen működjenek vele. Magyarán ez az alkalmazás csak néhány program tökéletes futását garantálja, azok listája viszont impozáns. Többek között az összes ismert *MS Office* verzió, I*nternet Explorer 6*, *Lotus Notes*, *Adobe Photoshop 7*, stb.
A projekt honlapja a ➲ http://www.codeweavers.com/ oldalon található.

Segédprogramok Wine-hoz

A továbbiakban két segédprogramot fogok bemutatni. Az egyik a *Wine* beállítását, a másik a *Wine*-alkalmazások installálását hivatott megkönnyíteni.

Winesetuptk

Ez a kis segédprogram egy nagyon kellemes beállítási lehetőséget nyújt nekünk. Az előbb vesézett bonyolult és hosszú beállítófájlt (illetve annak fő részeit) ezen alkalmazás segítségével pár kattintással beállíthatjuk. Egyetlen hátránya van: nincs a projektnek honlapja, így ha nincs feltelepítve a gépünkre, akkor a *Google* segítségével kell „levadásznunk” egyet a disztribúciónkhoz. Például a ➲http://rpmfind.net oldalon találhatunk csomagokat *Mandriva Linux*hoz. A *winesetuptk* nem része a *Wine*-nak.

WineTools

A *WineTools* egy menü-orientált alkalmazás közel 90 *windowsos* program biztonságos telepítéséhez.
Sajnos nem minden *Wine* verziót támogat (köztük a 20050211-et sem, ami nekem van), viszont a legújabb verziót kétségtelenül támogatja (*recommended* párosítás, azaz ajánlott).
A projekt honlapja a ➲ http://www.von-thadden.de/Joachim/WineTools/ oldalon érhető el.
A támogatott *Wine* verziókat a ➲ http://www.von-thadden.de/Joachim/WineTools/index.html#download oldalon tekinthetjük meg, ahonnan egyúttal le is tölthetjük az alkalmazást. A *WineTools* által támogatott programok listáját pedig a ➲ http://www.von-thadden.de/Joachim/WineTools/wt0.9jo.html webhelyen találjuk. Innen egyébként egyéb hasznos dolgok is letölthetőek (például hasznos *.dll* fájlok, *dcom95.exe*, *windows installer* telepítőállomány).
Akik szeretik biztonságban érezni magukat és szeretik a grafikus programokat, azoknak mindenképpen ajánlott kipróbálni a *WineTools*-t.

Apagyi György, (killall)
(killall@linuxforum.hu)

24 éves, jelenleg az ELTE programozó matematikus szakán másodéves hallgató. Hobbija a zene (gitározás), az olvasás (Stephen King) és a számítástechnika (Linux, Unix, VMS).

A Firefox kiterjesztései (2. rész)

Sok érv és ellenérv hangzott el arról, hogy miért is érdemes honlapot Firefox segítségével fejleszteni. Személy szerint én azt tapasztaltam, hogyha egy oldal helyesen jelenik meg e böngésző alatt, akkor más fajták alatt is helyes fog megjelenni. Ráadásul a Firefoxhoz telepíthető olyan kiterjesztések, amelyek a webfejlesztő munkáját segítik. Ilyen például a Web developer-t, az ezt kiegészítő HTML validator-t valamint a FireFtp.

Mielőtt ezek ismertetésére rátérnék szeretnék azon felhasználóknak segítséget nyújtani, akik szívesen bővítenék böngészőjüket, de eddig visszatartotta őket az, hogy nem vagy csak kevéssé tudnak angolul. Nekik érdemes ellátogatniuk a ➲ *firefox.hu/kiterjesztesek/firefox* címre, ahol jelenleg 98 kiterjesztés érhető el, melyek ráadásul honosítottak is. Telepítésük úgy történik, hogy a kiválasztott kiterjesztés leírása után található ***név.xpi***-re kattintunk. Az első telepítés előtt a képen látható figyelmeztetés jelenik meg, melynél a ***Beállítások szerkesztésére*** kell kattintani, és a megjelenő új ablakban fel kell venni a ***firefox.hu***-t az engedélyezett webhelyek listájára.
Ezt a műveletet csak egyszer kell megtenni. Ezután ismét a ***név.xpi***-re kell kattintani és a megjelenő ablakban elindítható a telepítés. Ha ez befejeződött akkor már csak újra kell indítani a böngészőt, s máris elérhető az új funkció.

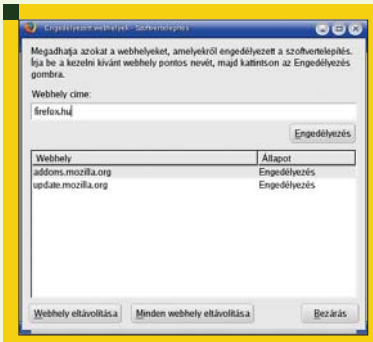
1. ábra A képen a Firefox szoftvertelepítésekhez kapcsolódó figyelmeztetése látható

HTML Validator

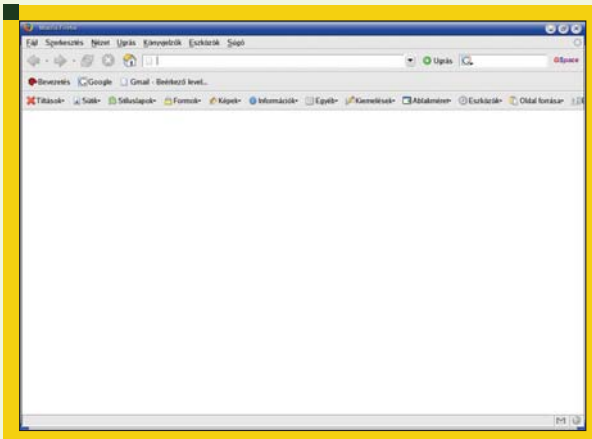
A ***HTML Validator*** a ***W3C*** által készített ***Tidy*** nevű program továbbfejlesztett változata. Segítségével leellenőrizhetjük, hogy az általunk készített honlap mennyire felel meg a ***W3C*** által elkészített szabványnak A program 3 csoportba sorolja az oldalakat:

- Hibákat tartalmazó. A hibák olyan problémák, melyeket a program nem tud automatikusan javítani vagy nem érti a kód adott részét.
- Figyelmeztetéseket tartalmazó. A figyelmeztetések olyan problémák, melyeket a program automatikusan tud javítani, és ezen javításokat meg is mutatja.
- Fentiek egyiket sem tartalmazó. Problémamentes oldalak.

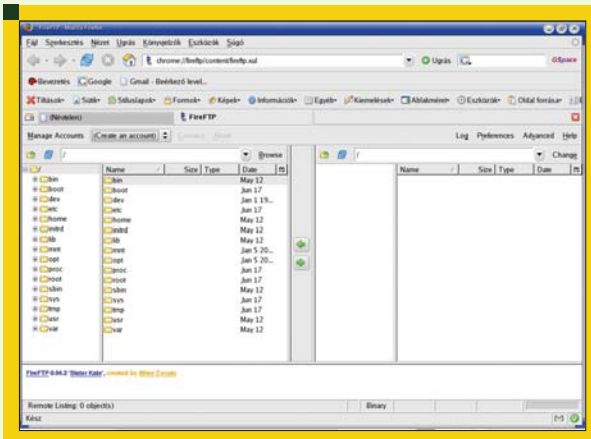
A kereteket tartalmazó oldalaknál, minden keret külön kerül osztályozásra. A vizsgálat lokálisan történik, tehát semmilyen adat nem kerül elküldésre harmadik fél számára. A vizsgálati eredményről a böngésző státuszsorában megjelenő ikon tájékoztat, melyre, ha kattintunk megjelenik a ***forrásnézet ablak*** ahol a program kijelöli a nem szabványos sorokat. A kereteket tartalmazó oldalaknál előbb ki kell választani, hogy melyikkel akarunk foglalkozni. Ha a jobb alsó sarokban lévő ***Forráskód javítása…*** gombra kattintunk megjelenik a program szerinti helyes kód. Itt összehasoníthatjuk az



2. ábra A képen Firefox engedélyezett webhelyek – szoftvertelepítés ablaka látható



3. ábra A képen Web developer eszköztára látható



4. ábra A képen FireFtp induló képernyője látható

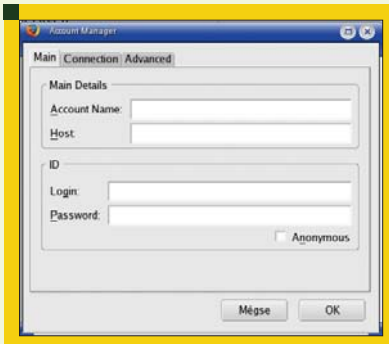
eredeti kódot és kinézetett az újjal. A javítottat meg lehet jeleníteni ***html*** és ***xhtml*** formátumban, ***ascii*** vagy ***unicode*** kódolással valamint felajánlja, hogy a lehetséges tulajdonságokat stílus lapra cseréli.
A ***HTML Validator*** kiegészíti a ***Web developer-t,*** tehát ezen funkciók elérhetőek az utóbbi programon belül is.

Web developer

A ***Web developer*** aktuális oldalhoz tartozó rengeteg információ megjelenítésére képes. Ezen információk a különböző elemekkel (ilyenek: képek, keretek, űrlapok, táblázatok, blokkok) együtt jelenek meg. Így nem kell kikeresni őket a forráskódból. Viszont vizuálisan jól láthatóak erre a legjobb példa, hogy a linkek szövege és az elérési útja egyszerre jelenik meg.
Letiltható a ***javascript*** illetve a ***java*** futtatása, a színek megjelenítése, a nyomkövetés. Ha egy oldalon megtetszik valamilyen szín, amit érdemes lenne használni saját oldalunkhoz, akkor az ***Információk/színinformációk-ra*** kell kattintani és egy új fülön megjelenek a kigyűjtött színek és azok kódjai. Szép táblázatos formában megjeleníthetőek az oldalhoz tartozó metainformációk is.
Lehetőség van továbbá a ***HTML*** kód, az oldalhoz kapcsolódó stíluslapok, linkek ellenőrzésére, de itt a vizsgálni kívánt kód a ***HTML Validatorral*** ellentétben elküldésre kerül az Interneten, valamint nincs lehetőség lokális fájlok ellenőrzésére.

FireFtp

Sokan kérdezik tőlem, hogy az elkészült honlapjuk hogyan kerül fel a világhálóra. Nekik mindig azt szoktam



5. ábra A képen FireFtp Manage Accounts ürlapja látható

tanácsolni, hogy első lépésben válasszanak ki egy tárhely szolgáltatót, ahova regisztrálás után már tölthetik is fel a kész honlapot.
Ennek legegyszerűbb módja, ha valamilyen ***Ftp protokolt*** (fájl továbbítási protokollt) ismerő programot használunk. Rengeteg lehetőség közül választhatunk, de nem kell sokat keresgélni, mert a ***Firefox*** erre is kínál megoldást, ez a ***FireFtp***. A program feltelepítése után az ***Eszközök/FireFtp*** segítségével indítható el.
Első lépésben a megjelenő új ablak bal felső sarkában lévő ***Manage Accounts***-ra kell kattintani. Ezután a ***New…*** lehetőséget kiválasztva kell megadni az adatokat, melyek segítségével létrejön a kapcsolat a tárhely szolgáltatónk szervere és saját gépünk között.
A megjelenő űrlapon a következő adatokat kell megadni:

- ***Account name***: A kapcsolat neve. Ha a későbbiekben több kapcsolatot is létrehozásra kerül, érdemes jól beazonosítható nevet megadni.
- ***Host***: A szolgáltató ftp címe.

6. ábra A képen FireFtp prefrences ablaka látható

- ***Login+passord***: A regisztráláskor megadott belépési nevünk és jelszavunk.

Az ***OK*** gomb megnyomása után már csak a ***Connect*** gombra kell használni. Ha mindent jól adtunk meg akkor a program kiírja, hogy él a kapcsolat és a középen található jobbra mutató nyíllal indulhat is a feltöltés.
Gyakorlottabbak tudják, hogy néha szükség lehet a feltöltés bináris ás ***ASCII*** módja között váltásra. Ezt megtehetjük, ha a kiterjesztés jobb felső sarkában lévő ***Preferences*** gombra kattintunk, majd a megjelenő ablakban a ***downloads/uploads*** fülön tudunk váltani a módok közül.

Fekete Imre
(imre.fekete@gmail.com)

Kedvenc böngészőm még mindig a Firefox.

Kontact – Személyi információkezelés KDE módra

A Kontact egy Linux operációs rendszeren, elsősorban KDE környezethez fejlesztett úgynevezett PIM („personal information management”, személyi-információ kezelés) alkalmazás-csomag. Egyetlen felhasználói felület alá csoportosítja az e-mail, napló, határidőnapló, címjegyzék, megjegyzések, hírcsoportok és hírcsatornák kezelését.

Hasonló megoldásokat kínál a linuxos ***Evolution*** illetve a ***Windows*** alapú ***Microsoft Outlook***. Ebben az írásban a teljesség igénye – és szándéka – nélkül megpróbáljuk a ***Kontact*** számos funkciója közül a legfontosabbakat bemutatni azzal a céllal, hogy betekintést nyújtsunk a ***Kontact*** eszközeibe és lehetőségeibe. Talán sikerül felkelteni az olvasó érdeklődését és bátorítást nyújtani a kipróbáláshoz.

A ***Kontact*** a ***KDE*** grafikus környezet fejlesztői gárdájának, pontosabban a ***KDE PIM*** csapat munkájának az eredménye. Nekik – és természetesen az egyes összetevők fejlesztőinek – köszönhető az, hogy a ***Kontact*** a 3.4.x verziószámhoz érve már egy szinte teljes értékű ***PIM*** alkalmazássá nőtte ki magát. A ***Kontact*** fejlesztése ***KDE/Qt*** alapokon nyugszik, így a ***KDE*** felülete alá illeszkedik natívan, de természetesen más grafikus asztali környezetek alatt is (***Gnome***, ***Xfce***, ***Enlightenment***, ***fwvm***, stb.) használható a szükséges alapkönyvtárak telepítése után.

Telepítés, ismerkedés

Ebben az írásban a ***Kontact 1.1.2*** verzióját mutatjuk be, ***KDE 3.4.2*** alatt. A telepítés menete – mint már megszokhattuk – eléggé disztribúciófüggő. ***Debian*** és változatai alatt (***Knoppix***, ***Mepis***, ***K/Ubuntu***, ***Libranet***, stb.) az ***apt*** csomagkezelővel parancssorból, illetve például a ***Synaptic*** alkalmazással grafikus felület segítségével a ***kontact*** nevű csomagot kell első lépésben telepítenünk (`apt-get install kontact`). Ezután a ***Kontact*** már parancssorból és a ***KDE*** menükből is elérhető (az „Office” csoportban) és indítható. Ha az összetevők nem voltak előzetesen telepítve, akkor alapesetben a ***Kontact*** gyakorlatilag semmiféle hasznos szolgáltatást sem nyújt. Ahhoz, hogy kihasználjuk a lehetőségeit, telepítenünk kell az egyes összetevőket, amelyek a *1. Táblázatban* láthatók. A telepíthető csomagok nevei megegyeznek a feltüntetett alkalmazás-nevekkel.

A ***Kontact*** gyakorlatilag egyik iskolapéldája annak, hogy milyen magas szintű professzionális megoldásokat készíthetünk ***KDE*** alapokon.

A ***Kontact*** egyes funkciókat megvalósító összetevői mind előzőleg már létező, külső ***KDE***-s alkalmazások, amelyeket természetesen külön-külön, egymástól teljesen függetlenül is használhatunk. Ezeknek az alkalmazásoknak az egy felület alá történő ilyen szintű integrálását a ***KDE KParts*** technológiája teszi lehetővé, aminek elsődleges célja pontosan a ***KDE***-s alkalmazások könnyű egymásba-ágyazhatósága és integrálhatósága. Ennek eredménye az, hogy noha külső alkalmazásokat integrál, a ***Kontact*** egy teljesen egységes felületet biztosít, amely könnyen kezelhetővé és áttekinthetővé teszi ezeknek az eszközöknek a használatát.

Telepítés után a ***Kontact*** első indítása az úgynevezett ***Summary View***-t mutatja, amely egy összefoglaló lapja az összes összetevőnek. Egy oldalon tartalmazza a közeledő időpontokat, új levelek számát, stb. Természetesen ehhez az egyes összetevőket be kell

1. Táblázat *A Kontact funkciói és a megfelelő összetevők*

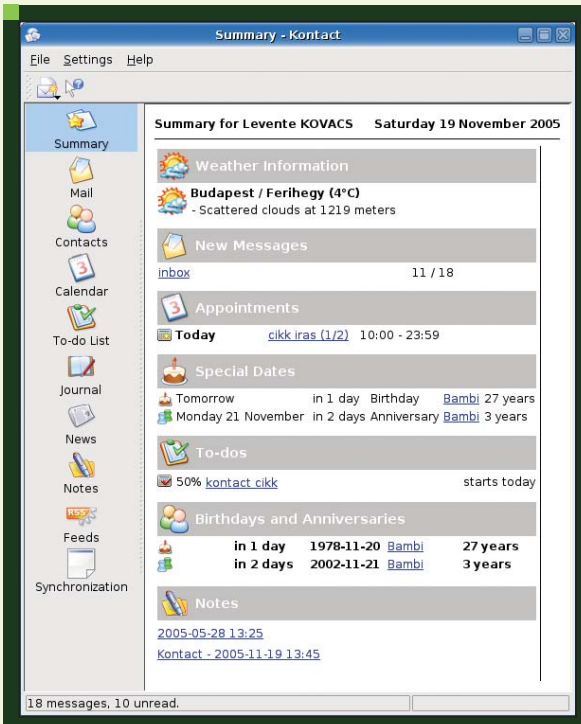
Funkció neve	Funkció leírása	Az összetevő neve
Mail	E-mail kezelés	*KMail*
Contacts	Címjegyzék	*KAddressBook*
Calendar	Határidőnapló	*KOrganizer*
Todo list	Tennivalók	
Journal	Napló	
News	Hírek, hírcsoportok	*KNode*
Notes	Jegyzetek	*KNotes*
Feeds	Hírcsatornák	*Akregator*
Synchronization	*Palm* szinkronizáció	*MultiSync*

állítanunk, például az e-mail kezelő *KMail* komponensben be kell állítanunk a levelezéshez szükséges adatokat, a határidőnaplóban vezetnünk kell a határidőket, stb. Ezekre a későbbiekben rátérünk. Az *1. ábra* egy olyan összefoglaló nézetet mutat, ahol már néhány összetevő beállításra került.

Az alkalmazás főablakának bal oldalán az egyes összetevők gyorsan elérhetők a megfelelő ikonra kattintva. A *Kontact* főmenüje annak függvényében változik, hogy éppen melyik összetevőt választottuk ki. Minden összetevő esetén a ***Kontact Settings (Beállítások)*** menüjéből elérhetők az aktuális összetevő beállításai, ahol ezeket személyre szabhatjuk.

Noha a Kontact első nézésre csupán egy keretet ad néhány külső alkalmazásnak, valójában több ennél. Nem csupán egy helyről teszi elérhetővé az összetevőit, hanem például a ***húzd és ejtsd (drag&drop)*** funkciót is biztosít a különféle összetevők között, ami nagyban növeli a funkcionalitást és a hasznosságát. A későbbiekben, és az egyes összetevők kapcsán erre a fontos tulajdonságra még részletesebben kitérünk.

A következőkben röviden sorra vesszük a *Kontact* összetevőit. Mindegyik esetében csak a legszükségesebb beállításokra szorítkozunk, amelyek ahhoz szükségesek, hogy az olvasó gyorsan birtokba vehesse a *Kontactot* és elkezdhesse a használatát. Az egyes összetevőkkel részletesen megismerkedhet az olvasó az ezekkel foglalkozó különálló cikkekből. Illetve minden összetevő esetén megadjuk a megfelelő webes elérhetőségeket, ahonnan az adott összetevőről további információk gyűjthetők. Ehhez egy jó kiindulási pont a *KDE* dokumentációs projektjének *Kontacttal* foglalkozó on-line dokumentációja.

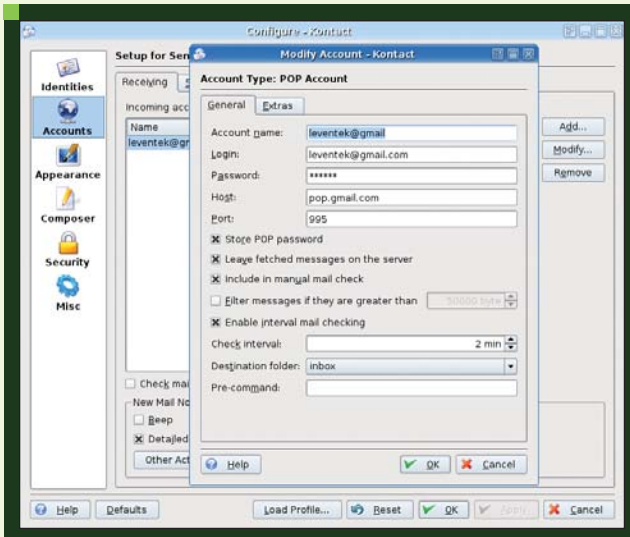


1. ábra A Kontact összefoglaló nézete („Summary View") amely az alkalmazás indításkor jelentkezik

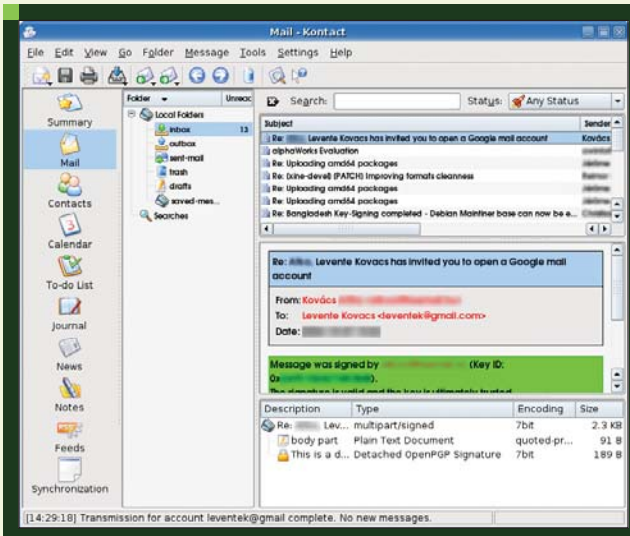
Levelezés

Az e-mail kezelést az integrált *Kmail* biztosítja. A *Kontactba* integrált és a különálló *KMail* között csak megjelenésbeli különbség van, működésbeli nincs! A levelek egy helyen tárolódnak, nem vesznek el, akár a *Kontactba* ágyazottan, akár a különálló *KMail*-ben végezzük a levelezést. A levelezés használatához a legfontosabb két lépés a bejövő és kimenő levelező szolgáltató adatinak beállítása. Ha a *Mail* összetevőt kiválasztjuk a *Kontactban*, akkor a *Settings* menüben a *Configure KMail* menüpontban találhatjuk a *KMail* beállításait.

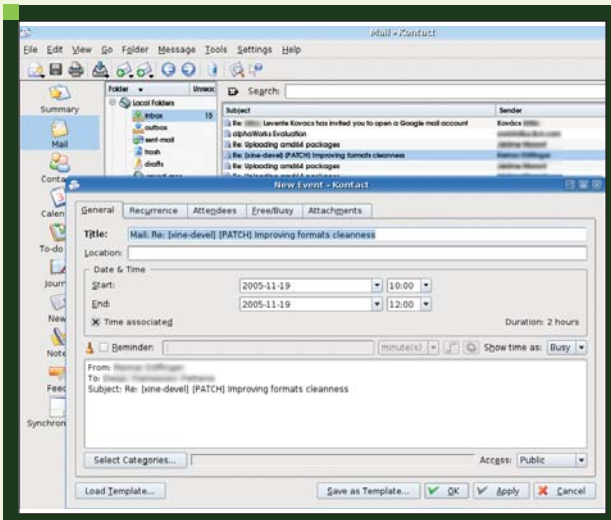
Első lépésekként az *Identities* és az *Accounts* tulajdonságokat kell megadnunk. Az elsőben (*Identities*, azonosítók, identitások) saját adatainkat kell megadnunk, azt a nevet és e-mail címet amelyek az általunk küldött levelekben szerepelni fog. A másodikban (*Accounts*, postafiókok) azoknak a postaládáknak a beállításait kell megadnunk, amelyekről érkező leveleket szeretnénk a *Kontactban* (és egyúttal a *KMail*-ben kezelni) ill. annak a szervernek az elérhetőségét, amelyik a leveleinket fogja küldeni. *POP* és *IMAP*



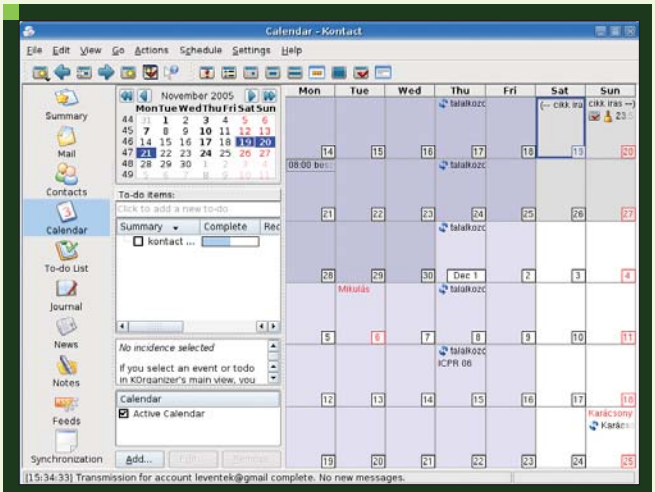
2. ábra Bejövő, POP e-mail postafiók beállításai



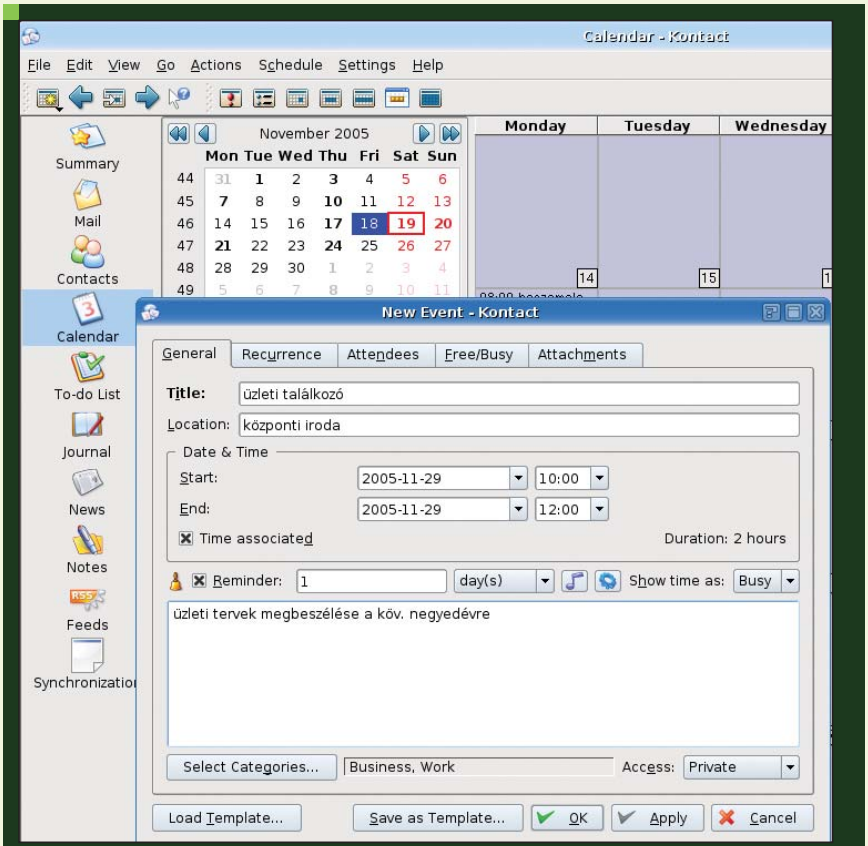
3. ábra Levelek olvasása beállított postafiók esetén



4. ábra Levél fejlécének a határidőnaplóba történő áthúzásakor új bejegyzés készüli



5. ábra A Kontact határidőnapló („Calendar") nézete



6. ábra Új időpont bejegyzése a határidőnaplóba

hozzáférést is megadhatunk, és egyszerre több postafiókot is, amelyekből beérkező leveleket külön mappákban tárolhatjuk. A *2. ábra* beérkező levelek fogadásához egy *POP* postafiók beállításait mutatja. A *3. ábra* egy beállított postafiók esetén levelek olvasási ablakát mutatja. Természetesen még számos beállítási lehetőségünk van a *KMail*-en belül (levél formátuma, digitális aláírás, kinézet, stb.) amelyekkel itt most nem tudunk részletesen foglalkozni.

A *Kontact* lehetőséget ad arra, hogy egy érkezett levélből automatikusan, húzd és ejtsd módszerrel gyorsan felvehessünk egy új kapcsolatot a címjegyzékbe, vagy pl. hogy új időpontot hozhassunk létre a határidőnaplóban. Ezekhez csupán azt kell tennünk, hogy az aktuálisan olvasott levél fejlécét egérrel ráhúzzuk a *Contact (Címjegyzék)* illetve *Calendar (Határidőnapló)* ikonra. A címjegyzékhez automatikusan hozzáadódik a küldő, a határidőnaplóban egy új bejegyzés készül, amelyet jóvá kell hagynunk *(4. ábra)*.

Címjegyzék

A *Kontact* címjegyzék összetevője egy egyszerű felületet ad a kapcsolatok adatainak gyors hozzáadásához ill. szerkesztéséhez, a *KAddressBook* integrálásával. Adott személyt kiválasztva jobb egérgombbal kattintva az *Edit contact*-ot választva szerkeszthetjük az adatait. Új kapcsolatot a *File->New contact* menüponttal ill. az eszköztár első ikonjával hozhatunk létre. Ha egy kapcsolat címét egérrel a határidőnaplóba vagy a tennivalók listájába húzunk (lásd később) automatikusan új találkozó-időpontot tudunk létrehozni.

Határidőnapló

A *Kontact* határidőnapló összetevőjét az integrált *KOrganizer* alkalmazás képezi. A határidőnapló nézetben *(5. ábra)* napi, heti, háromnapi, heti vagy havi nézet választható. Megadhatunk egyszeri vagy ismétlődő időpontokat, választhatunk vagy definiálhatunk új találkozó-típusokat (munka, randi, üzleti találkozó, stb.), egyedi színeket rendelhetünk hozzájuk, emlékeztetést kérhetünk adott idővel az időpont előtt, *(6. ábra)* stb.

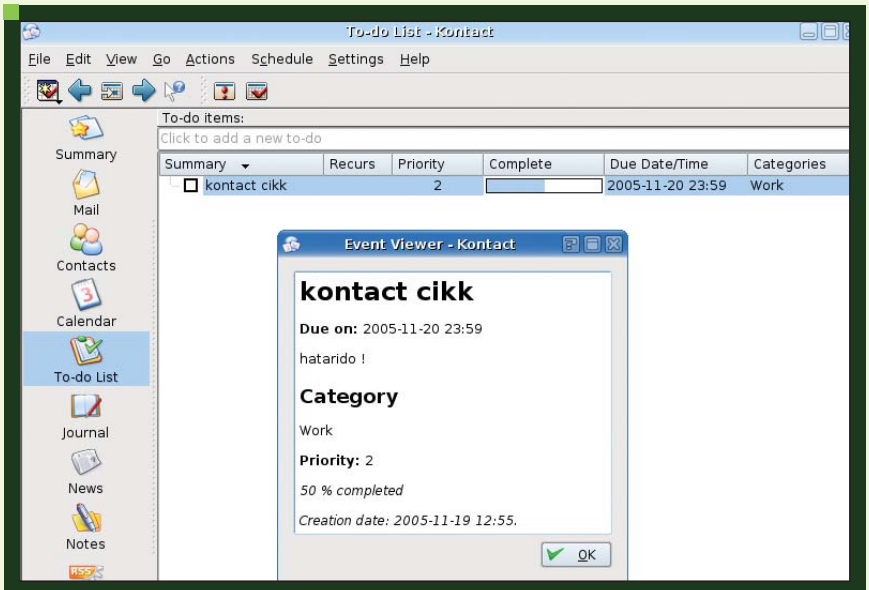
Tennivaló-lista, megjegyzések, naplóvezetés

Tennivalóinkat is nyilvántarthatjuk úgynevezett *„To-do”* listában, akár a határidőnaplóban közvetlenül *(5. ábra)*, akár a ***Kontact To-do List*** összetevőjében. Kategóriákba szervezhetjük a tennivalókat, nyomon követhetjük a teljesítés mértékét *(7. ábra)*. Valójában a tennivalók kezelése ugyanúgy a ***KOrganizer*** határidőnapló része, a ***Kontact*** viszont egy könnyebben áttekinthető formában ad lehetőséget a kezelésére.

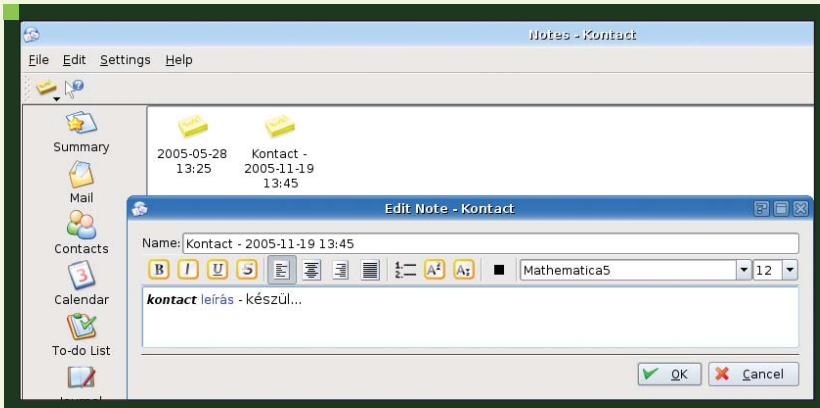
A címjegyzékből egy kapcsolatot egérrel a tennivalók listájába (***To-do List***) húzva automatikusan egy találkozó (***Meeting***) kategóriájú bejegyzés kerül a listába.

Szintén lehetőségünk van napló vezetésére a ***Kontacton*** belül. Ezt a funkciót szintén a ***KOrganizer*** szolgáltatja, de a ***Kontactban*** kiemelt helyett kapott. Napló nézetben (***Journal***) e naptár bármelyik napját kiválasztva szöveges bejegyzést készíthetünk, amely eltárolódik és utólag természetesen visszakereshetők, olvashatók és szerkeszthetők.

Egyszerű szöveges megjegyzések készítésére a ***Notes*** nézetben van lehetőségünk *(8. ábra)*, Könnyen, gyorsan készíthetünk feljegyzéseket, egy egyszerűsített szerkesztő segítségével, amiben a szöveg színét, méretét és a betűtípust alakíthatjuk igény szerint.



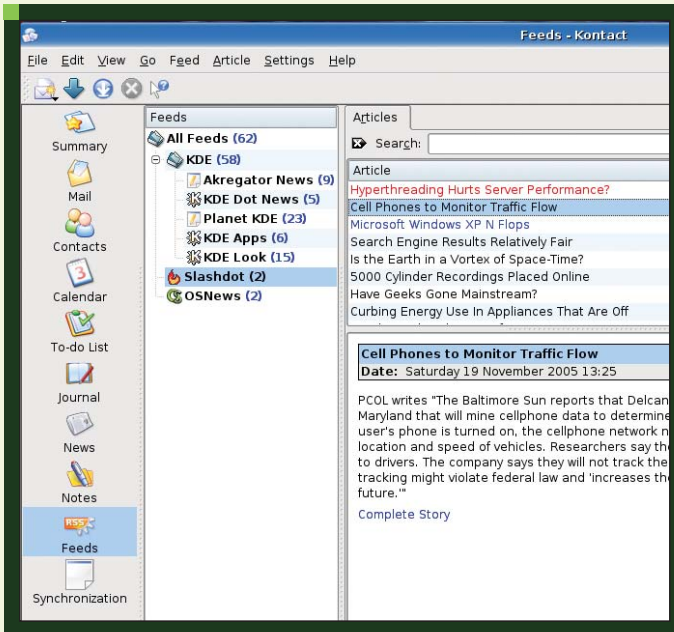
7. ábra Tennivalók listája, „To-do List” nézetben



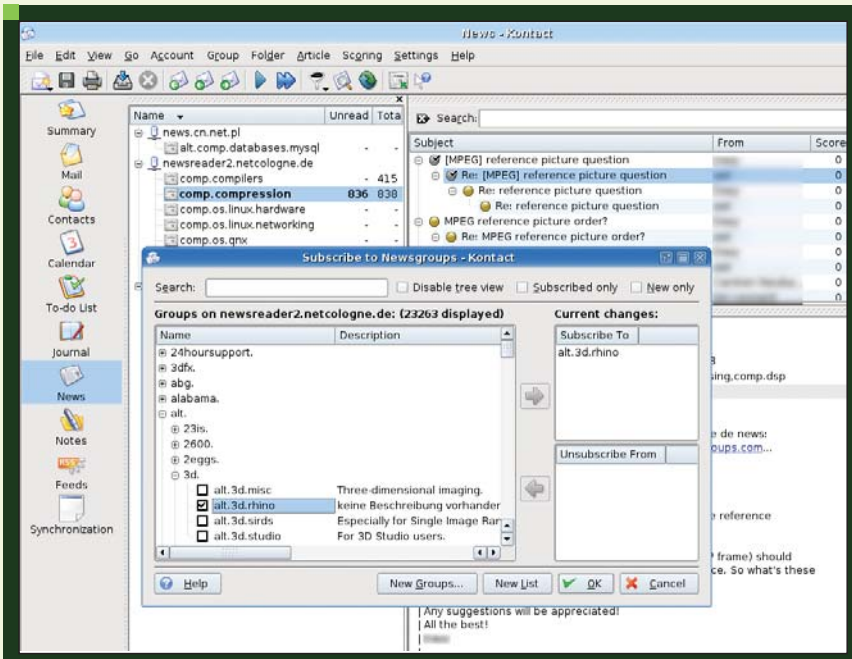
8. ábra Szöveges feljegyzések írására alkalmas a „Notes” összetevő

Hírek, hírcsoportok

A ***Kontactban*** elérhetünk integráltan két olyan funkciót, amely hírek/hírforrások távoli nyomkövetésére szolgáló kényelmes megoldást nyújtanak. Az egyik, a „Feeds” (hírszolgáltatás) összetevő, amit az integrált ***KNode*** alkalmazás szolgáltat, Web-oldalak hírszolgáltatásainak (RSS) olvasását teszi lehetővé *(9. ábra)*. A másik a ***News (Hírek)*** összetevő amit a ***Kontactba*** integrált ***akregator*** valósít meg, amely ***Hírcsoportok (Newsgroup)*** olvasását teszi lehetővé. Hírcsoportok olvasásához először egy ***hírszerverre (news server)*** kell jelentkeznünk, majd feliratkozni egy ***hírcsoportba (newsgroup)*** *(10. ábra)*. Hírszerverek megtalálásában segítségünkre lehet egy ***webes*** kereső szolgáltatás,



9. ábra Web-oldalak hírszolgáltatásainak (RSS feed-ek) olvasása a „Feeds” összetevőben



10. ábra Hírlistákra („newsgroup”) feliratkozás, olvasás, írás

ahol a *„free news servers”* címszavakra keresve hamar célhoz érhetünk.

Szinkronizálás

Az eddigiekben bemutatott összetevők mellett a ***Kontact*** egy olyan funkciót is tartalmaz, amely fontos lehet akkor, ha több helyen/számítógépen dolgozunk, ha több hordozható eszközünk is van (laptop, PDA, stb.) és szeretnénk ha a határidőnaplónk mindig friss lenne.

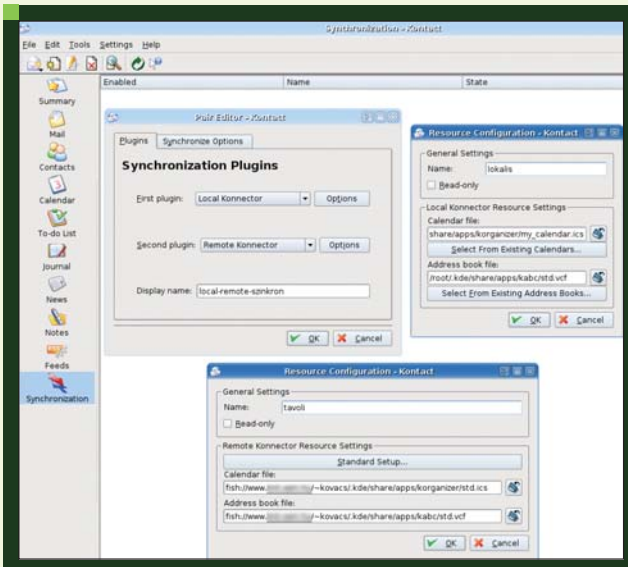
Ehhez ad segítséget a ***Kontact Synchronization (Szinkronizáció)*** összetevője. A *11. ábra* arra mutat példát, hogy hogyan szinkronizálhatjuk a lokális címjegyzékünk és határidőnaplónk egy távoli címjegyzékkel/határidőnaplóval, mindezt biztonságos kapcsolaton keresztül (***ssh/sftp***). Lehetőségünk van lokálisan tárolt különböző határidőnaplók között, illetve ***PDA***-nkkal is szinkronizálni, amelyet a ***KPilot*** alkalmazás tesz lehetővé.

Groupware szerverek támogatása

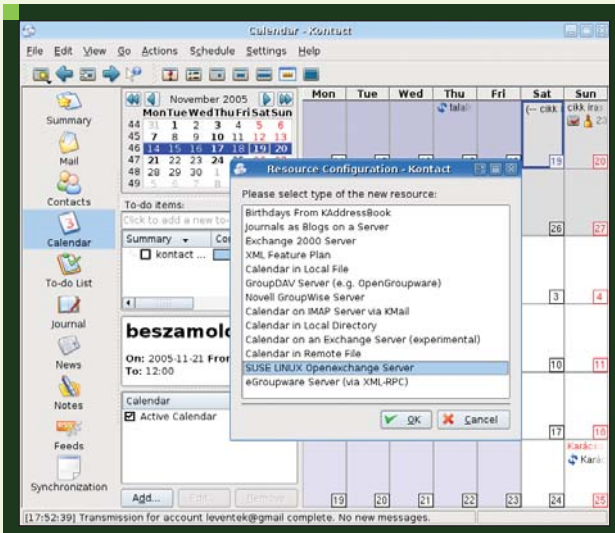
Fontos kérdés egy ***PIM*** alkalmazás esetében az, hogy milyen mértékben tud együttműködni központi ***groupware szerverekkel***. Ez azért is kiemelten fontos, mert a manapság vállalati környezetben szinte teljesen – a számok alapján – egyeduralmú ***Microsoft Windowson*** futó ***Microsoft Exchange*** szerver használata mellett fontos kérdés az, hogy egy adott ***PIM*** alkalmazás milyen szerverekkel képes együttműködni és milyen mértékben, hiszen ilyenkor – jó esetben – ez dönti el, hogy milyen helyi ***PIM*** alkalmazást választ a felhasználó. Természetesen a valóságban nem csupán ez a választási szempont, még akkor sem, ha az adott alkalmazás eléggé sokrétű és esetleg még ingyenes is, de ez egy másik történet.

A ***Kontact*** számos groupware szerverrel képes együttműködni *(11. ábra)*, mint a ***Suse Openexchange***, a ***Novell Groupwise*** vagy a ***Kolab***, de részlegesen a ***Microsoft Exchange*** szervert is támogatja és a közeljövőben teljes körű támogatás várható.

Más ***PIM*** alkalmazásokkal összevetve, és itt a ***Gnome/Novell Evolution*** és a ***Microsoft Outlook*** az amivel leginkább számolni kell, funkcionalitásban egyáltalán nem marad a ***Kontact*** hátrányos helyzetben. Előnyei egyike a ***KDE/KPart*** alapozás ami egyrészt nagyfokú



11. ábra Lokális és távoli címjegyzék és határidőnapló szinkronizálása ssh/scp biztonságos kapcsolaton keresztül



12. ábra A határidőnapló nézetben („Calendar”) az hozzáadás („Add”) gombbal egyszerre több szerverre és naptárra is feliratkozhatunk

modularitásban nyilvánul meg, másrészt nagy felhasználói bázisban, amely több visszajelzést és gyorsabb fejlesztést jelent. Fontos tulajdonsága, hogy – akárcsak a *Gnome/Novell Evolution* – nyílt forrású, *GNU GPL* licenszelésű, szabadon használható alkalmazás. Többen foglalkoztak és foglalkoznak a három *PIM* alkalmazás összevetésével. Általában arra a következtetésre jutnak, hogy mivel jelenleg még a *Microsoft Outlookon* kívül gyakorlatilag semmi nem támogatja teljes körűen a *Microsoft Exchange* szervert, ezért az erre épülő vállalati környezetekbe nem ajánlanak más *PIM* alkalmazást, minden más esetben a *Kontact* a hétköznapi használat teljes igényskáláját képes kielégíteni.

Zárszó

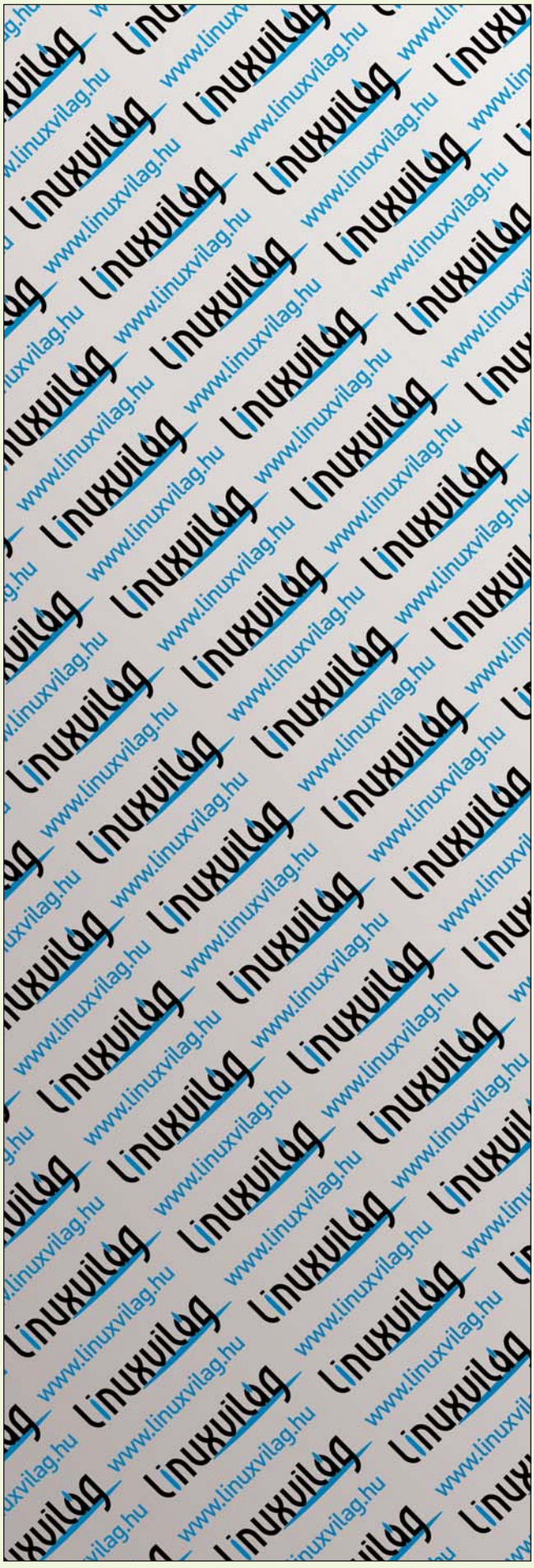
A *Kontact* használata során, noha felmerülnek néha kisebb „betegségek" – mint például az, hogy a *Summary View* elemeinek változtatása után (*Settings->Configure Summary View*) újra kell indítani a *Kontactot* a változások életbelépéséhez – összességében a *Kontact* nagyon kellemes hatást kelt. A használhatóságnak egyik alapvető oka az, hogy a beillesztett összetevőket úgy válogatták meg a fejlesztők, hogy önálló formájukban már jó ideje stabil, kiforrott és megbízható alkalmazásokat integráltak egy felület alá. A kitartó munka pedig a *KDE* asztali felület egyik legfontosabb alkalmazásává tette a *Kontacot*.
Ebben a rövid áttekintésben megpróbáltam a Kontact legfontosabb funkcióit bemutatni. Remélem sikerült kedvet adnom a kipróbáláshoz, ami talán ugyanúgy függőséget okoz majd, mint ahogy az a szerző esetében is történt.

KAPCSOLÓDÓ CÍMEK

- http://www.kontact.org
- http://www.gnome.org/projects/evolution/
- http://www-128.ibm.com/developerworks/linux/library/l-kparts/
- http://docs.kde.org/stable/en/kdepim/kontact/index.html
- http://kmail.kde.org/
- http://www.kde.org.uk/apps/kaddressbook/
- http://korganizer.kde.org/
- http://knode.sourceforge.net/
- http://akregator.sourceforge.net/
- http://www.slac.com/pilone/kpilot_home/
- http://www.kontact.org/groupwareservers.php
- http://osdir.com/Article7809.phtml

Kovács Levente (leventek@gmail.com)

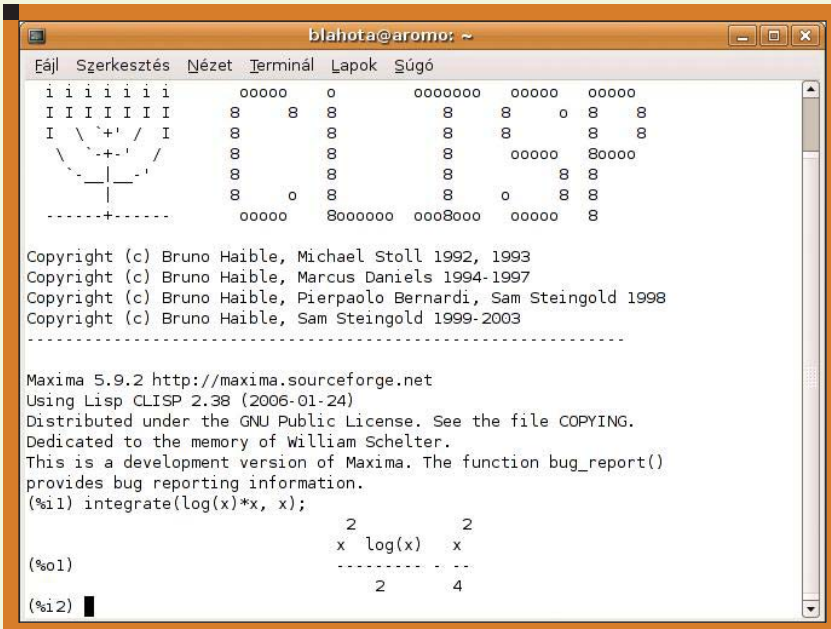
26 éves informatikus- és villamosmérnök. Évek óta használ különféle Linux disztribúciókat. Fontosnak tartja a nyílt forrású szoftverek és fejlesztés előnyeinek megismertetését az emberekkel.



Maxima – Egy nyílt forráskódú CAS alternatíva

A komputeralgebrai rendszerek segítenek a tudományban, valamint a gazdaságban és az iparban előforduló matematikai problémák megoldásában. Jó szolgálatot tehetnek viszont az oktatásban és önálló tanulás során is.

Sajátos informatikai gyakorlat az, amikor ágyúval lőnek a verébre. Két kör és egy szakasz rajzolásához piacvezető kereskedelmi szoftvert használnak, egy egyszerű hipotézisvizsgálat tanítását (amihez néha a zsebszámológép is fölösleges) már csak milliós értékű statisztikai programcsomagon tudják elképzelni. A magyarázat sokszor a „minden szoftver ingyen van” hamis és veszélyes tévhite. Ez a hozzáállás azonban hosszú távon nemigen tartható, mint ahogy ennek már számos jelét látni, bár egyelőre leginkább céges szinten és a közszférában. Tudomásul kell venni, hogy sok szoftver pénzbe kerül. Néha pedig sok pénzbe. Számos esetben találhatunk azonban jó, viszont olcsó, esetleg ingyenes alternatívát is. Így van ez a komputeralgebrai rendszerek esetén is.



1. ábra A Maxima parancssoros kezelőfelülete

A komputeralgebrai programokról általában

A komputeralgebrai program (más néven komputeralgebrai rendszer, angolul ***computer algebra system***, gyakran használt rövidítéssel ***CAS***) olyan szoftver, amely segítségével matematikai műveleteket hajthatunk végre, de nem csak számokkal, hanem algebrai kifejezésekkel, szimbolikusan is. Változókat, függvényeket, eljárásokat használhatunk és magunk is definiálhatunk. Rendkívül nagy pontossággal számolhatunk, sőt a legtöbb ilyen rendszeren belül programozhatunk is. A matematika számos területén nyújthatnak segítséget.

A komputeralgebrai programok története (így a manapság ismertebbeké is) a személyi számítógépek előtti korba nyúlik vissza. Tudományos kutatóintézetekben, egyetemeken már a ***unixos*** hőskorban is használták ilyen célokra a számítógépeket. Nem meglepő tehát, hogy az ilyen szoftverek nagy része nem szakadt el a parancssoros hagyományoktól, a grafikus felület (ha egyáltalán van neki) sokszor csak egy „bőr”, ami megkönnyítheti a használatot, de általában semmi tartalmi pluszt nem ad. Az is tipikus, hogy többségük nem csak ***DOS-Windows*** alá érhető el, hanem ***Unix-Linux*** alá is, hiszen eleve azon indult a fejlesztésük. Az első kereskedelmi ***Linux*** szoftverek között is ott találjuk a ***CAS*** rendszereket.

Megkülönböztetünk általános célú és speciális komputeralgebrai programokat.

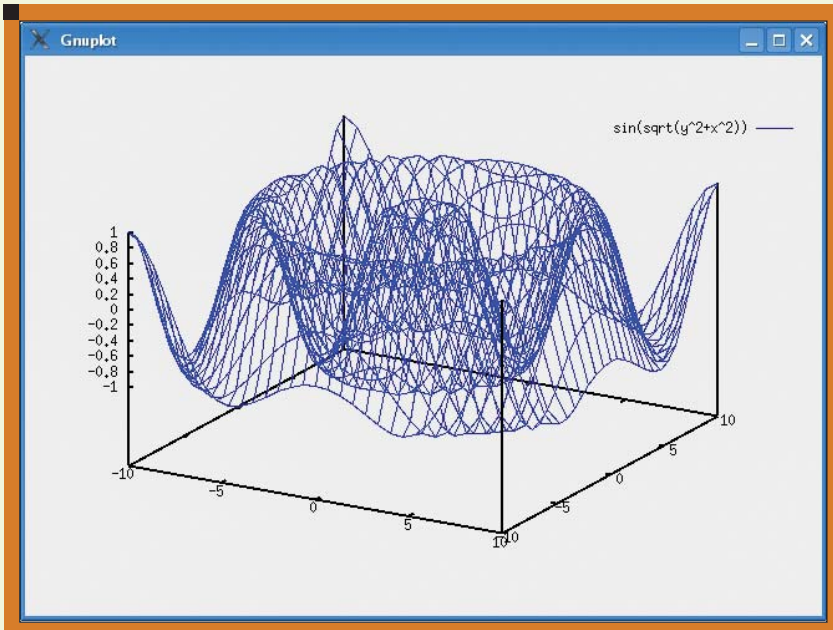
Ismertebb általános célú projektek (ABC sorrendben): ***Axiom, Derive, Mathematica, Matlab, Maple, Maxima, MuPAD, Octave, REDUCE, Scilab.*** Néhány speciális projekt: ***GAP*** (csoportelmélet), ***KANT/KASH*** (számelmélet), ***Magma*** (algebra, számelmélet, geometria).

A felsoroltak között között találunk kereskedelmi szoftvereket, de ingyeneseket is. Általában elmondható, hogy a kereskedelmi változatok drágák (áruk gyakran sok száz dollár, egy-egy teljes rendszer ára akár jóval ezer dollár fölött lehet), de barátságos grafikus felületű szoftverek, míg az

ingyenesek nyílt forráskódúak, de gyakran „fapadosak”, néha egyáltalán nincs grafikus felületük. Tudásuk azonban nem mindenben marad el drága társaik mellett.
Jómagam évekig használtam különböző *Maple* változatokat. Munkahelyem ugyan megvásárolta ezeket elegendő példányban, de hiába is tanítjuk, ha egy átlagos iskola vagy egy kisebb cég (a jelenlegi diákunk esetleges eljövendő munkahelye) nem tudja finanszírozni az ilyen összegű szoftverbeszerzéseket. Másrészt személy szerint mindig is bosszantott az az oktatásban sajnos eléggé elterjedt hozzáállás, miszerint illegális szoftverfelhasználásra biztatják, sőt olykor kényszerítik a diákokat, ezért – nem akarván rossz példát mutatni – mindenképpen más megoldás után kellett néznem.
Így az utóbbi időben kipróbáltam a *MuPAD* ingyenes és *shareware* változatait, de az új verziókból már nem készültek ingyenes, úgynevezett *Light* kiszerelések. Ekkor kezdtem el keresni valamilyen költséghatékony, lehetőleg nyílt forráskódú alternatívát, ami azért elég kiforrott és megfelel az átlagfelhasználó igényeinek (és képességeinek). Így találtam meg a *Maximát*.

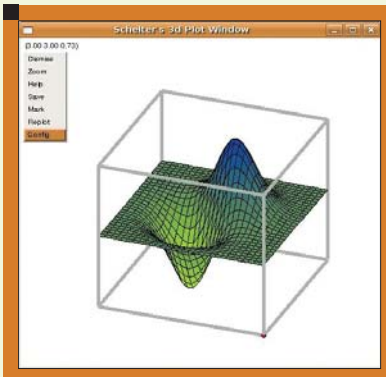
Mi is az a Maxima?

A *Maximát* eredetileg *Macsyma* néven kezdték el fejleszteni az 1960-as évek vége felé a *Massachusetts Institute of Technology*-n, *Lisp* programozási nyelven. A *Macsyma* korának úttörő programja volt, olyan rendszerek fejlesztését inspirálta, mint a *Maple* és a *Mathematica*. 1982-től *William Schelter* (ő 2001-ig állt a fejlesztések élén) vezetésével *Maxima* néven fejlődött tovább.
A *Maxima* 1998 óta *GNU General Public License (GPL)* alatt jelenik meg. Ez utóbbi tény felpezsdítette a *Maxima* és a hozzá kapcsolódó projektek fejlesztését.
Ilyen említésre méltó kezdeményezés például a *WebMathematics Interactive (WMI)* nevű, magyar fejlesztésű, web alapú interaktív matematikai szoftver fejlesztése. Ez egy nyílt forráskódú webes alkalmazás, amely több különböző komputeralgebrai rendszert (köztük a *Maximát*) is képes meghajtani. A program kipróbálható az alábbi címen: ➲ *wmi.sf.net*.



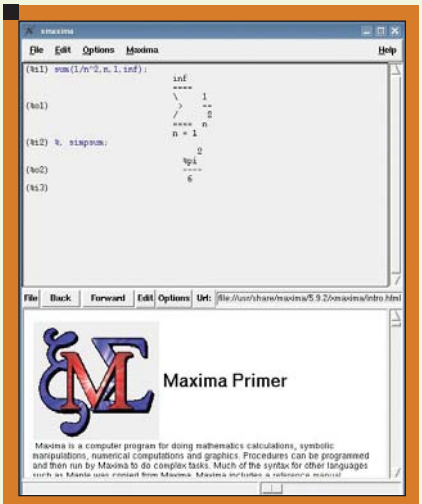
2. ábra Kétváltozós függvény ábrázolása gnuplottal,...

Hozzáláttam tehát a *Maxima* tüzetesebb vizsgálatához. Először az tűnt fel, mennyire jól dokumentált a rendszer. Például a projekt honlapjáról sok hasznos leírás tölthető le, köztük a (461 oldalas!) *Maxima Manual*, angol nyelven. Maga a szoftver is angol nyelvű, bár ez legfeljebb a hibaüzenetek olvasásánál lehet zavaró. (Mielőtt azonban valaki emiatt csalódna a *Maximában*, megjegyzem, hogy ismereteim szerint még a kereskedelmi *CAS* programok között sem találunk magyarítottat.)
Már az elérhető leírásokat átnézve nyilvánvalóvá vált számomra, hogy nagy tudású komputeralgebrai rendszerrel állok szemben, ami nagy segítséget nyújthat, leginkább a felsőoktatásban tanuló hallgatóknak.
És hogy mire is jó ez a *Maxima*? Gyakorlatilag mindenre, amire a konkurens termékek. A kalkulus jó része, a trigonometrikus és klasszikus polinomokkal, mátrixokkal való műveletek, klasszikus-, trigonometrikus-, polinomegyenletek, egyenletrendszerek és bizonyos differenciálegyenletek megoldása, nagy pontosságú aritmetika, mind-mind sajátja. A parancsok, eljárások szintaktikája egyszerű, sallangmentes és logikus. Ahogy a legtöbb hasonló rendszer esetében, ezen belül is programozhatunk.
Természetesen alkalmas egy- és kétváltozós függvények ábrázolására. Ehhez több programot is tud használni, mint például az egyszerűbb

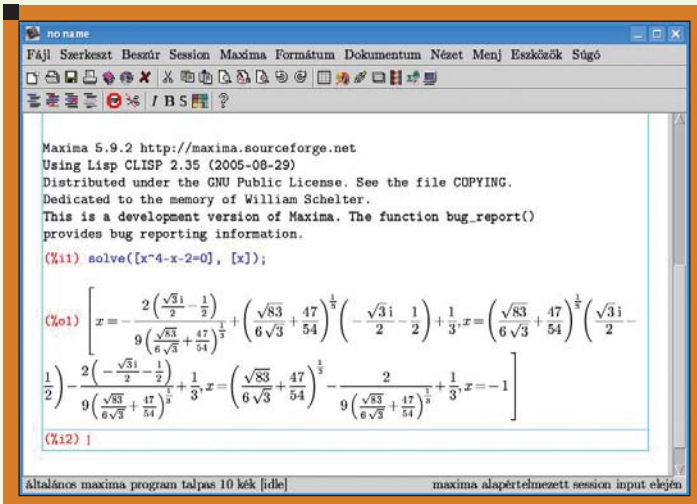


3. ábra ... és openmath segítségével

gnuplot (ez az alapértelmezés) és a nagyobb tudású *openmath* (ezeket a programokat általában a *Maxima* csomag tartalmazza), de menthetjük a képet *postscript* fájlba is.
Ha ismerünk más *CAS* rendszereket, érdemes némi időt szánni a szisztematikus összehasonlításra. Természetesen egy-egy konkrét esetben, részterületen mutathat a *Maxima* néha meghökkentő hiányosságokat, de ez a komputeralgebra – és persze a matematika – összetettségéből adódik. Akármelyik hasonló programot kezdjük „nyúzni”, csak rajtunk múlik, mikor futunk bele a korlátaiba.
A magam részéről lényegi különbséget leginkább a kényelmi funkciók terén találtam. Nem találtam lehetőséget például a *Maple*-ben megszokott úgynevezett *worksheet*-ek készítésére, bár ilyet a szintén nem olcsó *MuPAD* sem tud.



4. ábra Az xMaxima

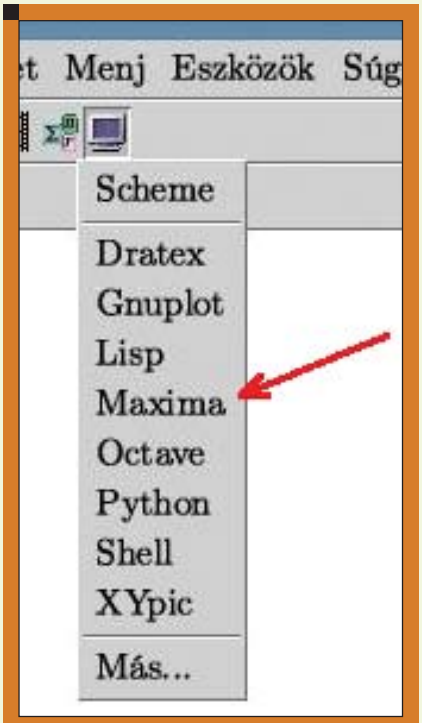


5. ábra A Maxima megnyitása a Texmacson belül

Mindezek mellett világos, hogy nem válthat ki teljesen egy professzionális *CAS* rendszert, de azok tudása sok esetben jóval meg is haladja a diákok és az egyszerű felhasználó igényeit. A tudásbeli különbség a program méretéből is sejthető, hiszen például a *Maxima* mindössze néhány megabájt, mindenesetre mérete jócskán eltörpül a kereskedelmi programok terjedelme mellett. (Természetesen ez az összehasonlítási módszer gyakran félrevezető lehet.)
A rendszer erejét mutatja, hogy hosszú évtizedek óta fejlesztett szoftver, amit aktívan fejlesztenek. A fejlesztést mi, egyszerű felhasználók is segíthetjük, hiszen honlapjáról kiindulva könnyen kezelhető hibabejelentőt találhatunk. Ugyanitt felhasználói fórum segíthet a *Maximával* való ismerkedésben. Kezdésnek azonban a *Maxima* honlapjáról elérhető kisebb-nagyobb leírások olvasgatását ajánlom.
A *Maxima* és az összes továbbiakban ismertetésre kerülő esetleges járulékos kiegészítője olyan nyílt forráskódú termék, ami egyaránt hozzáférhető *Windows* és *Linux* operációs rendszerekre is.

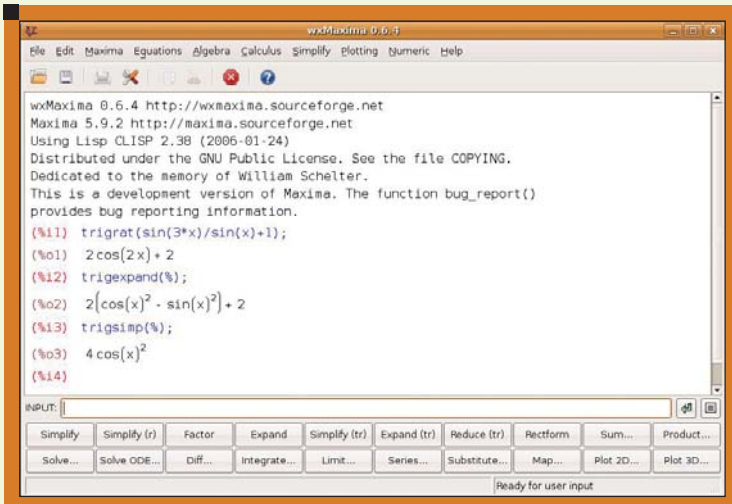
Grafikus felületek

A *Maxima* egy parancssoros program. Ez a tény bárki számára hasznos lehet, például ha valamilyen nagy teljesítményű gépen, távolról bejelentkezve futtatja a szoftvert. Ez azonban manapság nem túl gyakori. Bár rendelkezik némi kényelmi szolgáltatással – a korábban beírt parancsok között



6. ábra A Texmacs

a fel és le kurzorvezérlőkkel böngészhetünk – nem használható vele a „másolás-beillesztés” gyakran hasznos funkciója.
Mivel gyakran találkozhatunk – leginkább a *Linuxszal* ismerkedők körében – azzal az jelenséggel, hogy ha a felhasználó nem lát grafikus felületet, hajlamos pánikba esni, így a fejlesztők jóvoltából a *Maximához* alapból tartozik egy *xMaxima* nevű (elég kezdetleges és – mi tagadás – nem túl kényelmes) grafikus felület. Előnyös tulajdonsága, hogy indítás után az ablak alsó felében rögtön láthatunk egy kivonatos leírást néhány alapvető funkciójáról, példákkal, linkekkel.
Mivel a *Maxima* nyílt forráskódú program, nincs elvi (és jogi) akadálya, hogy bárki kiegészítő-szoftvereket készítsen hozzá. A projekt oldalán számos ilyen kezdeményezés honlapját találhatjuk belinkelve. Ezek közül tehát kettőt mutatnék be.
A *TeXmacs* egy sokoldalú, *WYSIWYG* típusú szövegszerkesztő speciális funkciókkal és pluginokkal. Kifejezetten képleteket tartalmazó, matematikai szövegek esetén lehet hasznos. A *TeX* karaktereit használja, lehetőség van importálni, exportálni *LaTeX*-be. A mi szempontunkból azért érdekes, mert bár alapvetően nem a *Maximához* készítették, együtt tud működni vele, *LaTeX*-es karaktereket megjelenítve. További kellemes tulajdonsága, hogy a korábban használt képletekhez egyszerűen (a kurzorvezérlő billentyűkkel, vagy az egérrel) visszatérhetek és módosíthatom azokat. Kezelőfelületét több nyelvre lefordították, így magyarra is.
Kezdőknek azonban leginkább a *wxMaxima* nevű programot ajánlom. Ennek a szoftvernek egyetlen célja van, hogy kényelmes, grafikus kezelőfelületet nyújtson a *Maxima* használatához.
Lényegesen felhasználóbarátabb *frontend*, mint az *xMaxima*. A gyakrabban használt függvényeknek gombjai vannak, számos további témakörök szerint csoportosított menükből érhető el, akár kezdhetjük a *Maximával* való ismerkedést ezek tanulmányozásával, próbálgatásával.



7. ábra Az Ubuntu Dapper Drake-ben lévő wxMaxima 0.6.4

A korábban beírt parancsok között a fel és le kurzorvezérlőkkel böngészhetünk és az ily módon „visszanyert” képleteket szerkeszthetjük. További könnyebbség még, hogy ha kinyitunk egy (bármilyen típusú) zárójelet, a bezárása automatikusan megjelenik. A képernyőn megjelenített munkafolyamatokat képes *HTML*-be exportálni. Gyors és megbízható program. Súgójában megtalálható a teljes *Maxima Manual*, és egy másik, tömörebb leírás is. Kényelmesebben használható mint a *TeXmacs*, hátránya viszont a nem *LaTeX*-es kimenet és hogy „nem tud magyarul”. (Ha valakinek van szabad kapacitása és kedve, szép és hasznos feladat lenne elkészíteni a magyarítását.)

A programok telepítése

A legtöbb disztribúcióhoz találunk csomagokat, legfeljebb nem a legfrissebb verzióból készülteket. Terjesztéstől függhet az is, hogy egy program egy csomagot alkot, vagy az egyes alkotóelemeket több darabra szétszedve csomagolták. Igény (vagy szükség) esetén a programokat forrásból is fordíthatjuk, bár ez a függőségek miatt nem mindig egyszerű.
Először telepítsük a *Maximát.*
Ha nincs csomagunk, letölthetjük a forrást a honlapjáról. A cím: ➲ maxima.sourceforge.net. A *Maxima* elérhető *Windows* és *Linux* operációs rendszerekre, továbbá (a leírás szerint) forrásból fordítható még *MacOS X*-re is.
A jelenleg letölthető legfrissebb stabil verzió az 5.9.2, valamint publikus még az 5.9.3-as fejlesztői verzió. Ez utóbbi tesztelése során elég sok – az előző verzióban nem jelentkező – hibába botlottam, így nem ajánlom „éles” használatra.
Ha a *Maxima* telepítésével megvagyunk, javaslom szerezzük be a *wxMaximát,* esetleg a *TeXmacsot*. Az esetleges bonyodalmak elkerülése végett itt is ajánlom, részesítsük előnyben a disztribúciónkhoz tartozó csomagolt verziókat. További információkért érdemes körülnézni honlapjaikon, ahonnan kiindulva forráskódjaikhoz is hozzájuthatunk. Ezek a következőek: ➲ *www.texmacs.org*, ➲ *wxmaxima.sourceforge.net*.
A *TeXmacs* legújabb stabil *Linuxos* verziója az 1.0.6. A *wxMaxima* elérhető legfrissebb változata a 0.6.5.

Bonyodalmak Ubuntu 6.06 alatt

Már-már azt írtam, hogy a programok telepítése problémamentes (jómagam eleinte csak *SUSE* és *Slackware* alatt használtam), amíg ki nem próbáltam *Ubuntu Dapper Drake*-kel. Mivel népszerű disztribúcióról van szó, leírom az ott tapasztaltakat.
Telepítettem a *wxMaxima és Maxima* csomagokat és először úgy tűnt, minden rendben van. Külön-külön működött mindkét program, de a *wxMaxima* nem akart együttműködni a *Maximával*. Némi *Internetes* keresgélés után világossá vált, hogy a „hiba nem az én készülékemben van”; valaki már talált is egy lehetséges megoldást a problémára. Így jártam el én is. Feltelepítettem a *clisp* csomagot, majd ezzel lefordítottam az időközben forráskódban letöltött *Maximát. A* fordítás (kicsomagolással együtt) az alábbi megszokott részekből állt:

```
$ tar -xzvf maxima-5.9.2.tar.gz
$ cd maxima-5.9.2
$ ./configure
$ make
```

Annyi volt még hátra, hogy a *wxMaxima* konfigurációjában (*Edit->Configure->Options->Maxima program:*) megadtam, hogy az így keletkezett *maxima-local* fájlt használja a csomagból feltelepített *Maxima* helyett. (Az eredeti *Maxima* csomagot nem távolítottam el, mert a függőségek miatt ez maga után vonta volna a *wxMaxima* törlését is.) Ezután minden rendben volt.
Az eredményt az 7. ábrán láthatjuk.
Az *openmath* működéséhez viszont még egy *Dapperre* jellemző hibát ki kell javítanunk. Írjuk át az */etc/X11/xorg.conf* fájlban az

```
RgbPath
"/usr/X11R6/lib/X11/rgb"
```

sort az alábbira:

```
RgbPath "/etc/X11/rgb"
```

Ez persze más *„Tk*-s” programunkra is „jótékony hatással” lesz.
Azt is jó tudni, hogy függvények *gnuplottal* való ábrázolásához *Ubuntu* alatt azonban szükség van még a *gnuplot-x11* csomag telepítésére is.

Összegzés

Általában elmondható, hogy a *Maxima* – kiegészítve például a *wxMaximával* – vállalható kompromisszumnak tűnik, könnyen kezelhető, jól használható, értelmes alternatívája lehet drága kereskedelmi programoknak.

Blahota István
(blahota@nyf.hu)

Matematikus és tanár, a Nyíregyházi Főiskola docense. Matematikai analízisből szerzett PhD fokozatot. Öt éve a Linux elkötelezett híve.

CUPS

Gutenberg nyomában...

Elemzők a 90-es években papírmentes irodát jósoltak. Nos az emberiség régi vágya – a papírra rögzítés miatt – ez még egy darabig váratni fog magára annak ismeretében, hogy az egyre olcsóbb nyomtatási lehetőségek emelték a papírfogyasztást. Talán az elektronikus papírok megoldják majd ezt a problémát.

Milyen nyomtatót válasszunk?

Mára a *Linux* nyomtatási lehetőségei igen sokat fejlődtek a kezdetiekhez képest. Sajnos azonban még mindig van olyan gyártó, amelyik nem figyel rá, hogy a nyomtatója *Linux* alatt is használható legyen. Számos olyan nyomtatót láttam eddig, ami csak *Windows* alatt ment különösebb trükkök nélkül. Persze az ilyen nyomtatókat is működésre lehet bírni de megvételük nem ajánlott. Amennyiben nyomtatót szeretnénk vásárolni, a támogatott nyomtatók listájáért látogassunk el a *LinuxPrinting.org* honlapjára (link a cikk végén), vagy hagyatkozzunk a jól bevált mondásra, miszerint amelyik nyomtató tudja hardveresen a *PCL* és/vagy a *PostScript* formátumokat, azzal nagy gondunk nem lehet.
A *LinuxPrinting* honlapon rákeresve a nyomtatóra, vannak olyan nyomtatók, ahol kompromisszumot kell kötni a nyomtatásért, pl. nem lehet kihasználni a legnagyobb felbontást, vagy mondjuk *Linux* alól nem lehet színesen nyomtatni vele.

Tehát *Perfectly* vagy *Mostly* jelzőt kapnak azok a nyomtatók, amelyek Linux alatt probléma nélkül vagy kevés problémával működnek. Jómagam egy *Okipage 14ex* nyomtatót használok, ami szinte az összes létező módon kapcsolódni tud a számítógéppel. Van rajta *párhuzamos port, USB*, sőt *hálózati csatoló* is. Ennek nagy előnye, hogy olyan géppel is használható, amelyen már foglalt az két népszerűbb port, illetve nem kell egy gépnek folyton bekapcsolva lennie, hogy a hálózaton bármelyik gépről használhassuk.

Kernelfordítás

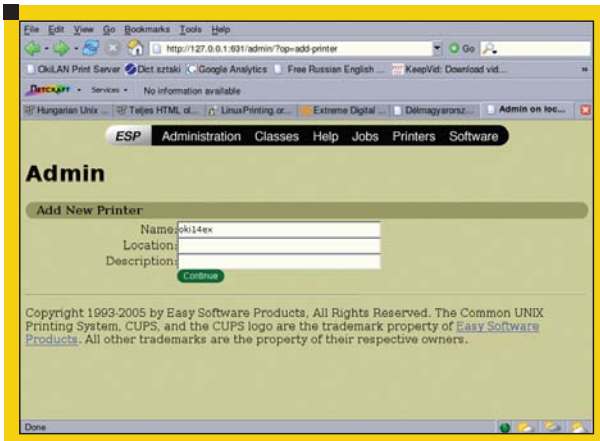
Talán eddig még nem volt olyan cikkem, amelyben ne kellett volna kernelt fordítani. Nos most is csak akkor kell, ha pár dolog nincs a kernelben, minden szükséges dolog a *Device drivers* részben található.
Ha egy USB nyomtató rendesen működik, akkor a `dmesg` által kiadott adatok között kell szerepelnie, illetve az `lsusb` paranccsal látnunk kell. Amennyiben nem látjuk, nézzük meg,

1. ábra Perfectly, azaz zöld jelzés. Kezdhetjük a telepítést.

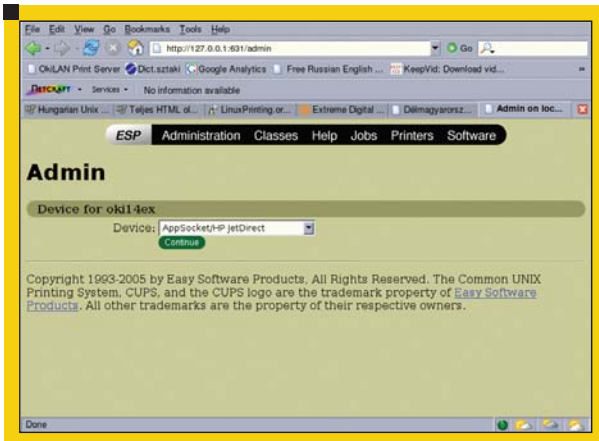
hogy kernelbe lefordítottunk-e mindent rendesen, illetve megfelelő *USB alrendszert* választottunk-e: *OHCI* / *EHCI* / *UHCI*.

CUPS telepítése

A legtöbb disztribúció tartalmazza a *CUPS*-ot előre fordított csomagként. *Debian*ban pl. az alábbi csomagok szükségesek: *cupsys, cupsys-bsd, cupsys-client* (ez utóbbi a nyomtatási parancsokért felelős).
Ami még fontos lehet, legalábbis sok kellemetlenségtől kíméljük meg



2. ábra Első lap, itt szinte elég egy sort begépelni



3. ábra Milyen típusú a nyomtató kapcsolódási felülete

magunkat, ha felrakjuk, a ***FooMatic*** csomagokat, debianban névszerint: ***foomatic-db, foomatic-db-en, foomatic-filters, foomatic-filters-ppds***.
Még szükség lehet – bár ez a nyomtatótól függ – a ***foomatic-db-gimp-print*** és a ***foomatic-db-hpijs*** csomagokra is.

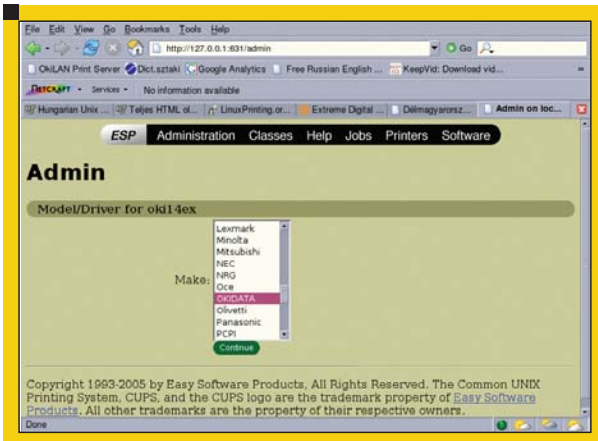
A CUPS beállítása

Nyissunk meg egy böngészőt és hozzuk be a ➲ ***http://127.0.0.1:631*** oldalt. Ez a ***CUPS nyitó oldala***, itt végezhetünk a legegyszerűbben bárminemű beállítást. A felső menüsor magáért beszél. Az ***Administration*** gombra kattintva meg kell adni a root jelszót. Ezután már bármit csinálhatunk. Kijelölhetünk osztályokat, törölhetünk beragadt nyomtatási feladatokat, illetve új nyomtatókat vehetünk fel, vagy a meglévőeket módosíthatjuk, törölhetjük.
Nézzünk egy konkrét példát!
A ***Printers*** menüben az ***Add Printer*** gombbal adhatunk meg új nyomtatót. Az első lapon megkérdezi a nyomtató nevét – ezen a néven fogjuk tudni elérni a programokból –, illetve a helyét és egy rövid leírást. Ez a két mező inkább csak nagyobb szervezeteknél jön jól, ahol sok nyomtatót kell felügyelni.
A második oldalon megkérdezi, hogy milyen protokollon/porton szeretnénk használni. Válasszuk ki a nekünk megfelelőt. Fontos: a ***CUPS*** csak azon csatlakozási lehetőségeket ajánlja fel, amelyek támogatottak az aktuális kernelben. Például, ha a párhuzamos portot nem ajánlja fel, úgy valószínűleg nem megfelelő a kernel.
Ha a nyomtató rendelkezik ***saját hálózati csatolóval***, akkor az ***AppSocket/HP Jetdirect***-et válasszuk.
A harmadik lapon megkérdezi a ***nyomtató gyártóját***. Ha kevés gyártó látható, akkor nem települtek rendesen a ***foomatic*** csomagok. Amennyiben kevés a merevlemezen a helyünk, úgy megtehetjük, hogy csak a számunkra fontos nyomtató meghajtót töltjük le a ***LinuxPrinting*** oldalról. Ez esetben a ***ppd*** fájlt a ***/usr/share/cups/model*** könyvtárba helyezzük el és indítsuk újra a ***CUPS***-ot:

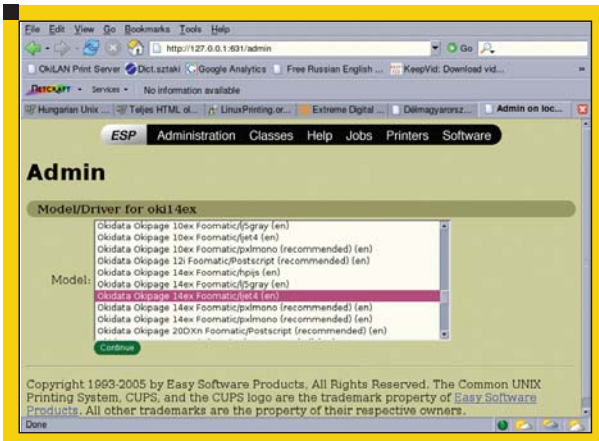
```
/etc/init.d/cupsys restart
```

A negyedik lapon ***konkretizálhatjuk a nyomtató típusát***. Nem biztos, hogy az ajánlott (recommended) nyomtatómeghajtó a legjobb. Nálam pl. az ajánlott bizonyos esetekben – például fénykép nyomtatásnál – sokkal lassabb, mint a ***LaserJet 4***-es emuláció. Ezzel persze még nem ért véget a nyomtató beállítása. A ***Printers*** fülre kattintva bökjünk a ***Configure Printer*** gombra. Itt lehet beállítani például a ***felbontást***, a ***lapméretet***, hogy ***melyik tálcából*** vegye a papírt, illetve pár nyomtató specifikus dolgot (pl. ***van-e lapfordító*** a nyomtatóban).
Nos ha úgy érezzük, nem felejtettünk ki semmit, akkor ***nyomtassunk*** egy ***tesztoldalt*** a ***Print Test Page*** gombbal. Ha mondjuk csak a ***tesztoldal fele látszik***, akkor valószínűleg ***kevés*** a nyomtatóban a ***memória***, próbáljuk meg

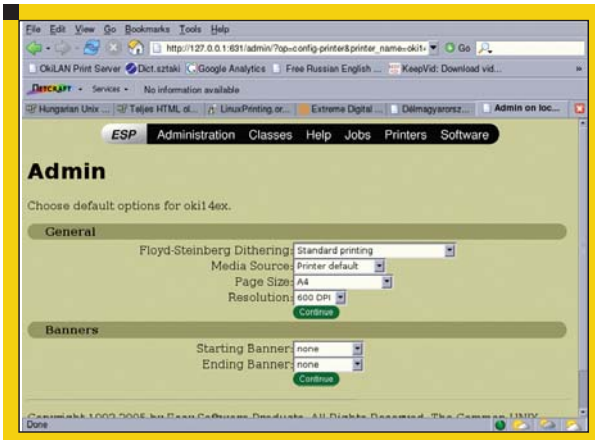
A kapcsolat módja	Szükséges beállítások
Párhuzamos portra kapcsolódó nyomtató	-> Parallel port support -> (itt szinte minden)-> Character devices -> Parallel printer support
USB kapura kapcsolódó nyomtató	-> USB support, illetve ezen belül UHCI, OHCI vagy EHCI, az alaplapnak megfelelően. Ha van működő pendrive vagy más USB-s eszközünk Linux alatt, akkor ez a rész rendben van. Illetve ezenkívül szükség lesz még az USB Printer support-ra is.
Hálózati nyomtató	Szükséges egy működő hálózati kapcsolat, de ez valószínűleg már megvan.
Sambán keresztül megosztott hálózati nyomtató	Amennyiben hagyományos Samba megosztásként szeretnénk kiajánlani egy nyomtatót – legyen az bármilyen a fenti háromból –, úgy szükség lehet a fájlrendszerek közül a SMB fájlrendszer támogatására mind kernel oldalon, mint userspace oldalon (például smbclient).



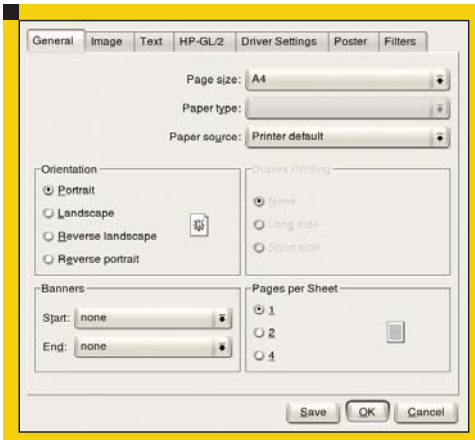
4. ábra A nyomtató típusa



5. ábra A nyomtató típusa, illetve az emuláció beállítása



6. ábra Finomhangolás



7. ábra A KDE-alapú nyomtatást segítő előtét

csökkenteni a *felbontást* – vagy szerezzünk a nyomtatóba memóriát, amennyiben bővíthető. Ha ezenkívül bármi probléma adódnak, a *CUPS* log fájlját (*/var/log/cups/error_log*) kell átnézni és vagy triviális a hiba – nem talál valamilyen programot –, vagy pedig a *hibaüzenetet a Google-ba bemásolva* nagy valószínűséggel *megkapjuk* a *megoldást*.

Feladatok törlése

Ami elromolhat, az el is romlik – szokta *Murphy* mondogatni. A nyomtatónak is néha lehet rossz napja, például olyan kimenetet küldünk rá, amit valamiért nem tud feldolgozni. Ekkor *törölnünk* kell a *nyomtatási sorból*, ezt a már korábban említett *Job fülön* érhetjük el.
Amennyiben parancssorból dolgozunk, úgy `lpq`-val kérdezhető le a feladatok listája, illetve törlésre az `lprm` parancsot használjuk.
Persze akár parancssorból, akár webes felületről dolgozunk, mindkét esetben *csak a saját nyomtatásunkat töröl-hetjük*, egyébként szükséges a rendszergazdai jogosultság.

Osztályok

A *CUPS* kezel osztályokat is. Ezzel például megvalósítható, hogy bizonyos nyomtatókat egyszerre tiltsuk a feladatok fogadását, illetve egyszerre nyomtassunk tesztoldalt. Gyakorlati haszna a hétköznapi felhasználónak nincs, de egy nagyvállalati környezetben – ahol sok nyomtató van – hasznos lehet, pl. osztályonként, tanszékenként csoportba gyűjteni a nyomtatókat.

Nyomtatás KDE és Gnome alatt

Noha jómagam nem használom egyik ablakkezelőt sem, fontos megemlíteni, hogy mindkettő rendelkezik nyomtatást elősegítő eszközökkel.
KDE alá pl. a `cupsdconf` remek alkalmazás, amely a *kdelibs-bin csomagban* foglal helyet. Ez főképp a *CUPS szerver finomhangolását* teszi lehetővé, a nyomtatók hozzáadásában, beállításában nem tud segíteni.
Ami a hétköznapi embert nagyban segíti az a `kprinter`. Ez menetközben alakítja a dokumentumot. Világosít, sötétít vagy akár poszter módban nyomtat, de amint a mellékelt képernyőképek is mutatják, igen széles skálán kalibrálható a program.
Gnome sem nélkülözi az ilyen jellegű programokat. A `gnome-cups-add` például azokat célozza meg, akik nem szeretik a webes nyomtató telepítést, ugyanis ez a program sokkal inkább egy *Windows* alatti nyomtató telepítésre hasonlít. A másik hasznos program a `gnome-cups-manager`. Ezzel pl. a meglévő nyomtatók *nyomtatási sorait* tarthatjuk szemmel a parancssor vagy a webböngésző használata nélkül.

Kvóta

Amikor az ember nyomtatót ad a munkatársai vagy a családtagjai kezébe, könnyen azon kaphatja magát,

8. ábra Ugye mennyire hasonlít a Windows-os nyomtató telepítésre?

1. Lista Nyomtatási kvóta beállítás (a /etc/cups/printers.conf részlete)

```
<DefaultPrinter oki14ex_ps>
Info
Location
DeviceURI socket://
➥192.168.1.200:9100
State Idle
Accepting Yes
JobSheets none none
QuotaPeriod 604800
PageLimit 100
KLimit 1024
</Printer>
```


hogy idő előtt elfogyott a papír és/vagy a festék. Szerencsére azonban a ***CUPS*** támogatja a ***kvótázást***. Sajnos azonban ***nem elég komoly*** ez a kvótarendszer ahhoz, hogy ***felhasználókhoz rendelhető*** kvóta legyen, csak nyomtatónként lehet állítani a kvótát egy általunk megadott időtartamra. Ezt a */etc/cups/printer.conf* fájlban tehetjük meg. A megfelelő nyomtató szekciójába helyezzük el a ***QuotaPeriod*** kulcsszót, ami a másodperceket jelöli, tehát: egy nap esetén 86400, egy hét esetén 604800. A ***PageLimit*** kulcsszó pedig a kvótaidőszak alatt maximálisan nyomtatható oldalak számát adja meg, míg a ***KLimit*** a nyomtatható méretet kilobájtban. Ha ***0 szerepel*** mindenhol, akkor ***nincs aktív kvótázás***. Miután ezt módosítottuk, ne felejtsük el újraindítani a ***CUPS***-ot.

Samba megosztás

Amennyiben otthon van pár számítógépünk, érdemes ***Sambaval*** megosztani a nyomtatónkat, hogy a családtagok is tudjanak nyomtatni, akár ***Windows*** alól is. Persze a ***sambás*** megoldásokat egyre inkább háttérbe szorítják a ***router***ek, amelyekben van ***beépített nyomtatószerver***.
A `print$` szekcióban elhelyezhetjük (persze nem kötelező) a ***Windows***-os meghajtó programokat, így a ***Windows*** nagy valószínűséggel nem fog telepítőlemezt kérni a megosztott nyomtatóhoz.

Ha nem nyomtat...

Gyakran futunk érthetetlen hibákba, de megosztok most párat, ami esetleg megkönnyíti a hibakeresést.
Helyi nyomtatóknál (USB, párhuzamos):

- Van-e elég hely a */tmp*-ben?
  A korábban említett beállító állományok mindegyike a */tmp* könyvtárat használja a munkához.
  Van-e elég helyünk?
  Nincs-e kvótázva a könyvtár?
  Egyegy nyomtatási feladat akár 20-30 megabájt is lehet.
- Van-e jogunk írni a */tmp*-be?
- Nem szakadt a nyomtató kábele?

Hálózati nyomtatóknál (***Samba***, egyéb):

- Nem fogja-e meg a tűzfal akár lokálisan, akár másképp a nyomtató – vagy a nyomtatót kezelő számítógép – felé irányuló forgalmat?
- Esetleg itt is gyanakodhatunk kábel hibára.

Remélem sikerült felkelteni az olvasó érdeklődését, persze nem győzöm hangsúlyozni, hogy ***Linux*** alatt csak megfelelő nyomtatóval érhetünk el sikerélményeket, és lehet, hogy az olcsó nyomtató később bosszúságok forrása lesz. Tehát mint minden hardver beszerzéskor, most is legyünk körültekintőek.

2. Lista Minimális Sambával történő nyomtatómegosztás (/etc/samba/smb.conf a nyomtatásra vonatkozó részlet)

```
[print$]
    comment = Printer Drivers
    path = /var/lib/
    ➥samba/printers
    browseable = yes
    read only = yes
    guest ok = no

[oki14ex]
    comment = OKI14EX
    path = /tmp
    public = yes
    guest ok = yes
    printable = yes
```

Medve Zoltán
(e-medve@e-medve.hu)

2001-ben kezdett „Linuxolni”, de már korábban is ismerkedett a szabad szoftverek világával.
Ha éppen nem a gép előtt ül, akkor fotózgat, olvasgat vagy bicajozik.

KAPCSOLÓDÓ CÍMEK

LinuxPrinting
➲ http://www.linuxprinting.org

Építsünk bioinformatikai célú szuperszámítógépet

Az OSCAR fürtöző környezetben futó bioinformatikai eszközök segítségével 17 újrahasznosított személyi számítógépünket felhasználói lekérdezések teljesítményét növelő rendszerré változtattuk.

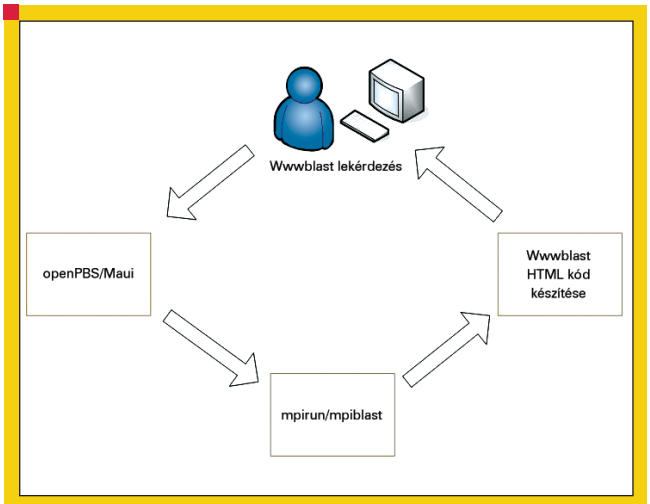
A bioinformatika egyre növekvő jelentőségű tudományterület, melynek egyik ágazata a *DNS* és fehérjeszekvenciák elemezése. A *National Center for Biotechnology Information (NCBI)* által kifejlesztett *Basic Local Alignment Search Tool (BLAST)* az ilyen sorozatok elemzésében segíti a tudósokat. Az eszköz nyílt változata a hálózaton elérhető vagy letölthető. Tekintve, hogy a *BLAST* weblapja igen népszerű, teljesítménye legalábbis hullámzónak nevezhető. A *Dél-Dakotai Egyetem (USD) Bioinformatikai Számítástudományi Csoportja* úgy döntött, hogy nyílt forrású eszközöket felhasználva megpróbálja kialakítani a *BLAST* párhuzamos változatát egy *Linux* fürtön.
A *BLAST* fürt amelybe kivénhedt asztali személyi számítógépek kerültek, felgyorsítja a kereséseket azáltal, hogy a kutatók kis csoportja számára naprakész adatbázist biztosít.
A fürt megoldásunk alapozását az *Open Source Cluster Application Resources (OSCAR)* alkalmazással kezdtük. Az *OSCAR* rendszert az *Open Cluster Group* tervezte fürtözött számításokhoz azzal a céllal, hogy a *Linux* alatti fürtkészítéshez valamennyi eszköz egyetlen csomagban elérhető legyen. Az *OSCAR* segít leegyszerűsíteni a telepítést, a karbantartást, sőt még a fürtprogram használatát is. A grafikus felhasználói felület lépésenként végigvezet bennünket a folyamaton majd később grafikus karbantartó eszközként használható.
Az *NCBI* által készített *WWW BLAST* web alapú kezelőfelület, megkönnyíti a *BLAST* felhasználók dolgát.
Ezt a rendszert választottuk mi is a *BLAST* fürtünkhöz.
A *WWW BLAST* könnyedén telepíthető minden webkiszolgálót (például *Apache*) futtató *Linux* gépre.
Míg a *WWW BLAST* a fürt használati értékét növeli meg, az *mpiBLAST* a teljesítményt fokozza.
Az *mpiBLAST* a *Los Alamos National Laboratories (LANL)* fejlesztése, melynek célja a *BLAST* teljesítményének növelése a lekérdezések párhuzamos végrehajtása által. Az *mpiBLAST* szíve a párhuzamos programokhoz gyakran használt az *üzenetközvetítő felület (Message Passing Interface, MPI)*. Az *mpiBLAST* minden programfeltételt biztosít számunkra a párhuzamos *BLAST* lekérdezések futtatásához.

Lekérdezések áttekintése

Fürtünkön a *BLAST* keresés a webalapú űrlappal indul. Alapértelmezés szerint a *WWW BLAST* nem támogatja a kötegelt végrehajtást és a feladatütemezést. Szerencsére az *OSCAR* csomag *OpenPBS* és a *Maui* eszközei képesek megbirkózni a feladatütemezés és a terhelésmegosztás problémájával. Ilyen támogatással a fürt már sokkal könnyebben ki tud szolgálni nagyobb közönséget is.
Az *OpenPBS* egy rugalmas köteg sorkezelő eszköz amelyet eredetileg a *NASA* számára fejlesztettek ki. A *Maui* az *OpenPBS* képességeit bővíti kiterjedtebb folyamatvezérlést és ütemezési házirendeket nyújtva.
Miután a felhasználó kiadta a lekérdezést, a *WWW BLAST Perl* parancsfájlja indul el. Ez a parancsfájl az űrlap adatai alapján egyedi munkafolyamatot hoz létre. A munkafolyamat egy program vagy feladat amely az *OpenPBS*-hez kerül végrehajtásra. A munkafolyamat átadása után az *OpenPBS* meghatározza a csomópontok elérhetőségét majd az ütemezései rendszabályok alapján végrehajtja a munkafolyamatot. A munkafolyamat elindítja a *helyi elérésű multiszámítógép (Local Area Multicomputer, LAM)* programot, amely lényegében egy felhasználói szintű, démon alapú, futásidejű környezet. A *LAM* az *OSCAR* telepítés része és *MPI* programok legtöbb szolgáltatását tartalmazza.
Az *OpenPBS* a munkafolyamatot az `mpirun` parancs segítségével indítja el, amely minden csomóponton elindítja a keresést, majd összegyűjt az eredményt. A *WWW BLAST* az eredményeket visszaküldi a böngészőnek, majd a felhasználó számára jól olvasható jelentést készít (*1. ábra*).
A párhuzamos *BLAST* kereséseket megvalósító fürtözési megoldás kialakítása némi programmódosítást igényelt. Sok általunk használt eszköz az alapértelmezett beállításokkal működik, ám a párhuzamos *BLAST* fürt már egyedi beállításokat igényel.

Fürtépítés az OSCAR-ral

A fürtöket többféle személyi számítógépből is elkészíthetjük. A mi 17 csomópontunk 533MHz-es *Intel Celeron* processzorral, 256MB RAM-al és 15GB merevlemezzel rendelkezett, ami mai viszonylatban alacsony kategóriásnak számít. A fürtözés kiépítésekor nem követelmény, hogy minden csomópont teljesen azonos alkatrészekből álljon, de mindenképpen



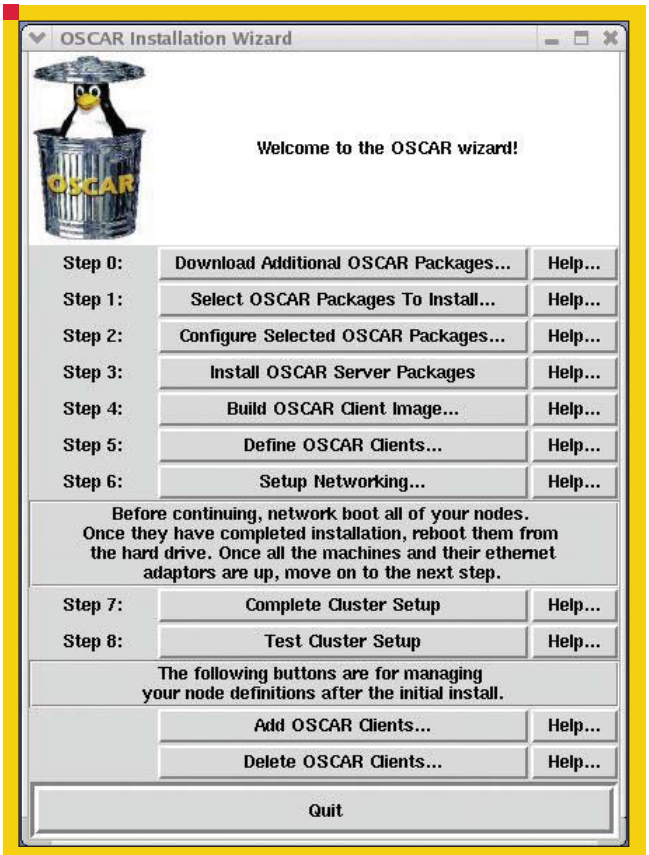
1. ábra Amikor a Webről egy lekérdezés érkezik, a WWW BLAST a munkafolyamatot átadja az OpenPBS-nek. Az OpenPBS az mpirun segítségével elindítja azt, majd a WWW BLAST megformázza a végeredményt.

csökkenti a fürt előkészítésére és karbantartására fordított időt és energiát. Miután a gépeket előkészítettük, ki kell választanunk a fejcsomópontként üzemelő gépet. Amennyiben nem azonos gépeket használunk, nem árt ha a legerősebbet tesszük meg fejcsomópontnak. Tekintve, hogy mi teljesen egyforma gépeket használtunk, véletlenszerűen választottuk ki a fejcsomópont gépet.
Miután beszereztük a szükséges gépi alkatrészeket, ***Linux*** terjesztést kell választanunk. Az ***OSCAR*** dokumentációja felsorolja valamennyi támogatott operációs rendszert, mi a ***Red Hat 9.0***-ás változatot választottuk. A ***Red Hat*** telepítése igen egyszerűnek bizonyult, mindenhol az alapértelmezett beállításokat választottuk. Minthogy az ***OSCAR*** program adott OS csomagoktól függ, az operációs rendszer frissítése nem javasolt az összeállítás után. Természetesen ennek komoly biztonsági vonzatai lehetnek, pontosan emiatt kell a fürtünket az internettől tűzfallal elválasztanunk.
Miután a ***Red Hat*** rendszert feltelepítettük a fejcsomópontra, letöltöttük az ***OSCAR 2.3.1 tar*** csomagját. A telepítési útmutatót a hálózati források között megtaláljuk. Az ***OSCAR*** állományát a root felhasználó saját könyvtárába töltöttük le, hiszen az ***OSCAR***-t root-ként kell majd telepíteni.
Az ***OSCAR*** telepítése mindössze a következő parancsok futtatásából állt:

```
tar -xvfz oscar-2.3.1.tar.gz
cd oscar-2.3.1
./configure
make install
```

A telepítés után a fejgépen valamennyi ***Red Hat 9.0*** RPM-et át kellett másolnunk a ***/tftpboot/rpm*** könyvtárba. Az ***OSCAR*** telepítő bizonyos csomagokat ebből a könyvtárból tölt le a telepítési folyamat során. A következő parancsokkal másoltuk át az állományokat:

```
cp /mnt/cdrom/RedHat/RPMS/*.rpm /tftpboot/rpm
```



2. ábra Az OSCAR segít elhelyezni, beállítani és tesztelni a fürtprogramot

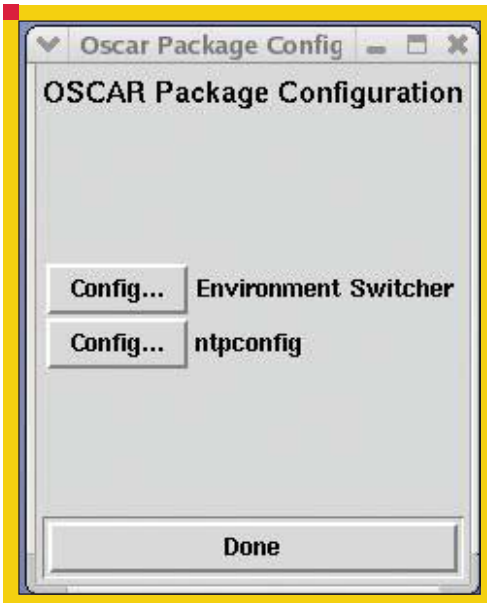
Miután valamennyi RPM-et átmásoltuk, nekifoghatunk az ***OSCAR*** telepítésének. Az ***OSCAR*** grafikus telepítővarázslóval segíti a munkánkat. Helyettesítsük be saját belső hálózatunk ***Ethernet*** csatolóját; a mienk az `eth1` volt:

```
cd $OSCAR_HOME ./install_cluster eth1
```

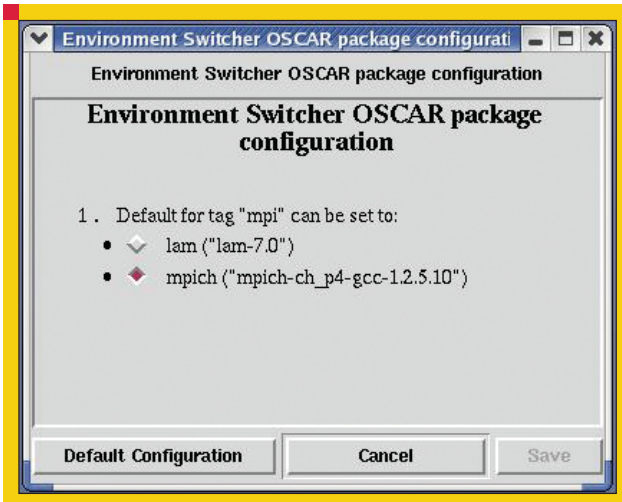
Néhány másodperc múlva az ***OSCAR*** telepítővarázsló megkezdi a betöltést. A varázsló grafikus felületen nyolc lépéses folyamaton végigvezetve segít meghatározni a fürt beállításokat (*2. ábra*). A telepítésünkben az egyetlen eltérés az volt, hogy az alapértelmezett ***MPI*** megvalósítást ***LAM/MPI***-ra állítottuk az ***MPICH*** helyett. Azért választottuk a ***LAM***-ot, mert az ***mpiBLAST*** helyes végrehajtásához erre volt szükségünk. A 2. pontra ***(Configure Selected OSCAR Packages)*** kattintva egy kis űrlaphoz jutunk (*3. ábra*). Itt választhatjuk ki az ***Environment Switcher*** gombot és állíthatjuk be a ***LAM*** rendszert a telepítés alapértelmezettjeként (*4. ábra*).
A további lépéseket az ***OSCAR*** dokumentációnak megfelelően követtük és elkészítettük a csomópontokra szánt telepítőlemezeket. Miután minden csomópontot feltelepítettünk és kipróbáltuk a rendszert, letöltöttük és feltettük az ***mpiBLAST***-ot.

Az mpiBLAST telepítése párhuzamos keresésekhez

Letöltöttük a ***mpiBLAST***-ot, és a ***README*** állományban olvasható dokumentációnak megfelelően telepítettük.
Az ***mpiBLAST*** beállításához mást nem is kellett tennünk, mindössze az `mpiblast` és `mpirun` programokra mutató



3. ábra A beállításokra kattintva... állíthatjuk a környezetet LAM-ra



4. ábra LAM lesz az alapértelmezett környezet

közvetett hivatkozásokat kellett elkészítenünk, úgy, hogy a `$PATH` változó szerint elérhetőek legyenek.
Az *mpiBLAST* telepítése után le kellett töltenünk az adatbázis keresőt. Hogy az *mpiBLAST* helyesen fusson le az adatbázisnak *FASTA* formátumban kel lennie. Az *NCBI* minden adatbázisát összefoglalja *NCBI* weblapon található indexben, ahol a *FASTA* alkönyvtárban valamennyi adatbázist megtalálhatunk *FASTA* formátumban. Letöltöttük az *nr* adatbázis másolatát a */usr/local/mpiBLAST/db/ NFS*-megosztású könyvtárba, amelyet az *mpiBLAST* beállítása során készítettünk el.
Az *mpiBLAST* egyik parancsa az `mpiformatdb` amely az adatbázist szegmensekké alakítja. E szegmensek szám a fürtünk csomópontjainak számától függ.
Az `mpiformatdb` a létrehozott darabokat a megosztott könyvtárba helyezi el. Ezt a könyvtárat az *mpiblast.conf* állományban adhatjuk meg a telepítés során és valamennyi *mpiBLAST* program ezt használja. Nézzünk egy példát az adatbázis formázására:

```
# /usr/local/mpiBLAST/bin/mpiformatdb -N 16 -i nr
```

Itt a `-N` az adatbázis szeletek számát határozza meg amely általában megegyezik a fürtünk csomópontjainak számával az `-i` pedig az adatbázisformátum nevét határozza meg. Példánkban az *nr* adatbázist 16 külön szeletté alakítottuk. Az `mpiformatdb` nem másolja át a szeleteket a csomópontokra, következésképpen az első lekérdezés kiadásakor igen jelentős többletmunkát jelent amíg valamennyi csomópont lemásolja a maga szeletét. Minden csomópont csak egyszer másol le egy szeletet. Ha azonban a szeletet töröljük a csomópontról, a következő lekérdezéskor ismét átmásolódik.
A fürt kezelésének egyszerűsítése érdekében, készíttetünk egy parancsfájlt, amely letölti az adatbázis legfrissebb változatát, megformázza az `mpiformatdb`-vel, majd kiosztja azt a csomópontoknak egy egyszerű *BLAST* lekérdezés kiadásával. A parancsfájlt `cron` segítségével ütemeztük, hogy hetente induljon el. Amikor végül már képesek voltunk párhuzamos lekérdezéseket végrehajtani a *BLAST* segítségével, a *WWW BLAST*-ról feltelepítettük a web alapú előlapot.

A WWWBlastwrap.pl beállítása

Az *mpiBLAST* segítségével ugyan csak parancssori *BLAST* kereséseket adhatunk ki, ám tartalmaz két állományt (*blast.cgi* és *WWWBlastwrap.pl*) amelyek a web alapú előlappal alakítanak ki kapcsolatot. Ezek az állományok úgy vannak beállítva, hogy együttműködhessenek a *WWW BLAST* rendszerével. Következő teendőnk tehát a *WWW BLAST* letöltése volt a */var/www* könyvtárba, és létrehoztuk a */var/www/blast/* mappát. A *WWW BLAST*-ban néhány dolgot át kellett állítani, hogy a *BLAST* kereséseket párhuzamosan hajtsa végre.
A *WWW BLAST* saját könyvtárat biztosít az adatbázisainak. Mivel mi az *mpiBLAST* segítségével alakítottuk ki az adatbázisokat a *WWW BLAST db/* könyvtárát az *mpiBLAST* könyvtárára kellett irányítanunk. Ezért a *blast/* könyvtár db/ mappáját az *mpiBLAST /db* könyvtárára mutató közvetett hivatkozásként hoztuk létre.
A *WWW BLAST blast.cgi* állománya hajtja végre a *BLAST* lekérdezéseket. Az *mpiBLAST*-ban találunk egy másik *blast.cgi* állományt, amely párhuzamos *BLAST* lekérdezéseket hajt végre a *WWWBlastwrap.pl* segítségével. A *WWWBlastwrap.pl Perl* parancsfájl készíti le az *mpiBLAST* által végrehajtandó lekérdezést.
A *WWWBlastwrap.pl* mindezt egy másik *Perl* parancsfájl alakjában teszi, amelyet a webes űrlapról begyűjtött adatokkal tölt fel. Az *OpenPBS* ezt a parancsfájlt kapja meg.
A *WWWBlastwrap.pl* több feladatot is ellát: egyrészt értelmezi az űrlap adatait, létrehozza a parancsfájlt, amelyet a munkafolyamat sorokat és terheléselosztást végző *OpenPBS* kap majd meg valamint a *BLAST* keresési eredményt böngészőbarát formájúra alakítja.
Mindazonáltal néhány módosítást kell végeznünk a *WWWBlastwrap.pl* állományban, hogy helyesen működjön a mi környezetünkben is. Az első változtatás a globális `$scratch_space` és `$MPIBLASTCONF` változók beállítása volt. Ezt a két változót a parancsfájl teljes futása során használja. A `$scratch_space` a lekérdezés alatt

használt ideiglenes állományok elérési útját tartalmazza. A `$MPIBLASTCONF` az ***mpiBLAST*** beállításállomány abszolút elérési útját tartalmazza. Mindkét könyvtárat az ***mpiBLAST*** telepítésekor állítottuk be. A két változót a következőképen állítottuk be:

```
$scratch_space="/usr/local/mpiBLAST/shared/scratch";
$MPIBLASTCONF="/usr/local/mpiBLAST/etc/mpiblast.conf";
```

A következő változtatást az `if` feltételes elágazásokon kell végrehajtanunk. Ezek az utasítások beégetik a `NUMPROC` környezeti változót az ***nt***, ***nr*** és ***pdb*** adatbázisok esetében. Mivel az adatbázisokat az ***mpiBLAST***-al előformázzuk, a lekérdezésenként felhasznált processzorszám állandó. Az alapértelmezett 20-as értéket 16-ra állítottuk, hiszen mi ennyi processzort alkalmazunk:

```
if($data{'DATALIB'} eq "nt"){
    $data{'NUMPROC'} = 16;
}
```

A parancsfájl további részében találjuk a `ValidateFormData` függvényt. Ez az alprogram biztosítja, hogy a felhasználó a megfelelő adatbázis kiszolgáló párosítást válassza, és 500-as kiszolgáló hibát ad vissza ha nem a megfelelő kombinációt választottuk. Ezt az alprogramot módosítottuk az alábbiak szerint, hogy a `tblastx` program hajtsa végre a lekérdezéseket az nr adatbázison:

```
#### ELÖTTE ####
# a nucleotide adatbázisra kell alkalmazni
if($data_ref->{'DATALIB'} ne "nt"){

#### UTÁNA ####
# a nucleotide adatbázisra kell alkalmazni
if($data_ref->{'DATALIB'} ne "nt" ||
   $data_ref->{'DATALIB'} ne "nr"){
```

Később a parancsfájl elkészíti a ***mpiBLAST*** parancssori paramétereit majd eltárolja őket a `$c_line` változóban. Nekünk a -d kapcsolóhoz átadott paramétert kell megváltoztatnunk, amely a keresendő adatbázisról tájékoztatja az ***mpiBLAST***-ot. Alapértelmezés szerint a ***WWWBlastwrap.pl*** összefűzi az adatbázis nevét a processzorok számával és az eredményt átadja a -d kapcsolóval. következésképpen, ha az adatbázisunkat ***nr***-nek nevezik és 16 processzorunk van, akkor ***nr16***-ot adna át. Valószínűleg azért van ez így, hogy az adatbázis több verziójában is kereshessünk és az ***nr16*** a 16-szeletből álló, míg az ***nr8*** a nyolc szeletből álló adatbázist különbözteti meg. Két lehetőségünk van, vagy ilyen alakúra nevezzük az adatbázisunkat, vagy módosítjuk a parancsfájlt. Tekintve, hogy nekünk mindig csak egyetlen adatbázisverziónk lesz, inkább a parancsfájlt módosítottuk és eltávolítottuk a processzorok számát az adatbázis nevéből. A kód változtatásai összefoglalva a következők:

```
#### ELÖTTE ####
# az mpiBlast my-nak átadtott parancsfájl
# létrehozása
```

5. ábra 17 újrahasznosított PC-t tartalmazó fürtünk lerövidíti a felhasználói lekérdezések válaszidejét


```
$c_line = "-d $data_ref->{'DATALIB'}" .
          "$data_ref->{'NUMPROC'} " .
          "-p $data_ref->{'PROGRAM'} " .
#### UTÁNA ####
# az mpiBlast my-nak átadtott parancsfájl
# létrehozása
$c_line = "-d $data_ref->{'DATALIB'} " .
          "-p $data_ref->{'PROGRAM'} " .
```

A tesztlekérdezések futtatása során néhány `lcl|tmpseq_0: Unable to open BLOSUM62` figyelmeztetést találtunk az ***OpenPBS*** hibanaplójában. Ezeket a figyelmeztetéseket úgy tüntethetjük el, ha a ***BLASTMAT*** környezeti változót a ***BLAST*** mátrixok helyére irányítjuk ezért a következő változtatásokat végeztük el:

```
#### ELÖTTE ####
print SCRIPTFILE '#PBS -e '.
"$data_ref->{'ERROR_LOG_FILE'}\n\n";
print SCRIPTFILE 'if(-e $ENV{PBS_NODEFILE} )
➥{'."\n";

#### UTÁNA ####
print SCRIPTFILE '#PBS -e '.
"$data_ref->{'ERROR_LOG_FILE'}\n\n";
print SCRIPTFILE '$ENV{BLASTMAT} = '.
'"/usr/local/ncbi/data";'."\n";
print SCRIPTFILE 'if(-e $ENV{PBS_NODEFILE} )
➥{'."\n";
```

Az utolsó módosítással a parancsfájl vége felé kerültünk szembe a `HtmlResults` alprogramban. A kód, amely a felhasználót az eredményekhez irányítja az alapértelmezett ***URL***-t használja, mi pedig szinte biztosan nem ezt szeretnénk. Az alapértelmezett ***URL***-t átállítva a webkiszolgálónkra az ügyfél webböngészője meg tudja jeleníteni a ***BLAST*** lekérdezés eredményét:

```
#### ELÖTTE ####
print "Location: https://jojo.lanl.gov/blast/".
"BlastResults/$results_file\n\n";
```

```
#### UTÁNA ####
print "Location: http://domain_name/BlastResults".
"/$results_file\n\n";
```

Összefoglalás és eredmények

Helyi fürtünk az *NCBI* weblapján találhatónál kevesebb párhuzamos felhasználóval és jobb összesített átviteli idővel képes keresni a naprakész adatbázisban. Fürtünk és az *NCBI* weblap között egyszerű falióra módszerrel tettünk különbséget. Nyolc egyszerű, fehérje és *DNS* sorozatokat tartalmazó lekérdezést használtunk. Az időzítőt a lekérdezési weblap űrlapjának elküldése után indítottuk és az eredménye megjelenésekor állítottuk le. Az *NCBI* weblapjának eredményei két hét távlatában erősen változtak. Mind a nyolc eredményt átlagoltuk és összehasonlítottuk a fürt idejével. Az eredményeket azért az elküldés pillanatától a megjelenés pillanatáig mértük, mert a felhasználó is ezzel az értékkel találkozik. A fürtnek átlagosan kevesebb időre volt szüksége a lekérdezés végrehajtásához.

Linux Journal 2005. május, 133. szám

Josh Stroschein (jstrosch@usd.edu) Számítástechnikai valamint bűn- és igazságügyi szakértői diplomáját tervezi mostanában. Josh az USD jóvoltából dolgozhat a fürt projekten. Ezen kívül a Vermillioni (SD) Walton Internet Solutions cégnek is dolgozik.

Doug Jennewein (djennewe@usd.edu) informatikus kutatómérnök aki 1998 óta dolgozik az USD-nél. 2004-ben szerezte meg diplomáját az USD-n. Doug fő kutatási területe a nagysebességű számítások.

Joe Reynoldson (jreynold@usd.edu) a kutatás fő irányítója és útmutatója a Számítástechnikai Tanszéken, és 1994 óta dolgozik az USD-nél. Informatikusi diplomáját 1997-ben szerezte az USD-n. Joe Perl, rendszerkezelés és webfejlesztő órákat ad.

KAPCSOLÓDÓ CÍMEK

- oscar.openclustergroup.org
- oscar.openclustergroup.org/tiki-list_file_gallery.php?galleryId=2
- www.linuxjournal.com/article/7462
- www.ncbi.nih.gov/BLAST
- mpiblast.lanl.gov
- ftp.ncbi.nih.gov/blast/executables/release
- ftp.ncbi.nlm.nih.gov/blast/db/FASTA

LIDS – A Linux betörésvédelmi rendszere

A fájlok jogosultságainak beállítása néha kevés. A LIDS olyan kernelszintű eszköz, amelynek hatása messze túlmutat a fájlrendszer beállításaival elérhető biztonságon.

Amint egyre többen használnak ***Linuxot***, egyre több biztonsági hibára derül fény. Ez egyrészt természetes, másrészt a fejlesztés nyíltsága miatt a legtöbb ilyen problémát villámgyorsan megoldják a programozók.
Néha azonban az is előfordulhat, hogy nem sikerül valamelyik problémát időben megoldani, így a köztes időszakban a kérdéses szoftvert futtató gépek sebezhetőek lesznek. A rosszindulatú felhasználók a rést kihasználva esetleg root jogosultságot szerezhetnek, és akár nagy pusztítást is végezhetnek. Ez az a pont, ahol nagy hasznát vehetjük a ***LIDS*** rendszernek (***Linux Intrusion Detection System; Linux Behatolásérzékelő Rendszer***).
A ***LIDS*** egy kernelfolt, ami a rendszermag szintjén tesz lehetővé a hozzáférési jogosultságok ellenőrzését, illetve a kapcsolatok kezelését. Két adminisztrációs segédeszköz tartozik hozzá a `lidsconf` és a `lidsadm`, amelyekkel minden szolgáltatása beállítható.
A `lidsadm` tulajdonképpen egy olyan segédprogram, amellyel a ***LIDS***-et egy adott terminálra nézve üzemen kívül helyezhetjük. Így elvégezhetünk olyan beállításokat, amelyeket amúgy maga a ***LIDS*** hivatalból meggátolna. A programmal egyben a jelenleg érvényes beállításokat is megjeleníthetjük.
A `lidsconf` arra szolgál, hogy segítségével egyes fájlok hozzáférési jogosultságait beállíthassuk. Ezek lehetnek bináris fájlok vagy természetesen bármi más. A ***LIDS*** nevezéktanában ezeket a fájlokat objektumoknak (objects) hívják, a velük kapcsolatos engedélyezett vagy tiltott képességek illetve műveletek pedig a szubjektumok (subjects). A ***LIDS*** automatikusan felülbírálja az olyan beállításokat mint például a fájlrendszerben tárolt jogosultságok. Ez azt jelenti, hogy a ***LIDS*** segítségével elérhetetlenné tehetünk gyakorlatilag bármit, legyen az egy fájl, nyers adatokat kezelő eszköz, memóriaterület vagy I/O port. Ráadásul a korlátozások magára a rendszergazdára is érvényesek. Röviden, a ***LIDS*** egy teljes biztonsági modell ***Linuxon*** megvalósítva.

Telepítés

A fejlesztők a telepítéssel kapcsolatos utasításokat szokásos módon a csomagban található ***INSTALL*** nevű fájlban írták le. Ez alapján ugyan mindenki elboldogulhat a rendszerrel, ezzel együtt talán nem árt, ha végigmegyünk a fontosabb lépéseken. A ***LIDS*** stabil változatait mindig az eredeti (vanilla) ***Linux*** rendszermaggal együtt lehet használni. Ennek megfelelően a fejlesztők azt javasolják, hogy a ***LIDS*** foltot mindig az eredeti kernelforrásra, és nem valamilyen terjesztés-specifikus forrásra alkalmazzuk. Utóbbi esetben különböző fordítási hibákba ütközhetünk, mivel a legtöbb terjesztés valamilyen testreszabott kernelt tartalmaz.
Ezen kívül a ***LIDS***-nek közismerten gondjai vannak a nem ***i386*** alapú architektúrákkal.
Nézzük tehát a telepítés lépéseit.
A például a ***lids-2.2.1-2.6.13.tar.gz*** nevű csomagban található változatot a névnek megfelelően a 2.6.13-as kernelforrásra kell alkalmazni.
Az első lépés a foltozás:

```
patch -p1
➥/dir_to_the_patch_file/
➥patch-lids-2.2.1-2.6.13
```

Ha ezzel megvagyunk, a

```
make [x/menu]config
```

paranccsal végezzük el a szükséges beállításokat. Ne felejtsük el a biztonsági szakaszban engedélyezni a ***LIDS*** használatát. Ha ezzel is készen vagyunk, nincs más hátra, mint lefordítani az új kernelt a következő parancsokkal:

```
make
make modules_install
```

A második lépésre természetesen csak akkor van szükség, ha modulba tettünk bizonyos funkciókat.
Másoljuk át a `bzImage` nevű fájlt ***/kernelpath/arch/i386/boot*** könyvtárból saját rendszerünk ***/boot*** könyvtárába, majd konfiguráljuk újra a rendszertöltőt. A következő újraindításnál már a ***LIDS***-szel megerősített kernel fog futni.

A ***LIDS*** állapotáról a

```
lidsadm -v
```

paranccsal kérhetünk jelentést.
Ha itt bármilyen hibaüzenetet kapunk, az azt jelenti, hogy a ***LIDS*** nem épült be a rendszermagba.

Ilyenkor mindenképpen újra kell fordítani a kernelt, persze csak az után, hogy kiküszöböltük a hibát.

A hozzáférési jogosultságok beállítása

Mielőtt hozzálátnánk a különböző szerveralkalmazásokkal kapcsolatos jogosultságok beállításához, nem árt megismerni a `lidsconf` parancs használatának általános szintaxisát:

```
lidsconf -A [-s subject]
➥-o object [-d] [-t from-to]
➥[-i level] -j ACTION
```

A `subject` mezőbe egy program neve kerül, amellyel kapcsolatban valamilyen képességet akarunk beállítani. Az `object` helyén felbukkanhat valamilyen bináris fájl, könyvtár, socket neve, vagy egy képesség.
A `-d` kapcsoló hatására a ***LIDS*** a megadott tartományt végrehajtási tartományként fogja kezelni.
Az idővel kapcsolatos korlátozások beállítására a `-t` kapcsoló szolgál, míg a `-i` az öröklődés szintjét határozza meg.
A `-j` kapcsoló után egy műveletet kell megadni, ami a a következő közül az egyik lehet:

- `DENY`: Megtagadjuk egy adott objektumhoz való hozzáférést.
- `READONLY`: Csak olvashatóvá teszi a kérdéses objektumot.
- `APPEND`: Általában naplózási célra használatos. Azt teszi lehetővé, hogy egy program hozzáfűzzön bejegyzéseket egy bizonyos fájlhoz, de ne törölhesse azt.
- `WRITE`: Bináris állományoknak engedélyezi, hogy írjanak a fájlba.
- `GRANT`: Egy képességgel együtt használatos. Jelentése az adott képesség engedélyezése.
- `IGNORE` és `DISABLE`: Az elsővel egy adott objektumra vonatkozó adott engedélyt lehet visszavonni, a másodikkal egy kiterjesztést lehet hatályon kívül helyezni.

A ***LIDS*** által támogatott képességekről a

```
lidsadm -h | grep CAP
```

paranccsal kérhetünk listát.

A jelenleg támogatott képességek a következők:

- `CAP_CHOWN`: chown/chgrp.
- `CAP_DAC_OVERRIDE`: DAC hozzáférés.
- `CAP_DAC_READ_SEARCH`: DAC olvasás.
- `CAP_FOWNER`: tulajdonos azonosítója, nem azonos a felhasználóval.
- `ID CAP_FSETID`: effektív felhasználói azonosító, nem azonos a tulajdonossal.
- `ID CAP_KILL`: valódi/effektív ID, nem azonos a folyamat-azonosítóval.
- `ID CAP_SETGID`: `set*gid`(2).
- `CAP_SETUID`: `set*uid`(2).
- `CAP_SETPCAP`: átviteli képesség (transfer capability).
- `CAP_LINUX_IMMUTABLE`: változtathatatlan (immutable) és hozzáfűzési fájlattribútum.
- `CAP_NET_BIND_SERVICE`: kapcsolódás 1024-es alatti portokhoz.
- `CAP_NET_BROADCAST`: üzenetszórás (broadcasting) illetve többszörös üzenetszórás (multicast) vétele.
- `CAP_NET_ADMIN`: interfész/tűzfal/útválasztási szabályok változtatása.
- `CAP_NET_RAW`: nyers (raw) socket-ek.
- `CAP_IPC_LOCK`: megosztott memóriaszegmensek zárolása.
- `CAP_IPC_OWNER`: IPC-tulajdonjog ellenőrzése.
- `CAP_SYS_MODULE`: kernelmodulok beillesztése és eltávolítása.
- `CAP_SYS_RAWIO`: `ioperm`(2)/`iopl`(2) hozzáférés.
- `CAP_SYS_CHROOT`: `chroot`(2).
- `CAP_SYS_PTRACE`: `ptrace`(2).
- `CAP_SYS_PACCT`: folyamatok kezelésének beállítása.
- `CAP_SYS_ADMIN`: számos adminisztratív képesség.
- `CAP_SYS_BOOT`: `reboot`(2).
- `CAP_SYS_NICE`: `nice`(2).
- `CAP_SYS_RESOURCE`: erőforrás-határok beállítása.
- `CAP_SYS_TIME`: rendszeridő beállítása.
- `CAP_SYS_TTY_CONFIG`: tty beállítása.
- `CAP_MKNOD`: `mknod` műveletek.
- `CAP_LEASE`: fájlok „bérlése” (lease).
- `CAP_HIDDEN`: rejtett folyamat.
- `CAP_KILL_PROTECTED`: védett programok kilövése.
- `CAP_PROTECTED`: folyamat védelme jelektől.

LIDS segítségével biztosított kiszolgáló felállítása

A továbbiakban feltételezzük, hogy az olvasónak sikerült feltelepítenie mind magát a ***LIDS*** rendszert, mind a vele kapcsolatos adminisztratív eszközöket. Egy viszonylag szoros biztonsági hálóval védett rendszert fogunk felállítani, amelyen egyedül a ***MySQL***, az ***Apache*** és a ***Bind*** futhat.
A beállításoknál feltételezzük, hogy az Apache webkiszolgálót a */usr/local/apache* helyre telepítettük, illetve hogy a naplófájlok a */var/log/httpd* könyvtárban találhatók. Szintén feltesszük, hogy az Apache beállítóállományait tartalmazó könyvtár a */etc/httpd*. Hasonló feltevésekkel élünk a ***MySQL*** rendszerrel kapcsolatban is. A telepítési könyvtárnak a */usr/local/mysql* helyet fogjuk tekinteni. Ha az olvasó rendszerének nem ugyanez a szerkezete, akkor az itt felsorolt parancsokat természetesen megfelelően módosítani kell.
A rendszer teljes biztosításához szükséges dolgok tárgyalása természetesen messze túlmutat e cikk keretein. Ugyanakkor a bemutatott beállítások jó alapot nyújtanak ahhoz, hogy használatban vegyük a ***LIDS*** rendszert.

A rendszer felállítása

Miután elindítottuk a ***LIDS***-szel kiegészített rendszermagot, elkezdhetjük megadni azokat a biztonsági beállításokat, amelyek a rendszer egyes részeihez való hozzáférést szabályozzák. A következő parancsokkal a */sbin*, */bin*, */usr/bin* és a */lib* könyvtárakat tehetjük csak olvashatóvá:

```
lidsconf -A -o /sbin -j
➥READONLY
lidsconf -A -o /bin -j READONLY
lidsconf -A -o /usr/bin -j
➥READONLY
lidsconf -A -o /lib -j READONLY
```

A következő lépésben megadunk néhány hasonló beállítást a */opt*, */etc*

és */usr/local/etc* könyvtárakkal kapcsolatban is, amelyeknek szintén csak olvashatónak kell lenniük. A */etc/shadow* fájlhoz, és a rendszertöltőt tartalmazó fájlokhoz mindennemű hozzáférést megtagadunk:

```
lidsconf -A -o /etc -j READONLY
lidsconf -A -o /usr/local/etc
➥-j READONLY
lidsconf -A -o /etc/shadow -j
➥DENY
lidsconf -A -o /etc/lilo.conf
➥-j DENY
```

Mivel a */etc/shadow*-hoz immár egyetlen folyamat sem férhet hozzá, ez egyben a bejelentkezéseket is megátolja, hiszen a rendszer nem tudja hitelesíteni a felhasználókat. Éppen ezért csak olvasási hozzáférést kell adnunk erre a fájlra a `login` és a `vlock` programoknak. Emellett a `su` parancsnak szintén olvasási joggal kell rendelkeznie a */etc/shadow*-val kapcsolatban, tehát engedélyezzük ezt is:

```
lidsconf -A -s /bin/login -o
➥/etc/shadow -j READONLY
lidsconf -A -s /usr/bin/vlock
➥-o /etc/shadow -j
➥READONLY
lidsconf -A -s /bin/su -o
➥/etc/shadow -j
➥READONLY
```

Ami a `su` használhatóságát illeti, be kell állítanunk még néhány hozzáférési jogosultságot ahhoz, hogy kezelni tudja az *UID* és *GID* értékeket:

```
lidsconf -A -s /bin/su -o
➥CAP_SETUID -j GRANT
lidsconf -A -s /bin/su -o
➥CAP_SETGID -j GRANT
lidsconf -A -s /bin/su -o
➥/etc/shadow -j
➥READONLY
```

A következő lépésben engedélyeznünk kell az `init` és `login` programoknak, valamint az ezekkel kapcsolatos folyamatoknak, hogy írjanak a naplófájlokba:

```
lidsconf -A -o /var/log -j
➥APPEND
lidsconf -A -s /bin/login -o
➥/var/log/wtmp -j WRITE
lidsconf -A -s /bin/login -o
➥/var/log/lastlog -j WRITE
lidsconf -A -s /sbin/init -o
➥/var/log/wtmp -j WRITE
lidsconf -A -s /sbin/init -o
➥/var/log/lastlog -j WRITE
lidsconf -A -s /sbin/halt -o
➥/var/log/wtmp -j WRITE
lidsconf -A -s /sbin/halt -o
➥/var/log/lastlog -j WRITE
lidsconf -A -s /etc/rc.d/
➥rc.sysinit -o /var/log/wtmp
➥-i 1 -j WRITE
lidsconf -A -s
➥/etc/rc.d/rc.sysinit -o
➥/var/log/lastlog -i 1
➥-j WRITE
```

Ismét egy fontos lépés: meg kell adnunk a root felhasználó saját könyvtárával kapcsolatos jogosultságokat. Az egyetlen művelet, amit engedélyezünk az, hogy a *Bash* hozzáfűzhessen bejegyzéseket a `history` fájlhoz:

```
f -A -o /root -j READONLY
lidsconf -A -s /bin/bash -o
➥/root/.bash_history -j APPEND
```

Végezetül engedélyezzük az `init` folyamatnak, hogy leállításkor kilője a folyamatokat:

```
lidsconf -A -s /sbin/init -o
➥CAP_INIT_KILL -j GRANT
lidsconf -A -s /sbin/init -o
➥CAP_KILL -j GRANT
```

Ezen kívül engedélyeznünk kell még az `fstab`-nak és az `init` szkripteknek, hogy fájlrendszereket csatoljanak be, folyamatokat állítsanak le, illetve fájlrendszereket távolítsanak el:

```
lidsconf -A -s/etc/fstab -o
➥CAP_SYS_ADMIN -j 1 -j GRANT
lidsconf -A -s /etc/rc.d/
➥init.d/halt -o CAP_INIT_KILL
➥-i 1 -j GRANT
lidsconf -A -s /etc/rc.d/
➥init.d/halt -o CAP_KILL -i 1
➥-j GRANT
lidsconf -A -s /etc/rc.d/
➥init.d/halt -o CAP_NET_ADMIN
➥-i 1 -j GRANT
lidsconf -A -s /etc/rc.d/
➥init.d/halt -o CAP_SYS_ADMIN
➥-i 1 -j GRANT
```

Az Apache webkiszolgálóval kapcsolatos jogosultságok beállítása

Az *Apache*-nek *setuid* és *setgid* képességekkel kell rendelkeznie ahhoz, hogy működni tudjon. Szintén hozzá kell férni a naplófájlokhoz, más binárisoknak pedig meg kell tiltanunk, hogy a `httpd` bináris állományt módosíthassák:

```
lidsconf -A -s /usr/local/
➥apache/bin/httpd -o
➥CAP_SETUID -j GRANT
lidsconf -A -s /usr/local/
➥apache/bin/httpd -o
➥CAP_SETGID -j GRANT
lidsconf -A -o /etc/httpd -j
➥DENY
lidsconf -A -s /usr/local/
➥apache/bin/httpd -o
➥/etc/httpd -j READONLY
lidsconf -A -o /usr/local/
➥apache -j DENY
lidsconf -A -s /usr/local/
➥apache/bin/httpd -o /usr/
➥local/apache -j READONLY
lidsconf -A -o /var/log/httpd
➥-j DENY
lidsconf -A -s /usr/local/
➥apache/bin/httpd -o
➥/var/log/httpd -j APPEND
lidsconf -A -s /usr/local/
➥apache/bin/httpd -o /usr/
➥local/apache/logs -j WRITE
```

MySQL

Ami a *MySQL*-t illeti, itt is meg kell tiltanunk minden más bináris állománynak, hogy a `mysql` programot módosíthassa. Korlátoznunk kell a *mysql/var* könyvtárhoz való hozzáférést is úgy, hogy ahhoz legfeljebb hozzáfűzési joga legyen a folyamatoknak, magának a `mysqld` démonnak pedig csak olvasási jogot szabad adni a *mysql* könyvtárra:

```
lidsconf -A -o /usr/local/
➥mysql/var -j APPEND
lidsconf -A -o /usr/local/mysql
➥-j DENY
lidsconf -A -s /usr/local/
➥mysql/libexec/mysqld -o
➥/usr/local/mysql -j READONLY
lidsconf -A -s /usr/local/
➥mysql/libexec/mysqld -o
➥/usr/local/mysql/var -j
➥WRITE
```

Bind

A *Bind* rendszernek számos képességgel kell rendelkeznie ahhoz, hogy megfelelően tudjon működni:

```
lidsconf -A -s /usr/sbin/named
➥-o CAP_NET_BIND_SERVICE 53
➥-j GRANT
lidsconf -A -s /usr/sbin/named
➥-o CAP_SETPCAP -j GRANT
lidsconf -A -s /usr/sbin/named
➥-o CAP_SYS_CHROOT -j GRANT
lidsconf -A -s /usr/sbin/named
➥-o CAP_SYS_RESOURCE -j GRANT
lidsconf -A -s /usr/sbin/named
➥-o CAP_SETUID -j GRANT
lidsconf -A -s /usr/sbin/named
➥-o CAP_SETGID -j GRANT
```

Login

A `login` az a program, amely a felhasználók bejelentkezését kezeli egy *GNU/Linux* rendszeren. Ennek megfelelően a következő beállításokat kell vele kapcsolatban megadnunk:

```
lidsconf -A -s /bin/login -o
➥/etc/shadow -j READONLY
lidsconf -A -s /bin/
➥login -o CAP_SETUID -j
➥GRANT
lidsconf -A -s /bin/login -o
➥CAP_SETGID -j GRANT
lidsconf -A -s /bin/login -o
➥CAP_CHOWN -j GRANT
lidsconf -A -s /bin/login -o
➥CAP_FSETID -j GRANT
```

A fenti beállítások elvégzése után le kell zárnunk a kernelt ahhoz, hogy a *LIDS* érvényesülni tudjon. Ehhez adjuk hozzá a következő sort a *rc.local* fájlhoz:

```
lidsadm -I
```

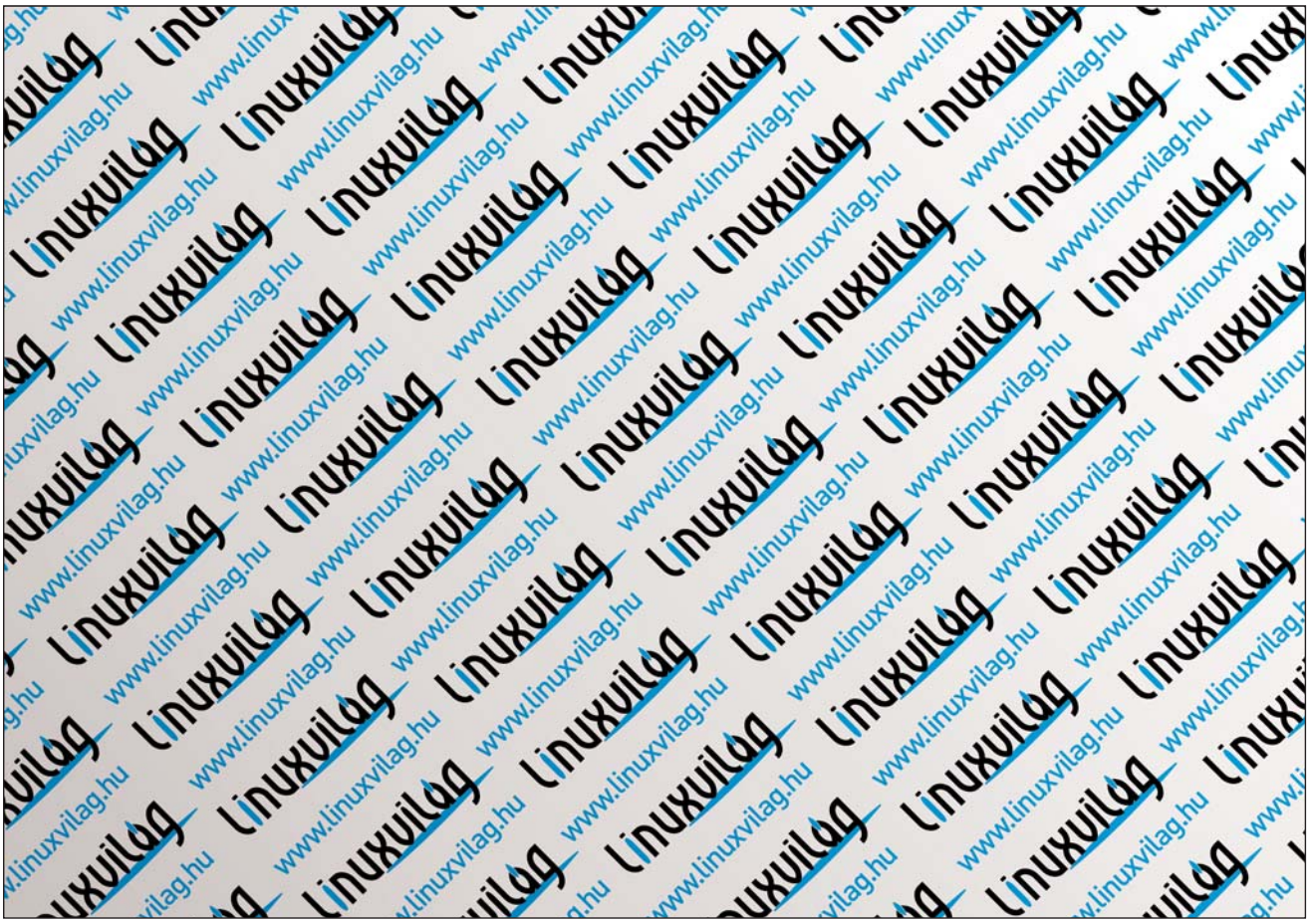
Indítsuk újra a gépet, hogy a beállítások érvénybe lépjenek. Fontos megjegyezni, hogy a bemutatott beállításokkal nem fogjuk tudni futtatni az *X* kiszolgálót sem, mivel az nyers I/O műveleteket használ. Ez persze nem különösebben nagy gond, hiszen a legtöbb kiszolgálónak szánt gépen eleve nem fut az X. ha mégis szükségünk lenne a grafikus felületre, akkor egészítsük ki az eddigi beállításokat a következővel (feltéve, hogy az *X* kiszolgáló bináris állomány a */usr/X11R6/bin/startx*):

```
lidsconf -A -s /usr/X11R6/
➥bin/startx
```

A bemutatott példákból talán érzékelhet, hogy a *LIDS* egy igen hatékony biztonsági kiegészítés, amely akár magától a root felhasználótól is képes megvédeni a rendszert. Mellesleg a használata sem különösebben bonyolult.

Linux Journal 2006. 143. szám

Ifran Habib informatikushallgató a Pakisztáni Nemzeti Tudományos és Műszaki Egyetemen. Középiskolás kora óta erősen érdeklődik a Linux és a nyílt forrású technológiák iránt, a beágyazott rendszerektől egészen a webszolgáltatásokig. Az elmúlt két évben igyekezett népszerűsíteni a Linuxot Pakisztánban, és számos cikket is írt a témáról helyi újságokba és magazinokba.



Csináld magad linuxos csúcsgép

Először is vegyünk egy igazán kiváló AMD64 alaplapot, aztán a többi tulajdonképpen már jön magától...

Sokan vannak köztünk olyanok, akik maguk szeretik megépíteni a gépüket. Van valami egyedülállóan érdekes abban, hogy az ember maga választhatja ki és szerelheti be az összes alkatrészt. Így mellesleg lehetőségünk van arra is, hogy ésszerű kompromisszumokat kössünk az ár és a teljesítmény tekintetében, így olyan gépet alkothatunk, amely nem avul el a hat hónapon belül. A legtöbb *AMD64*-es alaplap például csak 4 GB *RAM*-ot képes fogadni, a mi kedvencünk azonban nyolcat. Persze nem különösebben valószínű, hogy ki fogjuk használni ezt a képességét, de akkor is megnyugtató a tudat, hogy ha úgy hozná a sors, akkor a rendszerünknek van hová fejlődnie. Ha készen vennénk meg a gépet, szinte biztos, hogy ez a lehetőség nem állna nyitva előttünk.

A türelmetleneknek rögtön a cikk elején elmondom, végül mi került a gépbe. Íme:

- Alaplap: *ABIT AN8 32X 939*
- Processzor: *AMD64 4200+ Athlon X2*
- Tápegység: *Silverstone SST-ST65ZF* 650 Watt
- Memória: két pár *Corsair 1Gx2 TWINX2048-3200PRO* modul (összesen tehát négy)
- Videokártyák: két összepárosított *eVGA GF 7900GT 256* (*NVIDIA SLI*)
- Ház: *Silverstone TJ07-S*
- Merevlemezek: 2x *Seagate Barracuda 300GB 7200 RPM* 8MB cache *SATA 3.0Gb/s*
- DVD R+W: *Plextor PX-716AL/SW SATA*
- Monitor: *Samsung LCD 204B 20.1"*
- Billentyűzet: *Logitech Cordless Comfort Duo* (egér is van a csomagban)
- Egér: *Logitech G7 Laser Cordless mouse*

A fenti listába úgy került be a *G7 Laser Cordless* egér, hogy nekem történetesen ez a kedvencem. Ugyanakkor az árlistában ezt a tételt nem szerepeltetem, mivel a billentyűzet és az egér a gép két legszemélyesebb alkotóeleme, tehát teljesen fölösleges lenne ezzel kapcsolatban tanácsokat osztogatni. (Sőt, talán még a monitor is ebbe a kategóriába tartozik.) Én a *Logitech Cordless Comfort Duo* csomagban található billentyűzetet kifejezetten szeretem, viszont az egeret nem. Így aztán ezt kicseréltem a *G7 Laser Cordless* típusra.

Mi is volt a cél

Ennek a listának az összeállításával az volt a célunk, hogy egy olyan, kifejezetten nagy teljesítményű, a *Linux* futtatására mindenben alkalmas asztali gépet alkossunk meg, amelynek a megvásárlásához nem kell előbb bankot rabolnunk. Közismert, hogy a legtöbb terméktípusnál az ár/teljesítmény mutatóban van egy olyan töréspont, amit ha túllépünk, akkor nagyobb teljesítményt vagy jobb minőséget kapunk ugyan, de aránytalanul több pénzért. A gép összeállításakor természetesen mindenütt ez alá a töréspont alá céloztunk. Ugyanezzel magyarázható az is, hogy a hűtést ventilátorokkal, és nem méregdrága (mellesleg nehezen szerelhető) vízhűtéssel oldottuk meg.

Az árérzékeny olvasók kedvéért összeállítottunk egy „gazdaságos" konfigurációt is, ami továbbra sem különösebben olcsó, viszont a teljesítménye sem sokkal marad le a drágább variánshoz képest.

Az alkatrészek összeválogatása során egyik legfontosabb szempontunk az volt, hogy a kész gép azonnal, minden különösebb trükközés nélkül képes legyen működni, ha Linuxot telepítünk rá. Éppen ezért kipróbáltuk rajta a *Debian*, *Ubuntu/Kubuntu*, *Fedora Core 5*, *SuSE 10* és a *Mandriva* terjesztéseket, vagyis azokat, amelyeket feltehetőleg a legtöbben használnak. Valamennyi terjesztés azonnal működött, nem volt szükség semmilyen speciális meghajtó vagy szoftver beszerzésére. (Eltekintve persze az *NVIDIA* meghajtóktól, amelyekre mindenképpen szükségünk volt az *SLI* képességek kihasználása végett.) A *Live CD*-k közül a *Knoppix*-ot, a *MEPIS*-t és a *Kanotix*-ot próbáltuk ki. Itt sem történt semmi említésre méltó, egyszerűen minden működött.

Az alaplap

A konfiguráció alapját egy *ABIT AN8 32X 939* alaplap és egy *AMD64 4200+ Athlon X2* (kétmagos) processzor képezi. Utóbbi esetében a választásnál a döntő tényező az ár volt. Mire a cikk megjelenik, az *AMD* már valószínűleg tovább csökkentette a kétmagos processzorok árát, így az Olvasó ugyanannyi pénzért akár nagyobb teljesítményt is vehet, mint mi. Kihasználandó az alaplap *dual chanel* képességét két pár 1 GB-os *Corsair* memóriát (*1Gx2 TWINX2048-3200PRO*) tettünk a gépbe, vagyis összesen 4 GB *RAM*-unk van.

Az alaplap kiválasztása minden csináld magad rendszernél nagyon fontos lépés. Mi a *Socket 939*-es foglalatú *AMD64* processzort támogató lapok közül összesen három típust vizsgáltunk meg, az *ABIT AN8 32X*-et, az *MSI KN8 Diamond Plus*-t és az *ASUS AN832-SLI Deluxe*-ot. Mindhárom körülbelül 200 dollárba kerül (a beszerzési forrástól függően), vagyis az árkülönbség ebben az esetben nem volt adekvát szempont a döntésnél. Hasonlóan mindhárom lap támogatja a

Socket 939-es *AMD64*-es processzorokat, és mindhárom képes dual chanel üzemmódra, tehát ezekkel megint nem jutottunk előbbre. Valamennyivel ki tudjuk használni az *SLI* nyújtotta lehetőségeket is, hiszen mindhárom képes fogadni két egyforma videokártyát. Apropó mi két *eVGA GeForce 7900GT* kártyát használtunk a teszt során *SLI* módban.

Igazából a fenti három közül bármelyik alaplap kiváló választás lehet egy ilyen géphez. Az *MSI* lapja *Creative Sound Blaster Audigy* hangrendszerrel érkezik, tehát az *Audigy* rajongók nyugodtan vegyék meg ezt. Mind az *MSI*, mind az *ASUS* lapon két *LAN* port található, a *ABIT*-en azonban csak egy van. Akinek tehát fontos a két *LAN* csatlakozó, az megint közelebb jutott a döntéshez. Ugyanakkor az is igaz, hogy nem különösebben nehéz újabb hálózati kártyát vagy hangkártyát beszerelni, tehát nem biztos, hogy ezek megléte minden esetben lényeges szempont.

Annak ellenére, hogy az *AMD64*-es rendszerek memóriakezelő alrendszere nem az alaplapon, hanem magán a chipen található, lehetetlen rávenni az alaplapokat arra, hogy másként kezeljék a *RAM*-ot, mint ahogyan megtervezték őket. Ez esetünkben konkrétan azt jelenti, hogy ha veszünk egy olyan lapot, amit 4 GB *RAM* támogatására készítettek, akkor abba soha, semmilyen módon nem fogunk tudni beletenni 8 GB-ot. Nálunk végül ez volt az a szempont, amiért az *ABIT* lapja mellett döntöttünk. Ez hivatalosan 8 GB *RAM*-ot támogat, ami azt jelenti, hogy jobban képes kihasználni az *AMD64* processzor memóriakezelési képességeit, mint az *MSI* vagy az *ASUS* amúgy meglehetősen hasonló tudású termékei. További csavar a történetben, hogy az *MSI* és az *ASUS* lapokat úgy tűnik a 32 bites *Windows XP* képességeit szem előtt tartva tervezték meg, ugyanis alapértelmezésként a maximálisan engedélyezett 4 GB-ból is csak 3 GB-ot látnak. Ez a korlát ráadásul akkor is érvényben marad, ha az *AMD64* processzorra optimalizált Linuxot telepítünk a gépre, vagyis ez is csak 3 GB memóriát fog látni, holott 4 GB van a gépben. A BIOS beállításaival persze itt is orvosolható a baki, az *ABIT* lapja azonban minden különösebb teketória nélkül kezelte a 4 GB-ot. (Az *MSI* alaplap leírása egyenes azt állította, hogy lehetetlen 3 GB-nál több *RAM*-ot az operációs rendszer számára látható módon beletenni, de ennek a kijelentésnek a helyességét sem cáfolni, sem megerősíteni nem tudjuk, mivel nem próbáltuk ki.)

Vettük tehát az *ABIT* alaplapot, és beletettünk négy, egyenként 1 GB-os modult. Ezt a mennyiséget egy megfelelően beállított kernellel akkor is ki tudjuk használni, ha 32 bites *Linuxot* futtatunk egy ilyen gépen. A rendszermag alapvetően két részre osztja a rendelkezésre álló *RAM*-ot. Az egyik részét saját céljaira tartja fenn, a másik a felhasználói térbe kerül. A *Linux* attól függően végzi el ezt a felosztást, hogy milyen paraméterekkel fordítottuk le a kernelt (illetve milyen paramétereket használt az általunk futtatott terjesztést készítő cég).

Mindhárom alaplapnak volt még egy rigolyája a memóriakezeléssel kapcsolatban. Ha valamennyi foglalatba tettünk memóriamodult, akkor a memória órajele szép csendben visszaesett 400 MHz-ről 333 MHz-re. Fontos hangsúlyozni, hogy ez a „szolgáltatás" nem a memória mennyiségétől függ, hanem az váltja ki, ha a BIOS valamennyi foglalatban memóriát talál. Az órajelet persze mindhárom esetben vissza lehet állítani az eredeti értékre, ha bolyongunk kicsit a *BIOS* beállítások között. Egyes lapoknál ez nehezebb, másoknál könnyebben megy. Mindenesetre mikorra ezt a trükköt felfedeztük, már elhatároztuk, hogy a mi választottunk az *ABIT* lap lesz, amelynél történetesen egész egyszerű az órajel átállítása. Mindössze annyit kell tenni, hogy a *DRAM* időzítési beállításait visszaállítjuk a *„By SPD"* értékre. Amúgy semmiféle instabilitást nem érzékeltünk a rendszer működésében a nagyobb sebességen sem, tehát nyugodtan üzemeltethetjük 400 MHz-en a 4 GB RAM-ot is. Ugyanakkor talán azt is érdemes mindehhez hozzátenni, hogy a memória órajelének megemelése nem sok látható teljesítménynövekedést fog okozni. Ehhez igazából a késleltetési időkkel kellene kicsit eljátszani, ezt azonban mi nem mertük megkockáztatni, tehát a modulok paraméterei maradtak a gyári beállításokon.

Gondolom most sokan azt kérdezik, minek egy gépbe 4 GB *RAM*. Készséggel elismerem, hogy sokan lehetnek olyanok, akiknek ez a mennyiség egyszerűen ezoterikus. Aki tehát úgy gondolja, hogy munkája során nem fogja egyhamar – netán soha – kihasználni az általunk használt mennyiséget, az nyugodtan megépítheti ugyanezt a konfigurációt kevesebb *RAM*-mal is. Ebben az esetben persze megint érdemes megfontolni az *MSI* vagy az *ASUS* lap használatát, mivel két darab 1 GB-os modult mindhárom lap egyformán jól tud kezelni. Aki viszont akkora memóriaéhségben szenved, mint mi, annak az *ABIT* a legjobb választás, akár 32, akár 64 bites *Linuxot* akar majd a gépen futtatni.

A tápegység

Soha nem becsüljük le egy jó tápegység jelentőségét, ha gépet kezdünk építeni, különösen akkor ne, ha – mint mi is – mindjárt két videokártyát akarunk használni *SLI* módban. Ha ugyanis egy rendszer bizonyos pillanatokban nem tud felvenni kellő teljesítményt a tápegységtől, akkor garantáltan mindenféle bizarr hibajelenségeket fogunk tapasztalni. Aki tehát két videokártyával *SLI* módban kíván játszani, az 500 wattos tápegységnél alább semmiképpen nem adja. Ha pedig további *PCI* kártyák is lesznek a gépben, még ennél is nagyobb teljesítményfelvételre számíthatunk.

„Kedvenc" rendszer monitor, billentyűzet és egér nélkül

- Alaplap: *ABIT AN8 32X 939* ($200 US)
- Processzor: *AMD64 4200+ Athlon X2* ($360 US)
- Tápegység: *Silverstone SST-ST65ZF* 650 Watt ($170 US)
- Memória: Két pár *Corsair 1Gx2 TWINX2048-3200PRO* modul (a négy összesen $500 US)
- Videokártyák: két *eVGA GeForce 7900GT 256MB (NVIDIA SLI)* ($678 US)
- Ház: *Silverstone TJ07-S* ($365 US)
- Merevlemez: 2x *Seagate Barracuda 300GB 7200 RPM 8MB cache SATA 3.0Gb/s* ($200 US)
- DVD R+W: *Plextor PX-716AL/SW SATA* ($150 US)

Összesen: $2,623 US (monitor, billentyűzet és egér nélkül)

Szintén fontos, hogy ellenőrizzük a beszerelni kívánt tápegység hatékonysági mutatóit. Előfordulhat ugyanis, hogy egy tápegység csúcsteljesítménye megfelel ugyan az igényeinknek, tartósan terhelve azonban már nem képes leadni a megfelelő áramot.
Ami a két meghibásodás között eltelt átlagos időt illeti (*Mean Time Between Failures*, *MTBF*), nos ez az a paraméter, aminek jószerivel semmi információtartalma nincs számunkra. Mi azt gyanítjuk, hogy ezek az értékek arra az esetre vonatkoznak, amikor a tápegységet egy halhatatlan ventilátor hűti. Sajnálatos módon azonban a ventilátorok halandóak. A légáramlás megáll, a tápegység túlmelegszik, aztán viszlát szép világ. Szóval ennyit a közepes időkről. Az Olvasó tapasztalatai természetesen eltérhetnek a mieinktől, de ránk a legjobb benyomást eddig az *Enermax* tápegységek gyakorolták.
Mindhárom általunk vizsgált alaplaphoz 24 tűs tápcsatlakozóval ellátott tápegység szükséges. A tápnak emellett rendelkeznie kell a videokártyák tápellátásához szükséges csatlakozóval is, ha azokat *SLI* módban kívánjuk használni. Mindezeknek a szempontoknak megfelelően mi egy 500 wattos *Enermax ELT500AWT* típusú tápegységet választottunk, ami beszerzési forrástól függően körülbelül 100 dollárba kerül. Egy hasonló konfigurációhoz kipróbáltunk egy 650 wattos *Silverstone SST-ST65ZF* típusú tápot is, ami valamivel drágább, 170 dollár körüli áron kapható. Igaz ugyan, hogy a cikk írásakor ezeket a tápegységeket még csupán néhány hete használjuk, de teljesen meg vagyunk elégedve velük.

A ház

A ház kiválasztásakor kiengedtük az összes féket, és ráböktünk egy *Silverstone TJ07-S* márkajelzésű darabra, amit potom 365 dollárért adnak. Ez ugyebár mindennek nevezhető, csak olcsónak nem, viszont megéri a befektetést. Először is nagy. Nagyobb, mint bármely más torony kivitelű ház, amit valaha volt szerencsénk kipróbálni. Bőven van benne hely bővítőkártyáknak, és a kábelezésnek. Persze ha valaki olyan házat keres, ami az asztalon is jól mutat, nem csak mellette, hát akkor ez nem az.
A merevlemezeket két kivehető keretbe lehet beszerelni, amelyek mindegyikét egy-egy 120 mm-es ventilátor hűti. A ház tetején van két további 120 mm-es ventilátor, hátul pedig két 92 mm-es, amelyek befelé áramoltatják a levegőt. A rendszer kifejezetten halknak mondható, különösen ha figyelembe vesszük, hogy a processzor, a tápegység és a lapkakészlet hűtőventilátorán túl hat másik ventilátor dolgozik benne. Ami pedig a hőmérsékletet illeti, biztosak lehetünk benne, hogy azzal nem lesz gond, mindezt pedig ráadásul anélkül értük el, hogy drága és nehezen szerelhető vízhűtéssel kellett volna foglalkoznunk.
A ház elején van egy lehajtható panel, amely mögött audio, *USB* és *FireWire* csatlakozókat találunk. A panel amúgy teljesen belesimul a felszínbe, a lenyitáshoz egyszerűen meg kell nyomni. Egyeseket talán zavarni fog a rugós nyitáskésleltető, vagy a nyomógomb hiánya. Nekünk ezek valahogy nem hiányoznak.
A *Reset* gombot csak úgy lehet megnyomni, ha egy vékony, hegyes, drótszerű tárggyal benyúlunk egy keskeny nyílásba. Ez megint olyasmi, amit egyesek imádni fognak az általa nyújtott védelem miatt, mások meg utálni – ugyanezért.
Aki keres, nyilván ennél olcsóbb házat is fog találni, vagy olyant, amin kényelmesebben vannak elhelyezve a kivezetések. Mi a magunk részéről ezzel a drága darabbal teljesen meg vagyunk elégedve. Nagyon könnyű benne dolgozni (köszönhetően a gigantikus méretnek), és az általa biztosított hűtés is kiváló.
Ha tehát valaki gazdaságosabb gépet szeretne magának összeállítani, rengeteg kiváló ház közül válogathat. A lehetőségek száma olyannyira nagy, hogy nem is nagyon merünk javaslatokkal előállni. Mindegyiknek megvannak persze a maga előnyei és hátrányai, akárcsak az általunk választott darabnak. A gazdaságos konfigurációhoz mi a *Thermaltake Tsunami VA3000BWA* típust választottuk, ami a legtöbb helyen alig kerül többe 100 dollárnál. Ez nem olyan igényes kivitelű, mint a korábban említett *Silverstone*, sőt mi úgy tapasztaltuk, hogy a *DVD* meghajtó beszerelése kifejezetten nehézkes, de ezért az árért összességében teljesen jónak mondható. Érdekessége, hogy van rajta egy felhajtható kezelőpanel. Ezt azok, akik szeretnek a gép tetején tartani dolgokat valószínűleg kényel-

Gazdaságos konfiguráció (közelítő árak)

- Alaplap: *ABIT AN8 32X 939* ($200 US)
- Processzor: *AMD64 4200+ Athlon X2* ($360 US)
- Tápegység: *Enermax ELT500AWT* 500 Watt ($100 US)
- Memória: egy pár *Corsair 1Gx2 TWINX2048-3200PRO* ($250 US)
- Videokártya: *eVGA GeForce 7900GT Signature 256MB* ($360 US)
- Ház: *Thermaltake Tsunami VA3000BWA* ($100 US)
- Merevlemez: *Seagate Barracuda 300GB 7200 RPM 8MB cache SATA 3.0Gb/s* ($100 US)
- DVD R+W: *PX-750A ATAPI* ($60 US)

Összesen: $1,530 US (monitor, billentyűzet és egér nélkül)

„Kedvenc" rendszer kiegészítőkkel

- Alaplap: *ABIT AN8 32X 939* ($200 US)
- Processzor: *AMD64 4200+ Athlon X2* ($360 US)
- Tápegység: *Silverstone SST-ST65ZF* 650 Watt ($170 US)
- Memória: két pár *Corsair 1Gx2 TWINX2048-3200PRO* (a négy összesen $500 US)
- Videokártyák: két *eVGA GeForce 7900GT 256MB (NVIDIA SLI)* ($678 US)
- Ház: *Silverstone TJ07-S* ($365 US)
- Merevlemez: 2x *Seagate Barracuda 300GB 7200 RPM 8MB Cache SATA 3.0Gb/s* ($200 US)
- DVD R+W: *Plextor PX-716AL/SW SATA* ($150 US)
- Monitor: *Samsung LCD 204B 20.1"* 1600x1200 ($400 US)
- Billentyűzet és egér: *Logitech Cordless Comfort Duo* ($75 US)

Összesen: $3,098 US

metlennek fogják találni, de ugyanezt tulajdonképpen a *Silverstone*-ról is elmondhatjuk, hiszen ott meg szellőzőnyílások vannak a ház tetején, amiket szintén nem tanácsos kézikönyvekkel vagy egyéb ingóságokkal letakarni.

A videokártyák

Nos igen... Ha valakit nem zavar, hogy a nyílt forrású *Linux* kernelt „összerondítsa" az *NVIDIA* zárt forrású meghajtójával, az nyugodtan használhat két *NVIDIA* kártyát *SLI* módban *Linux* alatt is. Mi egy pár 256 MB RAM-mal szerelt *eVGA GeForce 7900GT* kártyát választottunk erre a célra, ezek ugyanis szerintünk a legjobb kompromisszumot jelentik az ár/teljesítmény viszonyt vizsgálva. A két kártya együtt körülbelül 500 dollárba kerül, ami kevesebb, mint egyetlen *NVIDIA 7900 GTX* kártya ára (570 dollár). Az *eVGA* gyakorlatilag teljesen azonos kártyákat árusít, az árral csak az órajel változik. Aki például alacsonyabb árú gépet szeretne építeni, az vásárolhat egyetlen *eVGA GeForce 7900GT Signature 256MB* kártyát is. Ez ugyanaz az alaptípus, mint amit mi használtunk *SLI* módban, de magasabb az órajele, és így 360 dollárba kerül.
Természetesen nem muszáj *eVGA* kártyát vennie senkinek, a lehetőségek száma itt is meglehetősen nagy. Mi azért választottuk ezt a gyártót, mert árban nagyon széles skálán mozognak a termékei, így könnyű volt kiválasztani a nekünk legmegfelelőbbet. Ugyanakkor természetesen más *NVIDIA* kártyákat gyártó cégek termékeivel is jó tapasztalataink vannak.
Aki úgy dönt, hogy az *NVIDIA* zárt forráskódú meghajtóit használja, az győződjön meg róla, hogy az *xorg.conf* fájlban szerepel a következő sor:

```
Option  "SLI"   "Auto"
```

Szintén ajánlott alaposan átolvasni az *NVIDIA HOWTO*-t is azzal kapcsolatban, miként adhatjuk meg, hogy az analóg vagy a digitális kimenetet akarjuk-e használni. Egyes monitorok bekapcsoláskor hosszú másodperceket képesek azon elfilozofálni, hogy honnan is érkezik a videojel, ami egy idő után a legtöbbünket idegesíti.

Tippek gépépítőknek

1.
Óvatosan helyezzük be a processzort a foglalatába. Ez a művelet általában nem különösebben nehéz, az *AMD64*-es processzoroknál pedig kifejezetten egyszerű, mivel ezeknél a lábak elhelyezkedése eleve lehetetlenné teszi, hogy eltévesszük az irányt. (Ez egyébként a korábbi *AMD* processzorokra is igaz volt, de az *AMD64* esetében a dolog ha lehet még egyértelműbb.)
Ez volt a jó hír. A rossz az, hogy a hűtőborda és a ventilátor felhelyezése ugyanakkor jó esetben csak nehézkes, rossz esetben meg egyenesen rémálom. Hogy a kettő közül melyik, az több tényezőtől függ. Egyes alaplapokon a processzorfoglalat eleve igen „érdekes" helyen tud lenni, amit csak tetéz, ha a hőelvezető csövek vagy egyéb alkatrészek a rögzítőhoronyhoz képest kényelmetlen közelségben vannak. Persze az is előfordul, hogy semmi különösebb ésszerű oka nincs annak, hogy nehéz felhelyezni a hűtőbordát. Egyszerűen csak nehéz és kész.
Rengeteg gyártó forgalmaz különféle hűtési megoldásokat, így van miből válogatni. Válasszunk tehát olyan bordát, amit könnyű felhelyezni. A legjobb hűtőknek általában van egy olyan, rögzítésre szolgáló része is, amit az alaplap alá kell helyezni, vagyis az ilyen hűtő eleve csak akkor helyezhető fel, ha a lapot még nem szereltük be a házba. Általában is igaz, hogy egyszerűbb feltenni a hűtőt, ha az alaplap még „házon kívül tartózkodik", tehát az alapszabály az, hogy gondolkodjunk, mielőtt bármit cselekednénk.

2.
Vizsgáljuk meg alaposan a ház felépítését, mielőtt bármit beszerelnénk. Mi egyes esetekben (és ez most tényleg nem vicc) úgy találtuk, hogy az a legegyszerűbb, ha először a tápegységet tesszük be a helyére. Ennek pedig az az oka, hogy számos házban a táp pont azzal a hellyel átellenben található, ahova a *DVD/CD* meghajtót fogjuk beszerelni, így azt sokkal egyszerűbb lesz a helyére tenni, ha a táp már bent van, mint a tápot, ha már beszereltük a *DVD*-t. Persze ha erre csak utólag jöttünk rá, akkor sincs minden veszve, mert az esetek többségében a tápot ügyesen forgatva, kicsit zsonglőrködve azért be lehet tenni. Ugyanakkor az is igaz, hogy ha már más alkatrészek – például az alaplap – is a helyükön vannak, akkor ennek a manővernek akár katasztrófa is lehet a vége, mert letörhetünk néhány kiálló alkatrészt, vagy megkarcolhatjuk az alaplap felületét. Egy másik gyakori probléma a merevlemezek és a bővítőkártyák – például a videokártya - „földrajzi" viszonya. Találkoztunk olyan esettel is, amikor a videokártyát csak úgy tudtuk a helyére tenni, ha a merevlemezt áthelyeztük egy másik pozícióba.

3.
Ügyeljünk a ventilátorok aerodinamikailag értelmes beszerelésére, vagyis legyen olyan is, ami beszívja, meg olyan is, ami kifújja a levegőt. Azt pedig, hogy melyik ponton érdemes beszívni, és melyiken kifújni, a ház felépítése dönti el. Igyekezzünk kialakítani a lehető legjobb légáramlást. Tapasztalataink szerint elég könnyű a ventilátorokat teljesen összevissza beszerelni. Sokszor láttunk már olyant, hogy egy házban a légáramlás nem elszállította a processzorhűtők által kifújt forró levegőt, hanem visszafújta azt rájuk.

A merevlemezek

Ezen a téren – szerencsére – megint a bőség zavarával kell megküzdenünk, hiszen rengeteg kiváló *SATA* merevlemez közül válogathatunk, amelyek nem is különösebben drágák. Mi két *Seagate Barracuda 300GB 7200 RPM* meghajtót választottunk, amelyek 8 MB gyorsítótárral és *SATA 3.0Gb/s* csatlakozással rendelkeznek. Ezek a

4.
Ne felejtsük el, hogy a merevlemezek is számottevő hőt termeknek. Jelentősen meghosszabbíthatjuk tehát az élettartamukat úgy, ha ventilátorokkal levegőt fújunk rájuk. Ugyanez általában jótékony hatással van az egész gépház hőmérsékletére is. Egy jól megépített háznál erre már a tervezők gondolnak, ezért a merevlemezekhez általában külön hűtés tartozik. Egy ventilátor fújja rájuk a levegőt, egy másik pedig elviszi róluk a meleget.
Az általunk használt *Silverstone* ház történetesen nagyon jó hűtést biztosít a lemezeknek is. Ha a házban magában nincs ilyen megoldás, akkor vásárolhatunk kifejezetten a merevlemezek hűtésére tervezett eszközöket is, amelyeket a meghajtó aljára vagy tetejére kell felszerelni. Persze mielőtt megveszünk egy ilyent, előtte mindenképpen vizsgáljuk meg, hogy elfér-e az adott helyen, illetve hogy a rögzítése megfelel-e az általunk használt merevlemez geometriájának. Van úgy, hogy ezek az egyszerű mechanikai feltételek nem teljesülnek.

5.
Használjunk kábelkötözőt. A szabadon hagyott szalagkábelek és egyéb vezetékek idővel garantáltan valami galibát fognak okozni. A legegyszerűbb probléma az, amikor egy ilyen kósza „madzag” megtalál magának egy forgó ventilátort. Egyszóval ha összeszereltük és leteszteltük a gépet, és mindent rendben találtuk, akkor az utolsó művelet legyen az, hogy a kábeleket ésszerű helyeken összekötözzük. A kötegeket olyan helyeken kell kialakítani, vagy elhúzni, hogy véletlenül se érjék el a ventilátorokat, illetve ne akadályozzák a szabad légáramlást. No és persze ne feledkezzünk meg magának a kábelkötözőnek a hosszú végeiről sem. Ezeket csípjük le.

6.
A videokártyák meglepően sérülékenyek. Ha nem vagyunk elég óvatosak, egész könnyen eltörnek, vagy leválik róluk valamilyen alkatrész. Mi eddigi gépész pályafutásunk alatt legalább két videokártyát tettünk tönkre. Az egyiket ráadásul már akkor, amikor kivettük a csomagolásából. Egyszerűen rossz helyen fogtuk meg. A másik a behelyezéskor szenvedett maradandó károsodást hasonló okból kifolyólag. Az óvatlan gépszerelővel az is megeshet, hogy a kábelekkel való bajlódás közben egyszerűen kigyomlál egy kondenzátort a kártyából. Igazából nem értjük, a gyártók miért ilyen lazán forrasztják be ezeket az alkatrészeket, de tény, hogy a kitépésükhöz semmiféle gigászi erőre nincs szükség. Az ember csak hozzájuk ér, és szinte maguktól kiugranak a helyükről. Szóval csak óvatosan!

7.
Minden alaplapon van egy kapcsoló, jumper, vagy valami hasonló alkalmatosság, amivel törölni lehet a *BIOS* beállításait. Ezt a műveletet érdemes elvégezni, mielőtt először bebootoljuk az új gépet.

8.
A memóriamodulok behelyezése előtt mindenképpen olvassuk el az alaplap kézikönyvéből a vonatkozó részt. Mindhárom, általunk vizsgált lap támogatta a dual channel üzemmódot, de számos alaplapnál ez csak akkor működik, ha a modulokat meghatározott rend szerint helyezzük be.

Gazdaságos rendszer kiegészítőkkel

- Alaplap: *ABIT AN8 32X 939* ($200 US)
- Processzor: *AMD64 4200+ Athlon X2* ($360 US)
- Tápegység: *Enermax ELT500AWT* 500 Watt ($100 US)
- Memória: egy pár *Corsair 1Gx2 TWINX2048-3200PRO* ($250 US)
- Videokártya: *eVGA GeForce 7900GT Signature 256MB* ($360 US)
- Ház: *Thermaltake Tsunami VA3000BWA* ($100 US)
- Merevlemez: *Seagate Barracuda 300GB 7200 RPM 8MB cache SATA 3.0Gb/s* ($100 US)
- DVD R+W: *PX-750A ATAPI* ($60 US)
- Monitor: *Acer AL1515* ($150 US)
- Billentyűzet és egér: *Logitech Internet Pro Desktop*, *PS/2* billentyűzet és egér ($18 US)

Összesen: $1,698

paraméterek egyrészt biztosan jó működési sebességet biztosítanak, másrészt a lemezek mindhárom alaplaphoz csatlakoztathatók. Azért választottuk két egyforma lemezt, mert eleve egy *RAID 0* konfigurációt szerettünk volna kialakítani, amely hibatűrést ugyan nem biztosít, viszont még gyorsabb működést tesz lehetővé. Akinek tehát fontosabb a sebességnél az ár, az vásárolhat csupán egy ilyen lemezt is, körülbelül 100 dollárért. Ami a típust illeti, ahhoz sem kötelező ragaszkodni. Ha tehát valakinek rossz tapasztalatai lennének a *Samsung* merevlemezeivel, hasonló összegért más gyártó termékét is beszerezheti.
A *Plextornak* van egy *SATA* felülettel rendelkező *DVD R+W* meghajtója, nevezetesen a *PX-716AL/SW*. Ez a következő írási sebességekre képes: *DVD+R 16x, DVD+RW 8x, DVD-R 16x, DVD-RW 4x, CD-R 48x, CD-RW 24x, DVD+R DL 6x* és *DVD-R DL 6x*. Emellett *DVD-ROM*-ot *16x*, *CD-ROM*-ot pedig *48x* sebességgel képes olvasni. Ugyanakkor legyünk óvatosak, ha *SATA* felületű *DVD* meghajtót vásárolunk. Előfordulhat ugyanis, hogy az általunk használt terjesztésben olyan kernel van, amelyik nem ismeri fel megfelelően a *SATA* meghajtót, így a telepítésnél trükközni kell.
Mi végül egy *ATAPI* felülettel rendelkező típust, a *Plextor PX-750A*-t választottuk, elsősorban azért, mert ez mindössze 60 dollárba került. Ez persze még mindig nem kevés, tehát aki keres, az biztosan fog olcsóbbat találni. Ugyanakkor felhívjuk a figyelmet, hogy az ár és a készülék zajszintje illetve megbíz-

hatósága tapasztalataink szerint fordított arányban áll egymással.

Kiegészítők

Egy teljes rendszerhez értelemszerűen szükségünk lesz legalább egy monitorra, egy billentyűzetre és egy egérre. Ezen a téren nem nagyon szeretnénk semmiféle konkrétummal előhozakodni, hiszen ezek amolyan személyes jellegű darabok, kiválasztásuk tehát nagyon erősen függ a felhasználó személyes preferenciáitól, ízlésétől. Egyesek a széles monitorokat szeretik, másik nem. Egyesek imádják a vezeték nélküli billentyűzetet, mások gyűlölik. Egyes emberek valóban kényelmesnek találják az ergonomikusan megtervezett dolgokat, mások meg kifejezetten furcsának. Itt tehát nincs arany középút.

Ezzel együtt azért az árlista tartalmaz egy monitort, egy billentyűzetet és egeret, de csak azért, mert valahogyan jelezni szerettük volna, mekkora költségekre számíthatunk ezen a téren.

Az egér és a billentyűzet változtatása a végösszeget valószínűleg nem nagyon fogja befolyásolni, a monitor azonban már komolyabb darab. Egyszóval amit leírtunk, az csak tájékoztató jellegű, vagyis ha valaki egy szép nagyképernyős monitort választ a gépéhez, az hasonló változásra számítson az árcédulán szereplő számmal kapcsolatban is.

Mi egy *Samsung 204B* monitor mellett tettük le a voksot, mivel meglehetősen nagy képernyője van (*20''*), és egyaránt jó a felbontása (*1600x1200*), a kontrasztaránya (*800:1*), valamint a válaszideje (mindössze *5 ms*). Ezekkel a paraméterekkel mind munkára, mind játékra tökéletesen megfelel, és mindehhez csak 400 dollárba kerül.

Akinek ezen a téren az ár sokkal fontosabb tényező, mint a tetszetős műszaki paraméterek, az választhatja az *Acer AL1515* típust is, amely kisebb (*15''*), alacsonyabb a felbontása (*1024x768*), és lassúbb is (*16 ms*). Viszont ennek megfelelően az ára is jóval alacsonyabb, mindössze 150 dollárba kerül. Persze rögtön felhívjuk a figyelmet, hogy egy ilyen monitor kiváló ugyan irodai munkához, viszont egyenesen szörnyű élményben lesz részünk, ha játszani akarunk.

Ami az egeret és a billentyűzetet illeti, mi két *Logitech* kombinációt választottunk. Ez egyik a vezeték nélküli *Wireless Comfort Duo* 75 dollárért, a másik a hagyományos *Internet Pro* csomag mindössze 18 dollárért.

Ez aztán tényleg az a terület, ahol a variációk száma végtelen. Az általunk javasoltnál akár jobb egeret és billentyűzetet is kaphatunk egy kicsivel több pénzért, a *Comfort Duo* csomag azonban szerintünk a legtöbb átlagos felhasználó igényeit tökéletesen kielégíti. Ezzel együtt mi azért kicseréltük a csomagban található egeret egy *Logitech G7 Laser Cordless* darabra, amolyan úri szeszélyből. Erre aztán nyugodtan mondhatja bárki, hogy jó dolgunkban nem tudjuk mit csináljunk, mert ebben a jószágban bizony naponta kétszer kell elemet cserélni. Szerencsére eleve két lítium akkumulátorral szállítják, amelyek közül az egyik a töltőben regenerálódik, míg a másikat használjuk. Tekintettel minderre úgy gondoljuk, hogy a legtöbb embernek jó lesz a csomagban található eredeti egér is, tehát az árlistában ezt a módosítást nem tüntettük fel.

Játék az árakkal

Két rendszert állítottunk össze: egy műszaki szempontból nagyon jót, és egy kevésbé felszerelt, de egyben olcsóbb gépet. Persze akinek komoly összegek állnak rendelkezésére, az a mi jobbik gépünkből is építhet olyant, amitől már tényleg megborzong az ember, ha másért nem hát az árát látva. Bárki megteheti például, hogy a jelenleg kapható *NVIDIA* kártyák közül megveszi a két legjobbat, és azokat használja *SLI* módban. Aztán ott van a vízhűtés. Van olyan szett, amit az összes, a házban található és hőt termelő alkatrészhez csatlakoztatni lehet. A *4200+* helyett is választhatjuk a legjobb *AMD64* proceszszort. A végösszeg sokkal nagyobb lesz, de a teljesítmény is növekedni fog. Hogy mást ne is említsünk, az *AMD Athlon 64 FX60 Toledo* processzor ára alig valamivel kevesebb mint 1000 dollár.

Az általunk javasolt gazdaságosabb gépnél olcsóbb a ház, a rendszer csak egy videokártyát tartalmaz, és a memória is kevesebb. Ugyan máshol is lehetett volna spórolni, de mi igyekeztünk azokat a pontokat megtalálni, ahol a nagyobb teljesítményért jelentősen több pénzt kell fizetni. A cikk írásakor például az *AMD64 4000+* és az *AMD64 4200+* processzorok ára közti eltérés nagyjából 30 dollár. Ha viszont a *4200+* és a *4400+* árát vetjük össze, az eltérés rögtön felugrik 100 dollárra.

Amint azt korábban is említettük, az *AMD* minden valószínűség szerint át fogja strukturálni az árakat, mire ez a cikk megjelenik, tehát mindenki tájékozódjon, és válasszon az aktuális helyzetnek megfelelően (*a szerkesztő megjegyzése: a váltás azóta valóban megtörtént*).

Hasonló mondható el a videokártyák piacáról. Itt is van egy olyan pont, amelyen túl nem kapunk jelentősen nagyobb teljesítményt, de jóval magasabb szám van az árcédulán. Ilyenkor aztán vagy meg kell kötni a kompromisszumot, vagy le kell nyelni a békát. Az *NVIDIA* chipes kártyák piacán jelenleg az *NVIDIA 7900GT* sorozat az, ami egyszerre jó teljesítményű és megfizethető. Ha valaki olcsót szeretne, annak az *eVGA GeForce 6600GT 256MB PCI Express x16* típust tudjuk ajánlani 150 dollár körüli áron. Bár a 7900-as szériának a nyomába se ér, azért különösebben rossznak sem nevezhető.

Ami a monitorokat illeti, nos talán nincs egyetlen más olyan hardverelem sem, amelynél ilyen széles lenne a lehetőségek tárháza. Ezzel a témával igazán részletesen nem is foglalkoztunk, már csak azért sem, mert a monitor megválasztása a legszemélyesebb döntés egy rendszer összeállításakor. Senkinek nem tudjuk, és nem is akarjuk tehát megmondani, hogy mit szeressen. Ami pedig az árakat illeti, mindenki készüljön fel rá, hogy a személyes szempontjai alapján kiválasztott „elfogadható” és „kiváló” monitor ára között esetleg 1000 dollár eltérés lesz. Mi teljesen meg vagyunk elégedve a már említett *Samsung 204B*-vel, de pontosan tudjuk, hogy egyes olvasóink ennél sokkal kevesebbel is beérik majd, míg mások sokkal többre vágynak.

Nicholas Petreley

A Linux Journal főszerkesztője. Korábban fejlesztőként, tanárként, rendszerelemzőként és tanácsadóként dolgozott. Több mint tíz éve jelennek meg írásai a Linux operációs rendszerről.

LDAP azonosítás a gyakorlatban – egy esettanulmány (1. rész)
A kiszolgáló telepítése

Az LDAP telepítéséről sok dokumentáció jelent már meg a Linuxvilág oldalain és más helyeken is. Sokoldalúsága miatt én is elhatároztam, üzembe állítok egy ilyen nagy tudású rendszert. A beüzemelés közben persze számos nehézségbe ütköztem, ezért ez az írás talán segítséget jelent mindazok számára, akik hasonlóan nagy fába vágják a fejszéjüket...

Munkahelyemen előzőleg *NIS* azonosítást használtam. A *NIS* hibátlanul működött, de a következők miatt a csere mellett döntöttem:

1. Nem felel meg maradéktalanul napjaink biztonsági követelményeinek: *NIS* használata esetén a kódolt jelszóállományt a felhasználók egyszerűen lekérdezhetik.

2. Sok felhasználó esetén kényelmetlenné válik azoknak nyilvántartása a rendszerállományokban (*/etc/passwd* és */etc/shadow*). Munkahelyemen körülbelül 750 felhasználó, illetve munkaállomás fiók van, évente több mint 100 változik.

3. *Samba*-kiszolgálókkal a *NIS* együttműködése körülményes: Ha egyszerre többféle profiltípusú felhasználóink vannak (kötelező-, vándor- és helyi profil), ezek együttes kezelése úgy lehetséges, hogy a *Samba* kiszolgálónak a saját adatbázisára is szükség van, így duplán vannak nyilvántartva a felhasználók.

4. Ha a *Samba* a saját adatbázisát (is) használja, a *Samba PDC* (elsődleges tartományvezérlő) és *Samba BDC* (tartalék tartományvezérlő) között az adatbázis szinkronizálás nehezen oldható meg.

Az *LDAP* azonosítás mentes a fenti problémáktól, de bonyolultabb az üzembe helyezése. Az alábbiakban *Debian 3.1r1 „sarge”* verzión elvégzett telepítést mutatom be.

LDAP kiszolgáló telepítése

Az *OpenLDAP* kiszolgálót a *slapd* csomaggal lehet telepíteni. Mindjárt telepítettem az *ldap-utils* csomagot is, amelyben az *LDAP* adatbázis-kezelő parancsok vannak:

```
apt-get install slapd
➥ldap-utils
```

Telepítésekor a következőket kérdezte meg:

- A *DNS* tartomány nevét: *cegnev.hu* Ide természetesen mindenki a saját cégének *DNS* nevét gépelje. Ha szükséges, aldomain is megadható (például *egyseg.cegnev.hu*) A *DNS* névből állítja elő az *LDAP* kiszolgáló kiindulási pontját (*search base*) például *dc=cegnev,dc=hu* illetve *dc=egyseg,dc=cegnev,dc=hu*
- A szervezet nevét (tetszőleges név)
- Az *LDAP* adminisztrátor jelszavát
- Működjön-e a régi *LDAPv2* protokoll is (nem)

A kérdések megválaszolása után a *Debian* telepítette az *OpenLDAP* kiszolgálót, a válaszoknak megfelelően létrehozta az */etc/ldap/slapd.conf* beállítófájlt, és elindította a szolgáltatást. Az adatbázis fájljai a */var/lib/ldap/* könyvtárba kerültek. Az */etc/ldap/slapd.conf* beállító fájlt mindjárt módosítottam. A kiszolgáló, egy kérésre alapértelmezés szerint 500 bejegyzést jelenít meg maximum, utána bontja a kapcsolatot, így védekezve a túlterhelés ellen. Ez a beállítás nem jelent problémát amikor egy felhasználó bejelentkezik, vagy e-mail címeket keres. Amikor viszont valamennyi felhasználót szerettem volna megjeleníteni (megnéztem a `repquota -a` paranccsal kinek mennyi tárterülete van szabadon), csak az első 500 felhasználó neve jelent meg, a többinél pedig a felhasználó azonosító. Ezért a `sizelimit unlimited` sor beírásával kikapcsoltam ezt a korlátozást. Saját */etc/ldap/slapd.conf* állományom az *1. Listában* látható. A kiszolgálót leállítani, elindítani, illetve újraindítani a következő parancsokkal lehet:

```
/etc/init.d/slapd stop
/etc/init.d/slapd start
/etc/init.d/slapd restart
```

1. Lista A slapd.conf állomány tartalma

```
# 500 soros korlát kikapcsolása
sizelimit unlimited

# Ne muködjön LDAPv2
# allow bind_v2

# Schema fájlok definiálják az
adatszerkezetet
include         /etc/ldap/
➥schema/core.schema
include         /etc/ldap/
➥schema/cosine.schema
include         /etc/ldap/
➥schema/nis.schema
include         /etc/ldap/
➥schema/inetorgperson.schema

# Schema fájlok helyességének
az ellenorzését kapcsolja be
schemacheck     on

# Muködéshez szükséges fájlok
helye
pidfile         /var/run/
➥slapd/slapd.pid
argsfile        /var/run/
➥slapd.args

# Naplózási szint
(hibakereséskor kell magasabb
szintre állítani)
loglevel        0

# Modul jellemzok beállítása
modulepath /usr/lib/ldap
moduleload back_bdb

# Adatbázis beállítások
backend  bdb
checkpoint 512 30
database        bdb

# Adatbázis kiindulási
pontja(search base)
suffix          "dc=cegnev,
➥dc=hu"

# Adatfájlok elérési útja
directory       "/var/lib/ldap"

# Milyen index állományok
legyenek
index           objectClass eq

# Rögzítésre kerüljön-e minden
módosítás idopontja
lastmod         on

# Ez a beállítás csak akkor
kell, ha slave kiszolgáló is
lesz
# replogfile /var/lib/
➥ldap/replog


# Jogosutság a jelszavakhoz:
#  admin részére írási jog,
mindenkinek a sajátjára írási
jog
#  mindenki másnak tiltva a
hozzáférés
access to attrs=userPassword
        by dn="cn=admin,
➥dc=cegnev,dc=hu" write
        by anonymous auth
        by self write
        by * none

# Jogosultság a jelszavakon
kívüli adatokhoz
#  admin részére írási jog,
mindenki másnak olvasási jog
access to *
        by dn="cn=admin,
➥dc=cegnev,dc=hu" write
        by * read
```

Működését a következő módon ellenőriztem:

```
ldapsearch -x -b "dc=cegnev,
➥dc=hu" -h localhost
```

Természetesen a `dc=cegnev,dc=hu` kiindulási pontot értelemszerűen változtatva (pl. ha valakinek a telepítéskor beállított *DNS* név *penzugy.nagyceg.hu*, akkor: `dc=penzugy,dc=nagyceg,dc=hu`)

Ha a kiszolgáló nem lenne elérhető (`ldap:bind: Can't contact LDAP server (-1)`), annak egyik oka lehet, hogy a */etc/hosts.allow* fájlban nincs megfelelő engedély beállítva. Ekkor szövegszerkesztővel módosítsuk, hogy a szolgáltatás elérhető legyen. Ennél a beállításnál tapasztalatom szerint nem adható meg a gép neve, csak *IP* cím, vagy *IP* tartomány. A következő sor engedélyezi a kiszolgáló elérését a helyi gépen:

```
slapd: 127.0.0.1
```

Kiszolgáló elérésének engedélyezése helyi gépen és a ***10.0.1.0-10.0.1.255 IP*** című gépekről:

```
slapd: 127.0.0.1 10.0.1.
```

Hálózatban természetesen másik számítógépről is kereshető az adatbázis. Ekkor értelemszerűen az `ldapsearch` parancsban a `-h` után a gép *DNS* nevét, vagy *IP* címét kell megadni. Ne feledkezzünk meg az adott gép IP címére is engedélyezni az elérést!

LDAP segédprogramok beállítása

A */etc/ldap/ldap.conf* fájl módosításával lehetőség van arra, hogy a fenti `ldapsearch` parancsot és a többi adatbázis-kezelő parancsot is sokkal egyszerűbben lehessen használni. Minden olyan gépen célszerű elvégezni, melyről az adatbázist el szeretnénk érni. Szövegszerkesztővel beírtam a következő sorokat (természetesen a többi gépen a kiszolgálót futtató gép IP címét kell megadni a `HOST` után!):

```
HOST    127.0.0.1
BASE    dc=cegnev, dc=hu
```

A fenti módosítás után a következő egyszerű módon is elérhető az adatbázis:

```
ldapsearch -x
```

Biztonsági másolat készítése az adatbázisról

Leállítottam az *LDAP* kiszolgálót, majd a következő paranccsal biztonsági másolatot készítettem az adatbázisról:

```
slapcat -l ./root/
➥ldapbackup.ldif
```

Nagyméretű adatbázisnál célszerű tömöríteni a biztonsági másolatot:

```
slapcat | gzip > ./root/
➥ldapbackup.ldif.gz
```

A biztonsági másolat készítése nem kötelező, de ha van biztonsági másolat és valami baj történik az adatokkal, akkor egyszerűen helyre lehet állítani azokat, a kiszolgáló leállítása után:

```
slapadd -l ldapbackup.ldif
```

LDAP kiszolgáló futtatása megadott felhasználó nevében

A *Debian* az *LDAP* kiszolgálót a *root* felhasználó és csoport nevében futtatja alapértelmezés szerint. Ez nem biztonságos, célszerűbb egy olyan fiókot és csoportot létrehozni a kiszolgáló futtatásához aminek nincs semmi más jogosultsága. Leállítottam a kiszolgálót, majd létrehoztam *slapd* néven rendszer-fiókot és -csoportot:

```
adduser --system --group
➥--no-create-home slapd
```

A */etc/default/slapd* fájlban szövegszerkesztővel megadtam, hogy melyik felhasználó és csoport nevében fusson a kiszolgáló:

```
SLAPD_USER="slapd"
SLAP_GROUP="slapd"
```

A */etc/ldap/slapd.conf* fájlban módosítottam az argumentumfájl elérési útját olyan könyvtárra, melyre a *slapd* felhasználónak írási jogosultságot állítottam be:

```
argsfile /var/run/slapd/
➥slapd.args
```

Elvégeztem a szükséges fájl és könyvtár tulajdonos és jogosultság módosításokat, hogy a *slapd* felhasználó hozzáférjen ezekhez:

```
chown -R slapd.slapd /var/lib/
➥ldap
chmod 750 /var/lib/ldap
chown slapd.slapd /etc/ldap/
➥slapd.conf
chmod 640 /etc/ldap/slapd.conf
chown -R slapd.slapd /etc/ldap/
➥schema
chmod 640 /etc/ldap/schema/*
chown -R slapd.slapd /var/run/
➥slapd
chown -R slapd.slapd /var/
➥spool/slurpd
```

LDAP kiszolgálót elindítottam, majd ellenőriztem a működését a következő paranccsal:

```
ldapsearch -x
```

Ha a kiszolgálót az előbbiekben leírtak szerint parancssorból telepítettem, akkor rendben működött a *slapd* tulajdonos és csoport nevében. Azonban ha *dselect*-el telepítettem, csak a *root* nevében volt hajlandó működni. Az ok, hogy a *dselect* „ajánlottan” feltelepítette a *db2.4-util* csomagot is, amiben *BDB* adatbáziskezelő segédprogramok vannak. Ha a csomag fel van telepítve, akkor a *slapd* indító programja (*/etc/init.d/slapd*) egy *BDB* adatbázis visszaállító parancsot is lefuttat (`db_recover`) és a */var/lib/ldap* könyvtár fájljainak egy része vissza kerül a *root* tulajdonába, így a *slapd* felhasználó azokat nem tudja használni. Kétféle megoldást találtam. Az egyik lehetőség, az */etc/default/slapd* fájlban a helyreállítás kikapcsolása:

```
TRY_BDB_RECOVERY=no
```

Én azt a megoldást választottam, hogy az */etc/init.d/slapd* indítóprogramba megkerestem az adatbázis-visszaállító sort (140. sor körül) és utána szövegszerkesztővel beírtam egy tulajdonos-módosító parancsot:

```
echo -n "running BDB recovery"
for dbdir in $bdb_envs; do
  reason="`$DB_RECOVER_CMD -eh
  ➥$dbdir 2>&1`'  ||  \
    db_recover failed $dbdir
  # Új sor a slapd felhasználó
    tulajdonába adására
  /bin/chown -R slapd.slapd
  ➥/var/lib/ldap
done
```

Az LDAP admin jelszó módosítása

Hagyományos módon az *LDAP* adminisztrátor nevét és jelszavát az */etc/slapd.conf* fájlban szokás megadni, a `suffix` sora után beírva a következő sorokat:

```
suffix          "dc=jjsoft,
➥dc=hu"

rootdn "cn=admin,dc=cegnev,
➥dc=hu"
rootpw jelszo
```

Ilyen esetben az itt megadott jelszó átírásával és a kiszolgáló újraindításával lehet megváltoztatni a jelszót.
A másik lehetőség (ahogy az `ldapsearch -x` parancs kiadásakor látszik is), az *admin* fiók adatbázisban történő tárolása. Ennek módosításhoz létrehoztam szövegszerkesztővel egy *adminpasswdmod.ldif* fájlt a következő tartalommal:

```
dn: cn=admin,dc=cegnev,dc=hu
changetype: modify
replace: UserPassword
UserPassword: ujjelszo
```

Majd az `ldapmodify` paranccsal módosítottam a jelszót. A parancs kiadása után kétszer még bekéri az *admin* aktuális jelszavát:

```
ldapmodify -x -D "cn=admin,
➥dc=cegnev,dc=hu" -W -f
➥adminpasswdmod.ldif
```

Lehetőség van mind a két esetben, a jelszó titkosított megadására is. Titkosított jelszó a `slappasswd` paranccsal állítható elő, mely többféle titkosítást ismer. Én *SSHA* titkosítású jelszót állítottam elő, és a parancs kimenetét egy `jelszo` nevű fájlba írattam:

```
slappasswd -h {SSHA} > jelszo
```

A fájlból szövegszerkesztővel másoltam át, ezért az `adminpasswdmod.ldif` fájl utolsó sora a következő képen módosult:

```
UserPassword: {SSHA}Kaj3ZHujK
➥c9iE427TFhpoI8HnA
```

Adatok hozzáadása, módosítás és törlése

Ha valaki meglévő *UNIX*, illetve *Samba* fiókokat vagy csoportokat kíván hozzáadni, vagy meglévőket átvinni, akkor ezt a bekezdést ugorja át, ezeket a továbbiakban ismertetem. Ezt a lehetőséget én is csak a működőképesség tesztelésére használtam.
Mind a három esetben szövegszerkesztővel létre kell hozni egy megfelelő tartalmú *.ldif* fájlt, majd a megfelelő *LDAP* adatbáziskezelő paranccsal a fájl tartalmával a műveletet végrehajtani.
A hozzáadást a következő paranccsal végezhetjük el:

```
ldapadd -x -D "cn=admin,
➥dc=cegnev,dc=hu" -W -f
➥ldapadd.ldif
```

2. Lista Csoport és fiók hozzáadásánál az ldapadd.ldif fájl tartalma

```
#  Csoportok tárolására
alkalmas egység létrehozása
Groups néven
dn: ou=Groups,dc=cegnev,dc=hu
objectClass: organizationalUnit
ou: Groups

#  Felhasználó fiókok
tárolására alkalmas egység
létrehozása Users néven
dn: ou=Users,dc=cegnev,dc=hu
objectClass: organizationalUnit
ou: Users

#  users nevu csoport
létrehozása. A gidnumber értéke
a csoport azonosítója
dn: cn=users,ou=Groups,
➥dc=cegnev,dc=hu
objectClass: posixGroup
cn: users
gidnumber: 1100
description: Felhasznalok
➥csoportja

# nagypeter felhasználó fiók
hozzáadása, uidNumber értéke a
fiók azonosítója
dn: uid=nagypeter,
➥ou=Users,dc=cegnev,dc=hu
uid: nagypeter
cn: nagypeter
givenName: Peter
sn: nagypeter
objectClass: person
objectClass: organizational
➥Person
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
userPassword: {SSHA}A2414Ev0j
➥UGP$Ai8HbIlAsLwUZ4NtZtmTD1
shadowLastChange: 13016
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/bash
uidNumber: 1000
gidNumber: 1100
homeDirectory: /home/nagypeter
gecos: Nagy Peter,,,
mail: nagypeter@cegnev.hu
```

Ha módosítani szeretnénk a felhasználó e-mail címét, akkor a ***ldapmodify.ldif*** fájl tartalma így néz ki:

```
dn: uid=nagypeter,
➥ou=Users,dc=cegnev,dc=hu
changetype: modify
replace: mail
mail: ujcim@cegnev.hu
```

A módosításra szolgáló parancs:

```
ldapmodify -x -D "cn=admin,
➥dc=cegnev,dc=hu" -W -f
➥ldapmodify.ldif
```

Ha törölni szeretnénk a felhasználót, akkor az ***ldapdelete.ldif*** fájl tartalma:

```
dn: uid=nagypeter,
➥ou=Users,dc=cegnev,dc=hu
changetype: delete
```

A törlést a következő paranccsal hajthatjuk végre:

```
ldapmodify -x -D "cn=admin,
➥dc=cegnev,dc=hu" -W -f
➥ldapdelete.ldif
```

A parancsok végrehajtódását a szokásos módon, az ***ldapsearch -x*** paranccsal lehet ellenőrizni.

Szolga LDAP kiszolgálók telepítése

A biztonság kedvéért létrehoztam ***slave LDAP*** kiszolgálókat is, melyek a ***master*** kiszolgálón lévő adatokról tárolnak másolatot. Így ha leáll a ***master*** kiszolgáló, akkor is működik a hálózat. A beállításokat úgy végeztem, hogy csak a ***master*** kiszolgálón lehessen adatokat módosítani. ***Slave*** kiszolgálót az elsődleges (***PDC***), a tartalék (***BDC***) ***Samba*** tartományvezérlőkre, valamint a levelező kiszolgálónkra telepítettem. Teljes egészében az előbb leírtak szerint végeztem telepítésüket, de a ***master*** kiszolgálón módosítottam a beállítást és ***slave*** kiszolgálókon is eltérés van.
A ***master*** kiszolgálón a következő módosításokat végeztem el.
Létrehoztam egy ***replicator*** nevű felhasználót, akinek a nevében a kiszolgálók közötti automatikus szinkronizálás (replikáció) fog végrehajtódni minden egyes alkalommal, ha a ***master*** kiszolgálón változás történik. Ha éppen nem működik valamelyik ***slave*** kiszolgáló a változás idején, akkor a ***master*** kiszolgáló megjegyzi a szükséges változtatást és akkor frissíti, amikor először elérhető lesz.
A felhasználót az előbbiekben ismertetett módon hoztam létre az `ldapadd` paranccsal.

A ***replicatoradd.ldif*** fájl tartalma:

```
dn: cn=replicator,
➥dc=cegnev,dc=hu
objectClass:
➥simpleSecurityObject
objectClass: organizationalRole
cn: replicator
description: LDAP replicator
userPassword: jelszo
```

A fájl tartalmát az adatbázishoz adtam:

```
ldapadd -x -D "cn=admin,
➥dc=cegnev,dc=hu" -W -f
➥replicatoradd.ldif
```

Létrehoztam a */var/log/ldap* könyvtárat:

```
mkdir /var/log/ldap
```

A ***master*** kiszolgálót leállítottam, majd módosítottam az */etc/ldap/slapd.conf* fájl tartamát:

```
# Beállítottam a replikációs
napló fájl helyét
replogfile   /var/log/ldap/
➥replog.log

# replikáció beállítása. Több
slave kiszolgáló esetén
#  mindegyiknek létre kell
hozni egy ilyen bejegyzést!
# A host= után a slave
kiszolgáló IP címe, vagy DNS
neve van, majd a portja
# Figyelem!!!! Itt a
titkosított jelszó nem muködik!
replicahost=192.168.1.101:389
  bindmethod=simple
  binddn="cn=replicator,
  ➥dc=cegnev,dc=hu"
  credentials=
  ➥replicator_jelszava
```

```
# Módosítottam még a
hozzáférési jogokat is úgy,
#   hogy a replicator is
olvashassa a jelszavakat
access to attrs=userPassword
  by dn="cn=admin,dc=cegnev,
  ➥dc=hu" write
  by dn="cn=replicator,
  ➥dc=cegnev,dc=hu" read
  by anonymous auth
  by self write
  by * none

access to *
        by dn="cn=admin,
        ➥dc=cegnev,dc=hu"
        ➥write
        by * read
```

A *master* kiszolgálót ezután újraindítottam.
Slave kiszolgálókon végzett beállítások a következők voltak:

A slave kiszolgálókon is létrehoztam a *replicator* nevű felhasználót, az előbbiekben leírt módon.
Kiszolgálót leállítottam, majd módosítottam az */etc/ldap/slapd.conf* fájl tartamát:

```
#  A replog sort nem szabad
bekapcsolni!

# Kinek a nevében történik a
replikálás és a master LDAP
címe, vagy DNS neve
updatedn  "cn=replicator,
➥dc=cegnev,dc=hu"
updateref
➥"ldap://192.168.1.100"
```

Módosítottam még a hozzáférési jogokat is úgy, hogy a *replicator* módosíthatja, az *admin* pedig csak olvashatja a jelszavakat:

```
access to attrs=userPassword
  by dn="cn=admin,dc=cegnev,
  ➥dc=hu" read
  by dn="cn=replicdator,
  ➥dc=cegnev,dc=hu" write
  by anonymous auth
  by self write
  by * none

access to *
        by dn="cn=replicator,
        ➥dc=cegnev,dc=hu"
        ➥write
        by * read
```

A kiszolgálót újraindítottam.
Fontos, hogy a *replikálás* beállítása előtt a *master* és a *slave* kiszolgálók adatbázisának a tartalma egyforma legyen!
Ez biztosítható olyan módon is, hogy az ismertetett `slapcat` paranccsal a *master* kiszolgálóról készített biztonsági másolatot a `slapadd` paranccsal a *slave* kiszolgálókra másoljuk.

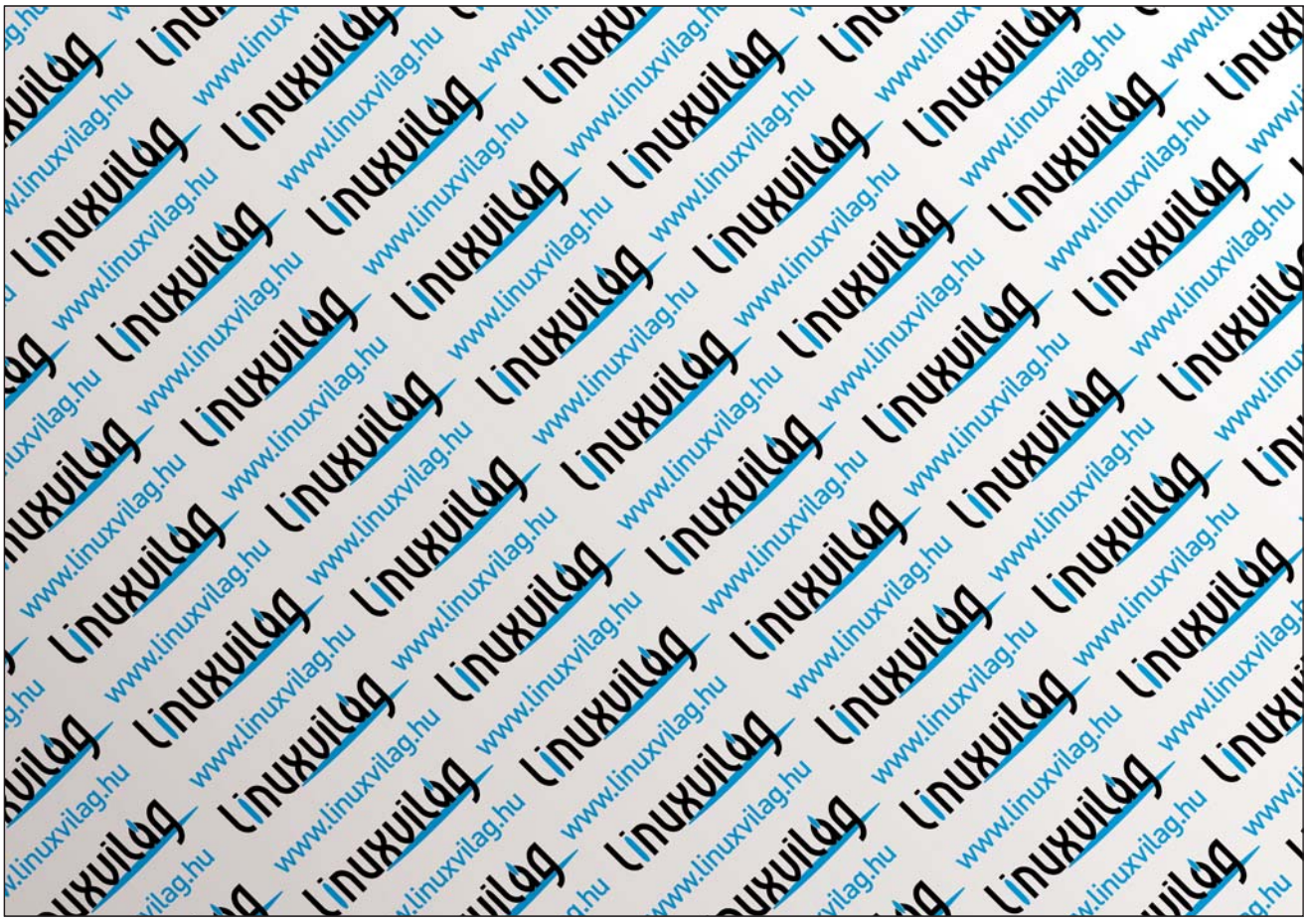
A második részben a *Linux* kliensek beállítását mutatom be.

Jászberényi József

Szeret biciklizni, kirándulni, olvasni, sörözni és szabadban főzni.

A stratégiai játékoktól a műszaki CAD programokig sok minden érdekli. Legtöbbet szerverprogramokkal foglalkozik és néha mérgelődik.

(jaszberenyij@pattantyus-gyor.sulinet.hu)



IPCop – Egy hálózsaru feljegyzései

Tilthatunk és engedélyezhetünk – például adott honlapok látogatását, s kis odafigyeléssel megelőzhetjük a betörési és fertőzési kísérleteket, ha feltelepítjük a megfelelő kiegészítő modulokat, vagy kézzel megírjuk a megfelelő szabályokat.

Kezdeti örömünket (működik a tűzfalszoftver!) gyorsan elronthatja az aggodalom: meddig marad ez védhető és jól prosperáló rendszer? A jó hír, hogy ez nagymértékben rajtunk múlik, a rossz pedig, hogy nem csak rajtunk. Mind a hálózati cím, mind a tevékenységek „ujjlenyomata" hamisítható, csakúgy, mint a felhasználók kiléte. Nem kell elkeseredni, inkább próbáljuk meg átgondolni, mi a teendőnk ahhoz, hogy optimálisan működjön a hálózat, s minél kisebb támadási felületet hagyjunk a rosszindulatú próbálkozóknak.
A következőkben néhány háttérinformációval, működésbeli sajátossággal foglalkozunk. Ezek módosítgatása első ránézésre kevésbé tűnik könnyed és egyszerű mozdulatnak, mint a grafikus felületen beállítani a programok viselkedését. Ne csüggedjünk, majdnem mindnek van kevésbé kifinomult, de könnyebben használható grafikus alternatívája. Ha kicsit megismerkedünk a parancssori változatokkal, látni fogjuk, mennyi terhet levesz vállunkról a könnyen beállítható ***IPCop*** és a hozzá írott sok beépülő modul.

Támadás kívülről

A tűzfalon kívüli veszélyforrások jelentik sokak szemében a legnyilvánvalóbb fenyegetést, noha egy belső hálózaton garázdálkodó egyén sokszor nagyságrendekkel nagyobb károkat képes okozni. Éppen ezért a támadások kivédésére elsődlegesen egy megfelelő biztonsági szabályzatot érdemes megfogalmazni, majd azt következetesen alkalmazni. Ezek után már elkezdhetünk azon is aggódni, vajon nem akar-e valaki jogosulatlanul, netán rosszindulatúan hozzáférni erőforrásainkhoz?
A tűzfal általában a legelső védelmi vonalat képviseli belső hálózatunk és az internet „vad világa" között. Normális esetben a tűzfal – akár szoftveres, akár hardveres megoldás – tartalmának értéke nem fogható a belső hálózaton található szerverek és kliensek információs bázisaihoz, tehát nem magát a tűzfalgépet kell félteni, hanem a védendő objektumokat. Azt azonban figyelemmel kell kísérni, hogy nem vált-e korrupttá a tűzfal (betörés, fertőzés, hardver- és szoftverhiba stb.), és történt-e betörési vagy kitörési kísérlet, s mivel esetünkben az ***IPCop*** útvonalválasztó (***router***) is egyben, korántsem mindegy, hogy valaki hozzányúlt-e a beállításaihoz. Előfordulhatna például, hogy egy rosszindulatú betolakodó hozzáfért rendszerünkhöz, s az útvonalválasztó tábla manipulálásával a kimenő és bejövő hálózati csomagokat az általa felügyelt számítógépre irányítva lehallgatja kommunikációt.
Ez ellen szerencsére a rendszer többszintű védelmet biztosít, s magunk is ellenőrizhetjük a beállításokat.

Csomagszűrés tűzfal módra

Nagyon leegyszerűsítve a ***TCP/IP*** protokoll szerint zajló – például internetes – kommunikációról azt mondhatjuk, hogy az alkalmazások kiválasztanak egy számmal azonosítható nyitott kaput a rendszeren – ezek az ún. portok – s ezen keresztül áramlanak az adatok. Ha ezt a kaput bezárjuk, az alkalmazás vagy nem képes ki- s bejuttatni a hálózati csomagokat, vagy kénytelen magának más portot választani – ha felkészítették erre az eshetőségre. 0 és 1024 között találhatók az úgynevezett jól ismert portok, ezek és egyéb tipikusnak mondható azonosítókat s a hozzájuk tartozó protokoll megnevezését megtalálhatjuk a ➲ ***www.iana.org/assignments/port-numbers*** címen.
Kernelszintű védelemmel rendelkeznek a Linux disztribúciók, a 2.4.x-es verziószám utáni változataikban már bizonyosan az ***Iptables*** elnevezésű változattal találkozhatunk; ezek a hálózati forgalmat „láncokba" fűzve, azaz szabályrendszerek szerint terelik. Három ilyen alapvető lánc a `FORWARD`, `INPUT` és `OUTPUT`, de ha kiadjuk az

```
iptables -L
```

utasítást, láthatjuk, hogy az ***IPCop*** ennél többet is használ.
A láncokon folyamatosan keresztülhaladva a megadott szabályok szerint történik annak elbírálása, mit kell tenni az adott fejlécet tartalmazó hálózati csomaggal. Ha nincs illeszkedő szabályrendszer, akkor az adott lánc alapértelmezett beállítása szerint engedélyezi (`ACCEPT`) vagy hibaüzenettel eldobja (`REJECT`) vagy csak csendben eldobja (`DROP`). Mindezt naplózással (`LOG`) illetve egyéb irányítással (`SNAT`, `DNAT`, `MASQUERADE`) is kezelhetjük, ráadásul a láncokat táblákká összefogva kezeli a rendszer, de ne bonyolódjunk jobban bele, nézzük csak az alapértelmezett csomagszűrést, a ***Filter*** táblát.
Ha a semmiből kellene tűzfalat készítenünk egy ***Linuxot*** futtató számítógép segítségével, akkor jó esetben úgy vágnánk neki, hogy minden letiltunk, majd a legszükségesebb szolgáltatáso-

kat engedélyezzük. Ez az `iptables` program segítségével nagyjából a következőképpen menne. Alapértelmezett irányelvként megadtuk a csomagok csendes eldobását:

```
iptables -P FORWARD DROP
iptables -P INPUT DROP
iptables -P OUTPUT DROP
```

Engedélyezzük a névfeloldást az első hálózati kártya esetében kívülről:

```
iptables -A INPUT -p udp -i
➥eth0 --sport 53 -j ACCEPT
```

Megtiltjuk, hogy az első hálózati kártya ping utasításokra reagáljon:

```
iptables -A INPUT -p icmp -d
➥-I eth0 -j DROP
```

És így tovább. Legutolsó utasításunkhoz hasonló eredményt kapunk, ha az *IPCop* grafikus felületén a *Tűzfal* fül alatti *Firewall options* -t választva beállítjuk az *icmp* válaszok tiltását a külső, vagy akár az összes hálózati kártyára vonatkoztatva, s ezáltal kívülről „láthatatlanná” tehetjük a tűzfalat, feltéve, hogy nem használunk olyan programot, mely igényli a *ping* utasításra visszaküldendő pozitív válaszok meglétét.
A csomagszűrés jó beállításaival elkerülhető a túlterheléses támadás, valamint sok manipuláció. Akit érdekel az *iptables* megoldások működése, a ➲ *iptables-tutorial.frozentux.net* oldalon sok hasznos információt gyűjthet e témában. Ritkán ugyan, de előfordulhat, hogy kézzel kell szerkesztenünk szabályokat – ilyen szituáció lehet a virtuális magánhálózat kialakítása – de az esetek túlnyomó többségében nem kell hozzányúlnunk beállításaihoz. Az *IPCop* –ot jól felvértezték, s ha külső program nem rontja el az átgondolt láncolatok sorozatát, erős védelmezőnk lehet.

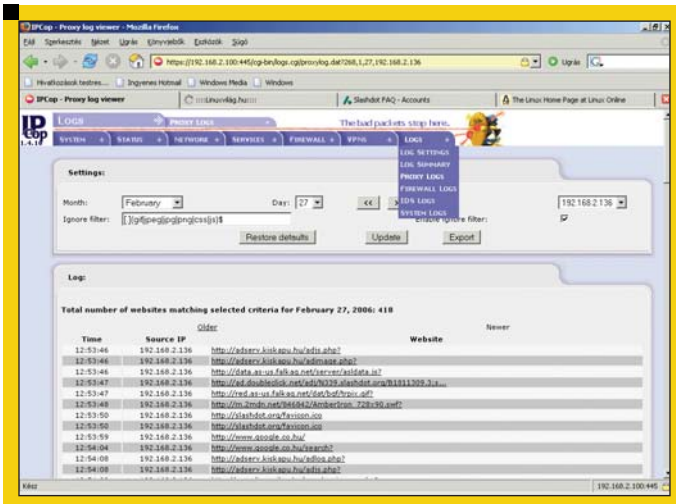
Ne hagyjuk leomlani a falat

Nagyon sok támadási kísérlet nem hozzáértőktől származik, hanem unatkozó, különböző sebeshetőségek után kutató programokat futtató emberektől, akik többnyire csak rövid szkriptprogramok elindításával próbálnak rést találni a rendszereken. Ellenük sokszor már az is segíthet, ha nem szokványos beállítá-

```
Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2006-02-26 17:37 CET
Initiating SYN Stealth Scan against ipcop.localhost (192.168.1.1) [1663 ports] a
t 17:37
Discovered open port 53/tcp on 192.168.1.1
Discovered open port 80/tcp on 192.168.1.1
Discovered open port 800/tcp on 192.168.1.1
Discovered open port 81/tcp on 192.168.1.1
Discovered open port 222/tcp on 192.168.1.1
Discovered open port 445/tcp on 192.168.1.1
The SYN Stealth Scan took 0.24s to scan 1663 total ports.
Host ipcop.localhost (192.168.2.100) appears to be up ... good.
Interesting ports on ipcop.localhost (192.168.2.1):
(The 1657 ports scanned but not shown below are in state: closed)
PORT    STATE SERVICE
53/tcp  open  domain
80/tcp  open  http
81/tcp  open  hosts2-ns
222/tcp open  rsh-spx
445/tcp open  microsoft-ds
800/tcp open  mdbs_daemon
MAC Address: 00:40:AF:53:D1:11 (Cameo Communications)

Nmap finished: 1 IP address (1 host up) scanned in 0.961 seconds
               Raw packets sent: 1665 (66.6KB) | Rcvd: 1664 (76.5KB)
```

1. ábra Nyitott portok után kutatunk

sokat használunk – például ilyen az *ssh* 22-es port helyett a 222-es alkalmazása az *IPCopban* –, nem hagyunk nyitott portokat, és megpróbáljuk naprakészen tartani a rendszert. Egy felkészültebb támadóval szemben ennél azért jóval többre is szükség lehet, így például hosszú és bonyolult jelszavakat használjunk s ahol lehet inkább nyitott kulcsú azonosítást, lehetőség szerint az összes hálózati forgalmat titkosítsuk (például *ssh* alagútba küldve), figyeljük a rendszer állapotát, és a szükséges minimumnál több szolgáltatást ne engedjünk futni, és csak akkor engedélyezzünk távoli hozzáférést, amikor használjuk is. A vírusok, kártevők, trójai programok, kéretlen levélküldözgetők mindkét kategóriába tartoznak, mert egyfelől a mechanikus ismétlődést és gépi intelligenciát könnyebb leleplezni, másfelől egy még fel nem fedezett biztonsági rés ismeretével és új áthatolási módszerekkel felvértezve akár belső hálózatunkról kifelé is próbálkozhatnak jutni, s adatokat továbbítani.
Sok biztonsági rés, puffertúlcsordulásos támadás leírását megtalálhatjuk a ➲ *packetstorm.linuxsecurity.com* honlapon. Utóbbiak ellen részleges védelmet jelenthetnek majd az újabb ezirányú hardverfejlesztések, illetve a jól megválasztott, frissített programcsomagok.
Csábíthat minket például az a tény, hogy egy rugalmasságáról híres *Linux* rendszer fut a hardveren, amelyre akár *Cups*, *Samba* stb. csomagot is telepíthetünk (higgyük el, van aki megteszi) belső hálózatunk fájl- és nyomtatószerver igényeinek kielégítésére – ha nincs más szabad szerverünk, ám óvakodjunk ezektől a látszatmegoldásoktól, mert minden egyes programmal nő a hibalehetőség és a javítatlan rések előfordulási esélye.
Az *IPCop* rendelkezik egy beépített betörésdetektáló szolgáltatással, mely a közismert *Snort* program egy verzióját tartalmazza. Bekapcsolni a *Szolgáltatások* fül *Behatolás Figyelés* menüpontban lehet. Nem árt tisztában lenni azzal, hogy ez általában nem a támadáselhárítást, hanem a megelőzést s az utánkövetést hivatott szolgálni. Ha figyelemmel kísérjük a betörési kísérleteket, talán még idejében be tudjuk zárni a kapukat a hívatlan látogatók előtt.
Nap mint nap készülnek új biztonsági résekről, próbálkozásokról szóló leírások, hibajavítások és sok egyéb, e témát érintő fejlesztés. Ha térítésmentesen regisztráljuk magunkat a ➲ *www.snort.org* honlapon s a kapott ún. *Oink* kódot beírjuk a grafikus felület megfelelő rovatába, lehetőségünk nyílik a program fejlesztőitől a tűzfalon keresztül megkapni a legújabb szabályrendszereket, s ezáltal a vakriasztások számát csökkenteni, a potenciális támadásokat pedig pontosabban előrejelezni. Jellemző lehet például, ha *Naplók* fül *BFR naplók* menüpontját böngészve egy rövid időközön belül ugyanarról a címről több portszámot próbálgatva gyanúra okot adó, naplózott tevékenységgel találkozunk, ilyenkor lehetséges, hogy a nyitott portok után kutat valaki vagy valami.
Ugyanígy leleplezhetjük azon kártevő programokat is, melyek kétségbeesetten próbálnának kijutni a rendszerből



2. ábra Vajon mi érdekelte ezt a felhasználót?

s fertőzni más gépeket is, de a tűzfal zárt ajtajairól visszapattannak.
Ha rákattintunk a ***BFR napló*** bejegyzései között található *IP* címekre, a program megpróbálja kinyomozni, kit is takarhat az adott cím. Ez persze nem jelenti azt, hogy a rosszindulatú próbálkozó az adott gép gazdája, lehet egy nyilvános ***proxyszerver*** vagy egy fertőzött gép is. Ha a ***SID:*** felirat melletti azonosítószámra kattintunk, a ***Snort*** honlapjára ugorva részletesebb leírást láthatunk az adott bejegyzésről. Érdemes néha kipróbálni a rendszert kívülről magunknak is: ha van lehetőségünk egy másik ***Linuxot*** futtató gépről ***nmap*** programot használni (legegyszerűbben `nmap -vv 192.168.1.1` feltéve, hogy előbbi a szerver belső hálózati címe, és a másik géppel látják egymást), megtudhatjuk, milyen nyitott portok találhatók a rendszerben. Ugyanezt elméletileg megtehetjük a külső (***RED*** interfész) *IP* cím végigpásztázásával is, de ezt már akár aggályosnak találhatja internetszolgáltatónk, mivel ő nem láthatja szándékainkat, csak egy ***portscan*** kísérlet nyomait.
Betörésdetektálásról angol nyelven olvashatunk például a ➲ *www.sans.org/resources/idfaq/* címen, magyarul pedig érdemes utánanézni a ➲ *wigwam.sztaki.hu* oldalán. Elszántabbak akár csapdát is állíthatnak a nem kívánt látogatóknak, összekapcsolva a tűzfalat egy külső betörésdetektáló rendszerrel: ilyen megoldásról olvashatunk a ➲ *hogwash.sourceforge.net/docs/baitnswitch.html* címen.

Barát és ellenfél a falon belül

Ha nem elég számunkra a csomagszűrés által nyújtott védelem és korlátozás, s szeretnénk gyorsítani valamelyest a böngészésen is, akkor eljött az ideje a ***proxyszerver*** funkció aktiválásának. Mint minden ilyen szolgáltatást, grafikus felületen ezt is a ***Szolgáltatások*** fül alatt érjük el. Ha kipipáljuk a ***Átlátszó Greent***, akkor egy úgynevezett transzparens proxyt üzemelünk be, mely a kliensek átállítása nélkül, számukra gyakorlatilag láthatatlanul üzemel. Így ők nem közvetlenül az interneten található webszerverrel, hanem a közbeiktatott proxyszolgáltatással tartják a kapcsolatot – az ***Upstream proxy*** választása esetén a proxyszervereket lehet egymással, például az internetszolgáltatóéval összefűzni –, ezáltal beazonosíthatóbbá válik a hálózati forgalom, és a sűrűn látogatott oldalak gyorsítótárba helyezésével csökkenhet a sávszélességigény s nő a böngészés sebessége.
A ***proxy*** működésének legjobban úgy tudjuk nyomon követni, ha a ***Naplózás bekapcsolva*** opciót kipipáljuk, s lementjük, majd átmegyünk a ***Naplók*** fülre, s a ***Proxy naplók*** menüpont alatt látható az adott napi felhasználásból származó információ. Ha csak egy adott felhasználót (értsd: gépet ill. *IP* címet) szeretnénk „tetten érni”, akkor a jobb felső sarokban található legördíthető menüpontban a ***Minden*** helyett válasszuk ki a megtekinteni kívánt címet.
A ***proxy*** beállítása után – ha nem akarunk kiegészítő programokat letölteni – legegyszerűbben az hozzáférést szabályzó úgynevezett ***ACL***-listák szerkesztésével tudjuk szűrni a látogatható tartalmakat. Alkalmazhatunk hasonló mechanizmust, mint például sok levelezőszerver: felveszünk feketelistás és fehérlistás felhasználókat illetve felhasználási módokat, s míg előbbiek teljes tiltást kapnak, utóbbiak korlátlan hozzáférést élvezhetnek, s mindezt fűszerezhetjük az éjszakai használat tilalmával. Ennek menete a következő:
Lépjünk be ***ssh***-val az ***IPCopba***, majd egy szimpatikus könyvtárat kiválasztva – például */usr/share* – hozzunk létre egy `tiltolista.txt` nevezetű fájlt. Minderre legegyszerűbb a ***vi*** szövegszerkesztőt használni, de ha kevésbé vagyunk gyakorlottak, megteszi a rendszeren található ***pico*** is.
A fájlba soronként írjuk bele a tiltani kívánt honlap címét – például a nem kívánt tartalmat biztosítókat, a reklámszervereket, a vírusgyanúsakat stb. – s mentsük el. Ugyanígy írjunk egy `szabadlista.txt` állományt, s ebbe írjuk bele azokat az *IP* címeket (értelemszerűen ennek elsődlegesen a belső hálózatunk megbízható gépeinek címét illik tartalmaznia), s azt is mentsük.
Ezek után írjuk át az ***squid*** (figyeljünk rá, hogy nem az */etc/squid* könyvtárban találjuk) konfigurációs beállításait, azaz a */var/ipcop/proxy/squid.conf*-ot. Szúrjuk be az `acl` kezdetű sorok közé a következőt:

```
acl tiltolistank url regex
➥"/usr/share/tiltolista.txt"
acl szabadlistank url regex
➥"/usr/share/szabadlista.txt"
acl ejszaka time 22:00-05:00
```

Ezzel tudatjuk a rendszerrel, hogy az adott fájlok tartalmát internetes címként olvasva a későbbiekben megfogalmazott szabályt érvényesítse reá nézve, az utolsó bejegyzés arról gondoskodik, hogy éjszaka ne használják a rendszert.
Keressük meg a `http_access` kezdetű sorokat, s oda a következő (a tényleges tiltást illetve engedélyezést ez váltja ki) információt helyezzük el egy új sorban:

```
http_access deny tiltolistank
http_access accept
➥szabadlistank
http_access deny ejszaka
```

Arra is figyelnünk kell, nem történt-e más szabálymeghatározás már korábban, például az általunk tiltólistára vett *IP* címekből nincs-e explicit engedélyezve vagy a megbízhatónak ítéltekből tiltva valamelyik, mert ez esetben legegyszerűbb az ellentmondó szabályokból a nem kívántat törölni. Szerencsénkre több kiegészítő alkalmazás is készült, mely arra hivatott, hogy a fáradságos gépelést kiváltsa, és az opciók kiválasztását grafikus felületű navigációval segítse. Ezen alkalmazások egyike az ***Advproxy*** (➲ *www.advproxy.net*), vagy például a *Cop+* csomag (➲ *home.earthlink.net/~copplus/*) is hathatós segítséget tud nyújtani a *proxyszerver* felhasználóbarát módon történő testre szabásához. Néha még a csomagszűrés és *proxy* együttes használata is kevésnek bizonyulhat. Az egyenrangú hálózatként ismertté vált fájlcserélő szolgáltatások (úgynevezett *p2p* alkalmazások) térhódításával új kihívások jelentek meg egyes rendszergazdák mindennapjaiban. Ezen alkalmazások jól tudnak alkalmazkodni heterogén környezetekhez, portszámot és *IP* címet tudnak váltogatni, s tevékenységüket akár másnak álcázva is képesek működni. Ellenük megoldást jelenthetnek az alkalmazásszintű szűrők, egyikük az ➲ *l7-filter.sourceforge.net* honlapról tölthető le.

Havi nagytakarítás

Saját magunknak is okozhatunk fejfájást, ha például hagyjuk a naplózásból származó fájlokat felszaporodni, ennek megoldására használható például a `logrotate` program (ciklikussá teszi a naplófájlokat, azaz törli a megadott időpontnál régebbi bejegyzéseket). Aktuális állapotuk az *IPCop* disztribúciókban egy jól áttekinthető felületről nyomon követhető, csakúgy, mint a rendszer egyéb fontos erőforrásainak állapota. Így elkerülhető, hogy olyan és rendszerállapot-mérő programokat, mint például a

```
ps aux
netstat -p
du -hs
```

folyamatosan futtatni kényszerüljünk, mert ezek kimenetéhez hasonló, de egységes és könnyen értékelhető grafikus és szöveges információkat közöl velünk a rendszer. Ha tartunk tőle, hogy a támadó maga után eltakarítja a betörés nyomait, legjobb, ha a rendszernaplózó `syslog` fájl kimenetét egy másik számítógépre, vagy még inkább nyomtatóra irányítjuk. Így – hacsak nincs fizikai hozzáférése a nyomtatóhoz és géphez – elméletileg minden változtatási kísérlete előtt papírra tudjuk vetni a gyanús eseményeket. Ne bízzuk el teljesen magunkat, a nyomtatósor átirányításával, vagy egy másik biztonsági rés kihasználásával sajnos némi esélye még marad ennek a megakadályozására is.

Biztos hogy én építettem ezt a falat?

A rendszer konzisztenciájának vizsgálatára részben segítséget kapunk az *IPCop* grafikus felületén a frissítések letöltésekor, tehát elméletileg csekély az esélye annak, hogy az újabb hivatalos kiadások használatakor tesszük sebezhetőbbé a tűzfalat. Ugyanakkor az interneten böngészve találhatunk külső fejlesztők által írt, hasznosnak ígérkező kiegészítőket, melyeket általában ezt az ellenőrzést megkerülve vagyunk kénytelenek telepíteni. Különböző segédprogramok és szoftvercsomagok állnak rendelkezésre a rendszerek védelmének megerősítésére és vizsgálatára. A fájlok módosítását például a *Tripwire* nevű alkalmazás segítségével tudnánk megnézni, de ha nem akarunk a telepítésével bajlódni, akkor egyszerűen kiadhatjuk parancssorból az

```
md5sum * > ellenorzo
```

utasítást, mellyel az aktuális könyvtárban található fájlokról készítünk egy ún. ellenőrző összeget (***md5 hash*** néven is találkozhatunk az eljárással) az `ellenorzo` nevű fájlba. Ez lehetővé teszi, hogy észrevegyük változásukat – például egy szándékos rosszindulatú módosítást.
Ellenőrzéshez csak ezt generált fájlt kell összevetni a könyvtárban találhatókkal:

```
md5sum –check ellenorzo | grep
➥FAILED
```

Ha problémát talál kiírja. Fontos lehet még, hogy az utolsó ellenőrzőösszeg generálása óta keletkeztek-e a könyvtárban újabb fájlok (természetesen első futtatás alkalmával az `ellenorzo` fájl létrehozása miatt eggyel több a darabszám!); ezt egyszerűen a következő héjprogram segítségével tudhatjuk meg:

```
most=`ls -l | wc -l`
korabban=`md5sum -check
➥ellenorzo | wc -l`
if test $most -gt $korabban
then echo "Ennyivel több darab
➥fájl van most a
➥könyvtárunkban:"
expr $most - $korabban
fi
```

Nekünk kell eldöntenünk, mely könyvtárakat érdemes az ellenőrzésbe bevonni, s ezután akár automatizálhatjuk is egy-két egyszerűbb ***shellszkripttel***, rekurzív könyvtárbejárást alkalmazva a vizsgálatot. Érdemes figyelni a rejtett fájlokra (például `ls .[!.]*` paranccsal kiszűrhetők és elkülöníthetők a szintén ponttal kezdődő könyvtárnevektől), a *swapfile* kihagyására, esetlegesen a */tmp* és */proc* átugrására is, viszont a */dev* bevonására is, s nemcsak a keletkező, de az eltünedező fájlok is gyanút kelthetnek bennünk.
Hálózati biztonsággal, tűzfalakkal foglalkozó tanulmányok és jótanácsok bőséges lelőhelye az internet. Hasznos olvasmány a ➲ *www.cert.org/security-improvement* valamint a *wikioldalak* között is fellelhető fontosabb fogalmak magyarázata. Készüljünk fel rá, hogy sokszor egymásnak ellentmondó tanácsokat és elveket fogunk olvasni. Általában saját magunknak kell kitalálnunk az üdvözítő megoldást, mivel legjobban mi ismerjük a védeni kívánt rendszerek sajátosságait, de sokat segíthet egy olyan „kulcsrakész” rendszer, mint az *IPCop*.

Tóth Virgil Zoltán
(m_v@c2.hu)
Szoftverfejlesztő informatikus és rendszergazda, kedvence a Debian disztribúció. Szabadidejét legszívesebben felesége és szépirodalmi regények társaságában tölti. Lenyűgözőnek tartja a Linux rugalmasságát, és a vele dolgozók aktivitását.

Biztonságos adattárolás – Szoftveres RAID kialakítása Linux alatt

Manapság egyre inkább előtérbe kerül az adatbiztonság kérdése otthoni felhasználóknál is. Konkrétan az adatok tárolásának biztonsága. Ezt ideig-óráig meg lehet valósítani szalagos vagy optikai egységgel, de belátható, hogy ezek az eszközök nem tesznek lehetővé valós idejű adatmentést. Erre találták ki a RAID-e.

A *RAID* rövidítés visszafejtésére általában kétféle „megoldást” talál az ember az interneten. Az egyik szerint *„Redundant Array of Independent Disks”* a betűszó jelentése, vagyis független lemezekből álló redundáns tömb, míg a másik megfejtés a *„Redundant Array of Inexpensive Disks”*, vagyis olcsó lemezekből álló redundáns tömb. Belátható, hogy mindkét írásmód jogos.

RAID-ek típusai

A *RAID*-eknek számtalan típusa van, de otthoni felhasználó általában a *RAID0*-át használja, ha sebesség szükséges és *RAID1*-et, ha az adatbiztonság a fontos tényező. Céges környezetben használják még a *RAID5*-öt is, illetve ezen háromfajta *RAID*-ek keverékét.

Linux kernel felkészítése és a szükséges csomagok

Ahhoz, hogy tudjunk *RAID* lemeztömböt *Linux* alatt kezelni, szükségünk van a kernelszintű támogatásra. 2.6-os kernel esetén keressük meg a *Device driver* pontot, és ebből válasszuk ki a *Multi-device support* pontot, ahol aztán már kedvünkre csemegézhetünk. Itt én csak a *RAID1*-et választottam ki, ugyanis így legalább nyugodtan alhatok, az adataim biztonságban lesznek. A *RAID0*-t én nem javasolnám senkinek, hiszen ha az egyik merevlemez meghalt a *RAID*-ből, akkor az egész tömbön tárolt adatoknak viszlát. Vagyis legyen inkább két fizikai eszközünk két fájlleíró táblával.
A *RAID0*-nak csak egy esetben van előnye: ha nagy adatátviteli sebességre van szükség, például videó-digitalizáláskor.
A fordítás sikerességét úgy tudjuk ellenőrizni, hogy megnézzük az új kernellel létezik-e a */proc/mdstat* fájl. Ha igen, nyert ügyünk van.
Ezzel megvagyunk, telepítsük fel az *mdadm* csomagot, ami a *RAID* eszközök létrehozásához és karbantartásához szükséges.

Következő lépés

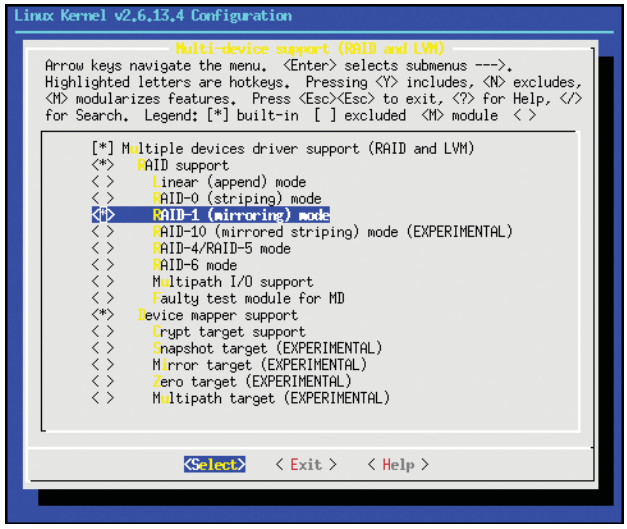
Tegyük fel, hogy van egy 80 gigabájtos merevlemezünk és ezt szeretnénk *RAID1*-be tudni. Jó, ha tudjuk, hogy a *RAID1* tárolókapacitása mindig a legkisebb merevlemez méretével lesz egyenlő. Vagyis, jelen esetben egy másik 80 gigás merevlemezre lesz szükségünk. Amennyiben csak 120-ast találnánk, úgy a merevlemez végén 40 gigabájt kárba vész – vagy két 120-ast veszünk.
Na de haladjunk csak sorjában. Mentsük le a jelenlegi merevlemez partíciós tábláját:

```
sfdisk -d /dev/sda >
➥particios.tabla
```

Majd írjuk ki a másik (jelenesetben üres) merevlemezre:

```
sfdisk /dev/sdb <
➥particios.tabla
```

Ha ez megvan, módosítsuk a */dev/sdb* partíciós tábláján található partíciótípusokat *FD*-re (*Linux raid autodetect*), még a swap-et is.



1. lista

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    defaults        0       0
/dev/md6        /               ext3    defaults,errors=remount-ro 0       1
/dev/md8        /home           ext3    defaults        0       1
/dev/md9        none            swap    sw              0       0
```

Barátunk, az mdadm

Hozzuk létre sorban az `md` eszközöket:

```
mdadm -C /dev/md6 -l 1 -n 2
➥/dev/sdb6 missing
```

Ezzel létrehoztunk egy egylemezes *RAID1* tömböt. Látni fogjuk, miért így kell eljárni. Hozzuk létre a fájlrendszert:

```
mkfs.ext3 /dev/md6
```

Ezután másoljuk át a */dev/sda6* fájlrendszer tartalmát a /dev/md6 fájlrendszerbe. Ehhez az szükséges, hogy mindkettő fel legyen mountolva. Miután ezzel megvagyunk, hozzuk létre a */etc/raidtab*-ot a */dev/md6*-en, amiben az alábbi bejegyzések sorakoznak (ez csak egy partíciót ír le), igény szerint kell többszörözni:

```
raiddev /dev/md6
        raid-level 1
        nr-raid-disks 2
        nr-spare-disks 0
        chunk-size 4
        persistent-superblock 1
        device /dev/sda6
        failed-disk 0
        device /dev/sdb6
        raid-disk 1
```

A */dev/sda6* alatti `failed-disk` bejegyzéstől ne ijedjünk meg, később majd ha minden rendben, eltávolítjuk. A *RAID* tömbön létre kell hozni a megfelelő */etc/fstab* fájlt. Nálam így néz ki: *1. lista*.

(re)Boot

Ezek után a *GRUB*-ot vagy a *LILO*-t konfiguráljuk. Ha kész, bootoljuk újra a rendszert a *RAID*-ről (jelenesetben */dev/sdb*). Ekkor a `mount` parancs kiadása valami ilyesmit eredményez, ha valóban működik a *RAID*:

```
/dev/md6 on / type ext3
➥(rw,errors=remount-ro)
proc on /proc type proc (rw)
/dev/md8 on /home type ext3
➥(rw)
```

A sikeres bootolás után a forrás merevlemezen (*/dev/sda*) módosítsuk a partíciók típusát a már ismert *Linux raid autodetect* típusra. Ha kész, akkor */etc/raidtab* fájlban írjuk át a `failed-disk` sorokat `raid-disk`-re. Adjuk hozzá a forrás partíciókat a meglévő *RAID*-hez:

```
mdadm -a /dev/md6 /dev/sda6
```

Ha minden rendben akkor ezt írja:

```
mdadm: hot added /dev/sda6
```

Majd elkezdődik a szinkronizálás (*cat /proc/mdstat*):

```
md6 : active raid1 sda6[2]
➥sdb6[0]
      11719296 blocks [2/1]
      ➥[U_]
      [>.....................]
      ➥recovery =  3.0%
      ➥(361216/11719296)
      ➥finish=3.6min
      ➥speed=51602K/sec
```

Innen megtudhatjuk, hogy hol tart a *szinkronizálás*. A szinkronizálás elég sokáig eltarthat nagyobb partíciók esetén. Ilyenkor érdemes elmenni ebédelni.
No igen, amiért a *RAID*-en újra létre kellett hozni a fájlrendszert. A `dd` nem működik jelenesetben, mert a *RAID* valamiért kb. Pár kilobájttal kisebb területet ad és hibát jelez a `mount`, ha esetleg `dd`-vel és nem szinkronizálással csináltuk volna a partíciók másolását.

Jó tanácsok

Akár *RAID0*-t, akár *RAID1*-et csinálunk, fontos, hogy ne ugyanazon az *IDE* kábelen/csatornán legyen az eszköz. Ez felesleges lassulást eredményez. Az elérhető sebességről: nálam 2 darab 80 gigabájtos *SATA* merevlemez megy (52 és 62 megabájt per másodperc), a *RAID1* rendszerben ez körülbelül 40-50 megabájt per másodpercre lassul. A sebesség tesztelésére a `hdparm` parancsot javaslom a `-t` opciót használva.

Karbantartás

A *RAID* tömbjeink állapotát a */proc/mdstat* állományban figyelhetjük. Bármi hibajelenség van, arról a rendszer azonnal küld e-mail-t. Ilyenkor érdemes mihamarabb cserélni a hibás merevlemezt. Amennyiben merevlemezcsere szükséges, úgy az alábbi dolgokat kell végig csinálni: partíciós tábla létrehozása az új merevlemezen (másoljuk le a régiről az `sfdisk` paranccsal), adjuk hozzá az újat az `mdadm` paranccsal, a többit majd elvégzi a szinkronizálás.

Medve Zoltán
(e-medve@e-medve.hu)

2001-ben kezdett „Linuxolni”, de már korábban is ismerkedett a szabad szoftverek világával. Ha éppen nem a gép előtt ül, akkor fotózgat, olvasgat vagy bicajozik.

KAPCSOLÓDÓ CÍMEK

RAID a Wikipediaban:
➲ http://en.wikipedia.org/wiki/RAID

GNU Screen, a virtuális KVM átkapcsoló

Bizonyára sokszor előfordult már a kedves olvasóval, hogy egy távoli szerverre több példányban kellett volna bejelentkezni ssh-n. Aztán hirtelen le kellett csatlakoznia, de később szerette volna ugyanazt a környezetet használni. Erre tökéletes megoldást nyújt a GNU Screen.

A *GNU Screen* nagy előnye – legalábbis én ezért szeretem –, hogy ***képes félbeszakítani*** a folyamatokat és úgymond ***felfüggeszti*** a konzolt (közben persze a ***screenben*** megnyitott programok futnak tovább, mint például a ***wget*** vagy a kernelfordítás), hogy máskor más gépről ugyanazt a környezetet lássuk minden külön bűvészkedés nélkül. Így például ha otthonról belépek, ugyanazt a shell állapotot látom, amit a munkahelyen hazaindulás előtt lecsatoltam.
Jogos a kérdés: miért hívom én ***KVM***-nek a ***screent***? Egyáltalán mi az a ***KVM***? A ***KVM*** nem más, mint egy monitor-egér-billentyűzet átkapcsoló – több géphez elég egy trió. (Helyszűkében lévő rendszergazdáknak előnyös.) A ***screen*** azért virtuális ***KVM***, mert képes arra, hogy bár egy ***ssh*** kapcsolatot nyitunk, mégis több virtuális konzolt ad. A virtuális konzolok állapota lecsatlakozás esetén megmarad. A program indítása a

```
screen
```

paranccsal történik. Amennyiben már egy korábbi ***screenhez*** szeretnénk csatlakozni, úgy a parancs

```
screen -R
```

lesz. A `screen` futása alatt a `w` paranccsal tudjuk megnézni, hogy mik futnak és melyik virtuális konzolon.

```
mc@ketchup(AMD64):"$ w
 14:02:39 up  2:37,  4 users,  load average: 0,11, 0,16, 0,10
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
mc       :0       -                11:27   ?xdm?   5:06   0,76s /usr/bin/xfce4-
mc       pts/10   :pts/9:S.0       14:00   12,00s  0,01s  0,01s mc
mc       pts/11   :pts/9:S.1       14:01    1:10   0,01s  0,00s nano
mc       pts/12   :pts/9:S.2       14:01    0,00s  0,00s  0,00s w
mc@ketchup(AMD64):"$
0$ bash  1-$ bash  2*$ bash
```

Az `S` betű mutatja, hogy `screen` alatt fut, a pont mögötti szám pedig, hogy hányas ***screen*** konzolon.

Medve Zoltán (e-medve@e-medve.hu)

KAPCSOLÓDÓ CÍMEK

GNU Screen ➲ http://www.gnu.org/software/screen/

A legfontosabb parancsok

Minden parancs előtt le kell nyomni a CTRL + a kombinációt. (A kis és nagybetűk természetesen különböznek.)

`0-9` – Kiválaszthatjuk az aktív terminált, amennyiben nem létező terminálra hivatkozunk, úgy alul megjelenik az aktív terminálok listája. *-gal jelöli a program az éppen megjelenített konzolt.
`A` – átírhatjuk a virtuális terminál nevét, megkönnyíti az azonosítást.
`CTRL + c` (vagy csak `c`) – Új héj indítása új virtuális ablakban.
`CTRL + d` (vagy csak `d`) – Leválás a screen környezetről (visszacsatlakozás: `screen -R`)
`h` – A pillanatnyi képernyő lementése ***hardcopy.n*** fájlba.
`H` – Az aktuális képernyő naplózásának ki/bekapcsolása (a ***screenlog.n*** fájlban lesz a végeredmény).
`k` – Bezárja az aktuális virtuális terminált.
`CTRL + t` (vagy csak `t`) – kiírja a pontos időt, az aktuális gépet és a terhelési információkat.
`CTRL + w` (vagy csak `w`) – Kiírja az elérhető ablakok listáját.
`CTRL + x` (vagy csak `x`) – Zárolja a teljes ***screen*** (az összes virtuális konzolt).
`CTRL + \` – Bezárja az összes virtuális konzolt és befejezi a ***screen*** a futást.
`CTRL + [` (vagy csak `[`) – Másolás.
`CTRL + ]` (vagy csak `]`) – Beillesztés.

Végül, ha nem sikerült ezeket elsőre megjegyezni, úgy az on-line súgó a `CTRL + a` és a `?` lenyomásával érhető el. Azt persze nem kell mondanom, hogy ez a pár kombináció csak a jéghegy csúcsa, de a mindennapokban viszont elég szokott lenni.

Szelek szárnyán

A megújult magazinban eddig nem esett szó repülőgép-szimulátorokról. Valójában méltánytalanul, hiszen a kategória széles rajongói táborral bír. A hiányosságot ezennel pótoljuk...

Amikor előzetesen átgondoltam ezt a cikket, sokáig töprengtem azon, hogy milyen mélységben mutassam be a témát, hiszen a közismert rendszerek (*DOS* és *Win32* felületek, *Linux*, *OSX*, *BeOS*) rengeteg megvalósítással rendelkeznek. Ezek között bőven akad kereskedelmi játék, de szerencsére elérhető sok ingyenes projekt is. Mindezen túl némelyik próbálkozás harci helyzetek szimulációjára termett, míg mások a polgári repülést modellezik hitelesen. A nagy választék ellenére végül úgy döntöttem, hogy csupán egy programot mutatok be: egy kiforrott, színvonalas és békés szimulátort választottam erre a hónapra (manapság úgyis ritkaságszámba megy bármely vérmentes játék). A kiszemelt projektet *FlightGearnek* hívják.

A műfajról dióhéjban

A polgári repülést modellező „játékprogramok" igen nagy kultusszal rendelkeznek. A szót szándékosan tettem idézőjelbe, hiszen jóval többről van szó, mint játékról: a kategória manapság igen komoly színvonalat képvisel. Némely projekt képes akár valódi, kompromisszumoktól mentes pilótaképző programként is funkcionálni, ahol a generált belső világ jellemzően „térképhű", így nem meglepő a megszállott „szimulátor-rajongók" magas lélekszáma.

Elég csak a *Microsoft Flight Simulator* sorozatára gondolnunk, mellyel a szoftvergyáros megmutatta: érdemes a fizika szigorú szabályait betartva, valós paraméterekre törekedve, a békésebb játékosok igényeit is kiszolgálni, lehetőleg igényesen modellezett virtuális légtérben. Mindennek

1. ábra A 737 hatalmas teste

2. ábra F-16, baljós pozícióban

tükrében, a *Linux* felhasználóinak kifejezetten jó hír lehet, hogy az előbb említett klasszikus képességeinek „ellenében" a *FlightGear* nagyon sok területen összemérhető a műfaj koronázott királyával. Kiforrott fizikai modellezését, látványos grafikáját, életszerű helyzeteit tekintve a polgári repülést szimuláló programok nagyjai közé tartozik, annak ellenére is, hogy a cikk írásakor még nem éri el a *v1.0* verziót (a fejlesztők a friss változatot *v0.9.10* azonosítóval jelölik).

Üzembe állítás

Kezdjük rögtön a szükséges hardverek megnevezésével! Bármely játékról legyen szó, nagyon ritkán fordul elő velem, hogy minimális igényt említsek. Hűen tehát az eddig gyakorlathoz, íme az általam mért, kompromisszummentes használathoz szükséges paraméterek: 1.5 GHz órajelen futó teljes értékű *x86 CPU* (nem *Celeron*), 512 MByte *RAM*, valamint *3D* grafikus hardver 64MByte memóriával (betöltött *OpenGL* meghajtójával). Ha ezt a szintet megüti számítógépünk, akkor joggal várhatjuk el a projekttől azt, amire született: az élethű, közel realisztikus szimulációt, legalább ennyire életszerű grafikai leképezéssel. A program komoly függési rendszerrel rendelkezik, így a *Linuxon* történő használatba vételhez szükségünk lesz *GLUT* (➲ *http://freeglut.sourceforge.net*), *PLib* (➲ *http://plib.sourceforge.net*), és *SimGear* (➲ *http://www.simgear.org*) komponensekre egyaránt. (Az egyéb, kisebb függéseket nem szükséges említenem, hiszen a *Zlib*, *JPEG* könyvtárak jelenléte joggal feltételezhető bármely korszerű asztali terjesztésben).
E három fő függőség a leírt sorrendben állítandó üzembe: szerencsére könnyedén fordíthatóak, így *root*-ként, a letöltött, kibontott archívokban kiadott `./configure`, `make`, `make install` parancsokkal lehelhetünk életet a forráskódokba. Mivel a *FlightGear* szabadon elérhető, így a szimulátor forrásállományaiért keressük fel a ➲ *http://www.flightgear.org* címet. Innen indulva juthatunk hozzá a bázis *tarball*-hoz, valamint a szükséges kezdő adatcsomaghoz is. A bázist kicsomagolva, az előbb említettek szerint itt is építsük fel a bináris állományt. Ezek után hozzunk létre egy mappát *FlightGear* néven az */usr/local/share* úton, majd bontsuk ide a letöltött adatcsomag tartalmát. A szimulátor ezek után egy felhasználóként terminálra gépelt `fgfs` paranccsal indítható. Az első indítást érdemes `--enable-save-on-exit` opcióval megtoldani, mivel enélkül nem képes létrehozni a személyes beállításainkhoz kötődő **.xml* állományt (*/home/$/.fgfs/autosave.xml*). Ezen túl, szükséges lehet a `--help` kapcsoló választát is tanulmányozni, mert nagyon sok létfontosságú információt tárol.
A kissé körülményes fordítás megkerülésétől óva intek mindenkit! Akár a hivatalos honlapon is elérhető, előre fordított csomagok tapasztalatom szerint meglehetősen instabilak, és szinte kaotikus módon válogatják függéseik szükséges verziószámát, legfőképpen a menüzésért felelős *GLUT* összetevőét (holott ez a kritikus komponens nem megfelelő verzióban pillanatok alatt képes összeomlasztani a programot).

3. ábra Panoráma a Wright fivérek gépéből

4. ábra Hamisítatlan éjszakai felszállás

```
General Options:
   --help, -h                  Show the most relevant command line options
   --verbose, -v               Show all command line options when combined
                               with --help or -h

   --fg-root=path              Specify the root data path
   --fg-scenery=path           Specify the base scenery path;
                               Defaults to $FG_ROOT/Scenery

   --language=code             Select the language for this session
   --disable-save-on-exit      Don't save preferences upon program exit
   --enable-save-on-exit       Allow saving preferences at program exit
   --control=mode              Primary control mode (joystick, keyboard,
                               mouse)

Aircraft:
   --aircraft=name             Select an aircraft profile as defined by a top
                               level <name>-set.xml
   --show-aircraft             Print a list of the currently available
                               aircraft types

   --min-status={alpha,beta,early-production,production}
                               Allows you to define a minimum status level
                               (=development status) for all listed aircraft

Initial Position and Orientation:
   --airport=ID                Specify starting position relative to an
                               airport
   --runway=rwy_no             Specify starting runway (must also specify an
                               airport)
   --carrier=[name|ID]         Specify starting position on an AI carrier
   --parkpos=name              Specify which starting position on an AI
                               carrier (must also specify a carrier)
   --vor=ID                    Specify starting position relative to a VOR
   --ndb=ID                    Specify starting position relative to an NDB
   --fix=ID                    Specify starting position relative to a fix
```

5. ábra A paraméterezett indítás gyakorlatilag elkerülhetetlen

A puding próbája...

Nézzük hát meg a fáradozásaink eredményét! Paraméter nélküli indítás után az alapértelmezett környezet fogadja a pilótát: az első repülőtér kifutóján pihenő *Cessna* (kicsit sem otthonos) kabinja. Érdemes a képi finomhangolással kezdeni, és hardverünk képességeihez mérten beállítani a *„nézet/renderelés”* menüpont alatt szereplő tételeket. Megfelelő beállítás esetén a leképezett táj kifejezetten szép, a modellezett fizika valóságérzettel párosul, a virtuális világ pedig összességében is hiteles. Ezek után ismerkedjünk a többi fontos menüvel is: *„fájl”*, *„robotpilóta”*, *„helyszínválasztó”*, *„időjárás és időgenerátor”*, *„környezeti beállítások”*, *„repülőtéri irányítás”*, *„debugger”* és *„segítség”*. Ezekről értelmetlen bővebben szólnom, mivel mind logikusan és érthetően épül fel. A *„segítséget”* kezdetben mindenképpen igénybe kell vennünk, mivel a szükséges kontrollbillentyűk itt vannak megnevezve: ezek alapján már „könnyen” kiengedhetőek a fékek, könnyen állítható a tolóerő. Miután kedvencünk eléri a felszálláshoz szükséges sebességet, óvatosan emeljük fel, majd húzzuk be a futóművét és próbálgassuk a légtér szabályait! Nem egy leányálom, példának okáért igazi megpróbáltatás egy ***F16***, vagy éppen egy ***Boeing 737*** navigálása: a gépek gyors manőverek hatására könnyedén átesnek, minek eredménye a talajba fúródott géptest. Ezen a ponton érdemes beszerezni egy megfelelő minőségű, ***Linux*** alatt is működő, tolóerő-szabályozós botkormányt.

Megéri tehát kitapasztalni mindegyik repülhető „madarat”, a szemfülesek pedig az adatcsomagban lévő ***UFO***-t is meghívhatják néhány látványos tiszteletkör erejéig. Kiegészítő repülők és küldetések az említett hivatalos honlapon érhetőek el. Emellett egy lényeges dologra még ki kell térnem: a projekthez mellékelt dokumentációban világosan le van írva (többek között) az is, hogy miként lehet a ***Microsoft Flight Simulator*** modelljeit és küldetéseit importálni a ***FlightGear*** rendszerébe. Kívánhat-e bárki többet?

Zárszó

A mostani „játéktéma” kitüntetett figyelmet érdemel, hiszen az emberek érdeklődésének mindig kiemelt szerepét töltötte be a repülés gondolata, nem is beszélve arról, ahogy valószínűleg sokunk „álmodta bele” magát egy vadászgép kabinjába gyermekkorában... Itt az alkalom, most bárki próbára teheti rátermettségét. Nem egyszerű feladat levegőben tartani a tonnás gépmadarakat, de a téma iránt érdeklődő, tapasztalt „játékosok” bizonyára élvezni fogják a kihívást. Aki pedig ennél is komolyabban szeretné megismerni a repülés és a navigáció sajátosságait, a ***FlightGear*** használatba vétele előtt mindenképpen látogasson el a ➲ ***http://www.navfltsm.addr.com/*** oldalra, ahol részletes információt talál a bevált technikákról.

Tartalmas időtöltést kívánok mindenkinek!

Kovács Zsolt (kovi@linuxforum.hu)

Quake fanatikus. Négy éve a debreceni linuxosok egyike. Töretlenül hisz a Slackware terjesztésben.

3D ábrázolás – PoVRay (8. rész)

A PoVRay által készített képek minősége arányos az elkészítésére (renderelésre) fordított idővel. Jobban mondva egy kép minőségét duplájára emelve a szükséges idő négyzetesen növekszik. Egy-egy kép esetén kivárható időket kapunk, azonban animációk esetén több száz képkocka elkészítése napokba is kerülhet. Mikor még tíz éve egy 486/66MHz gépen műveltem ezt a tudományt, egy 1024x768 felbontású kép másfél órán át készült.

A felbontás változtatása

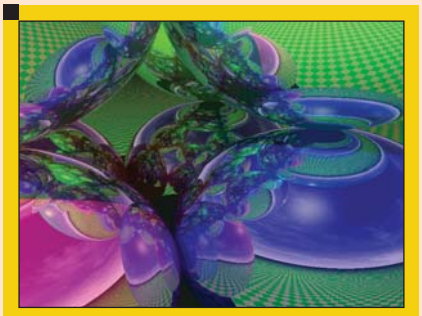
Egy kép minőségére leginkább a felbontása a jellemző, amelyet a pixelek száma határoz meg. A felbontással óvatosan kell eljárnunk, a vízszintes és a függőleges méret arányát a használt kamera határozza meg, ha eltérő az arány, akkor torzul a kép.

```
camera{
 up y
 right 1.0*x
 location <10,10,20>
 look_at <4,4,4>}
```

A kameránál a `right` kulcsszó adja meg az arány, amely szokványos képernyők esetén `1.333*x`, vagyis 4:3 az arány. Ha fotó formátumot szeretnénk, amely 3:2 arányú (például 15x10), akkor `1.5*x` kell a `right` kulcsszó után. A fenti példában a képformátum aránya 1:1, ha ezt 4:3 arányú képre készítjük el, akkor torzítva jelennek meg a gömbök (*1. ábra*), kissé laposabbak lesznek, hiszen csökken az ábrázolt magasságuk.

Progresszív renderelés

Nagyobb projektek esetén célszerű progresszív renderelést választani, amely esetén a ***PoVRay*** először nagyobb négyzeteket összevonva jelenít meg, majd ezeket fokozatosan finomítja (*2. ábra*). Több perces elkészítési idő felett elnagyolva láthatjuk a végeredményt, s megszakíthatjuk a renderelést, ha nem megfelelő a tárgyak elrendezése vagy a kinézete. Általában a megfelelő kameraállás kikísérletezésénél jön jól. Az elkészült kép nem lesz progresszív (néhány képformátumnak van ilyen tulajdonsága). A progresszív módot az `SP`, mint a kezdő négyzet pixelmérete, illetve az `EP`, mint az utolsó progresszív négyzet pixelmérete határozza meg.



1. ábra Torzult gömbök

2. ábra Progresszív renderelés

```
$ povray +W800 +H600 +P +EP2
➥+SP32 last01.pov
```

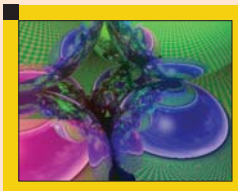
A sugárkövetés minősége

Mivel a ***PoVRay*** sugárkövetéssel dolgozik, sok opciója van a minőség beállítására. Elsősorban a közvetlen képminőséget tudjuk befolyásolni, amelyet egy 0-9 közötti számmal kell megadnunk, ahol a 0 a legrosszabb képminőség (*3-5. ábra*):

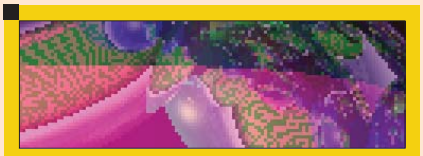
```
$ povray +W800 +H600 +P +Q4
➥last01.pov +Olast04.png
```

3. ábra Képminőség: 0

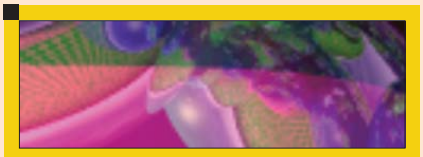
4. ábra Képminőség: 4



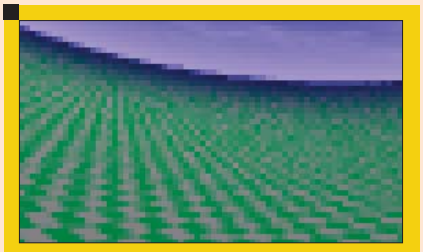
5. ábra Képminőség: 9



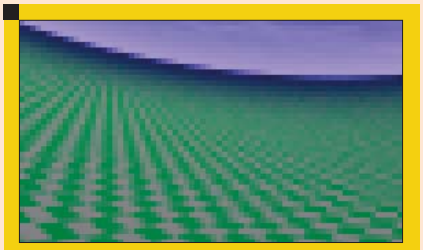
6. ábra Élsimítás nélkül



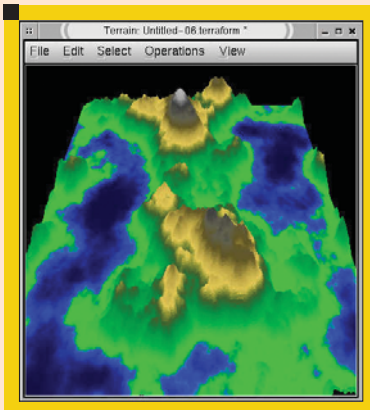
7. ábra Élsimítással



8. ábra Élsimítással



9. ábra Élsimítással és rekurzióval



10. ábra Terraform: hegyvidék

Élsimítás

A raszteres – vagyis pixeleken alapuló – képábrázolás egyik legrosszabb tulajdonsága az egyenesek ábrázolása, ugyanis ezen esetben törések lesznek a pixelek határán. Az élsimítást egy 0 és 1 közötti számmal tudjuk beállítani, minél közelebb megyünk a nullához, annál jobb képet kapunk eredményül (s annál tovább tart a renderelés is).

```
$ povray +W800 +H600 +P +A0.05
➥last01.pov +Olast06.png
```

Ahogy a *6-7. ábrán* is látszik, az élsimítás hatása, a *7. ábrán* élsimítással ugyanakkora felbontáson a kép sokkal szebben néz ki, eltűntek a pixelek közötti durva színbeli ugrások. Ennél jobb minőséget érhetünk el, ha az élsimítást többszörös rekurzióval kombináljuk, bár ezzel sokszorosára növekszik a kép elkészítésének ideje, de az eredmény kárpótol majd.

```
$ povray +W800 +H600 +P +A0.05
➥+R20 last01.pov +Olast09.png
```

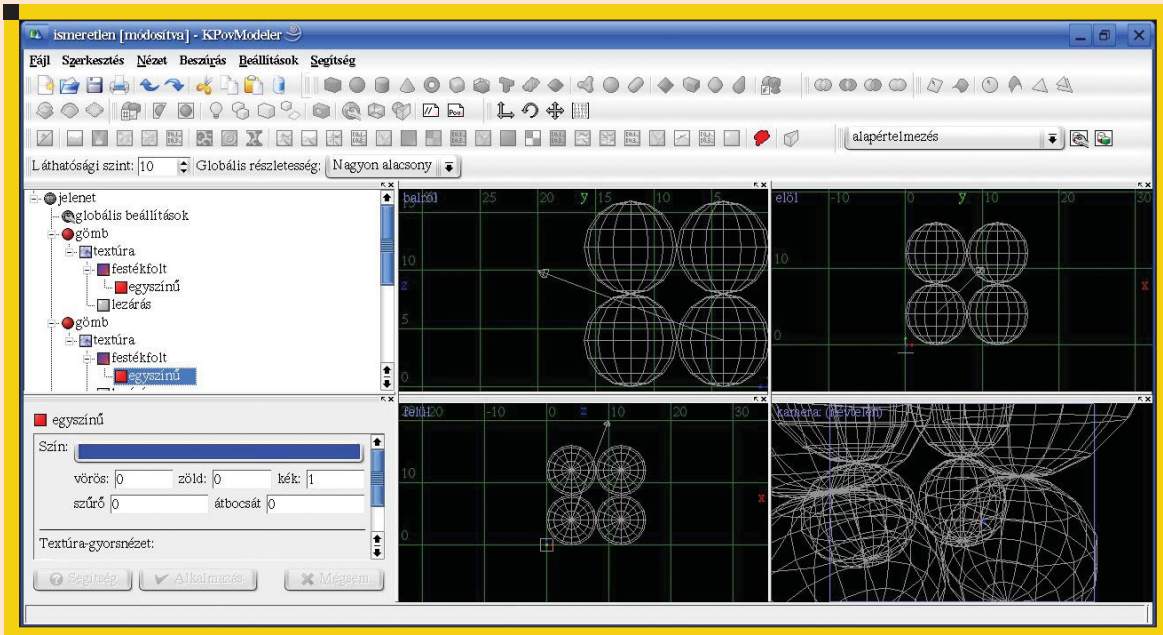
Minél nagyobb számot használunk rekurziónál, annál jobb lesz a képünk minősége olyan részekben, ahol egy-két pixelnyi információt kell megjeleníteni: ilyen a kockás padló tükörképe. A két kép különbségéből látható, hogy a rekurzióval készített képen több részlet látszik, holott a felbontás azonos.
Egy komolyabb 3D világ renderelése – átlagos felbontással és erőteljes élsimítással akár egy-másfél óráig is eltarthat egy vadonatúj 3GHz-es gépen.

Előtét programok

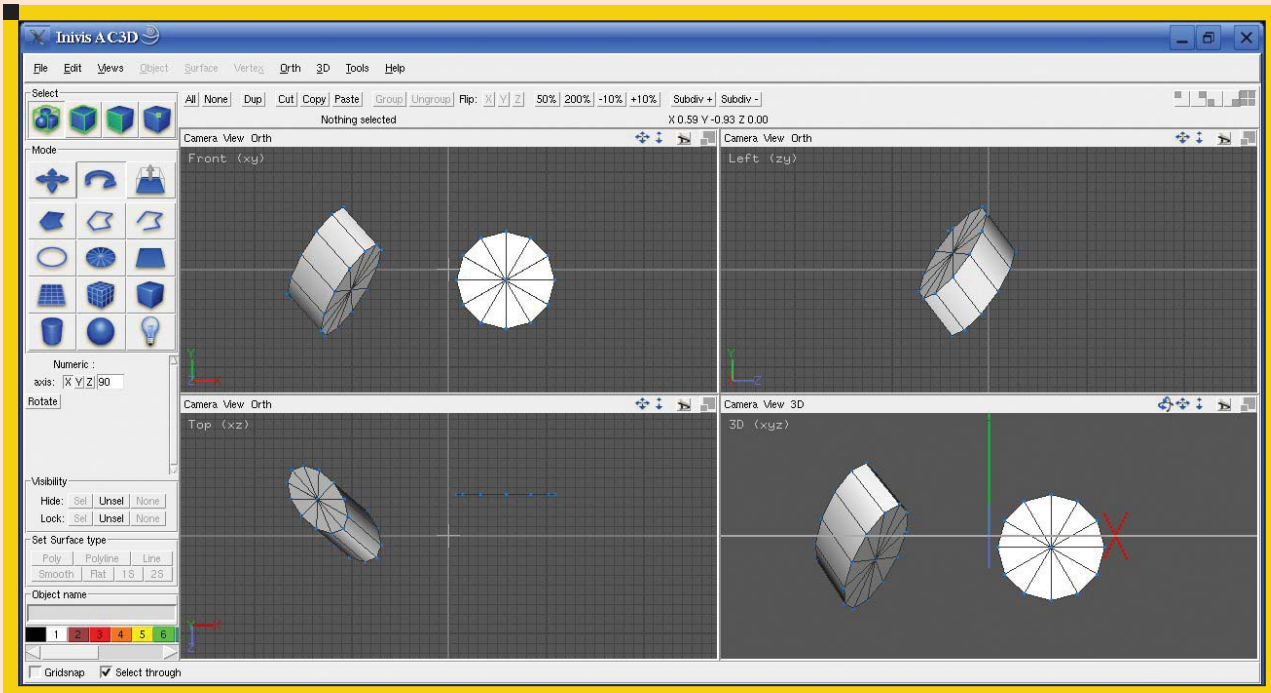
Többször is említettem, hogy a ***PoVRay*** nem igazán alkalmas önmagában való használatra, hiszen a nyelvezete és a vizuális kontroll teljes hiánya nehézkessé teszi a *3D* világ felépítését, az animáció pedig teljesen megöli a szárnyaló fantáziát. Programozói vénával nem rendelkező grafikus emberek pedig elborzadnak a gondolattól, hogy egy *3D* ábrát szöveges felületen készítsenek el.

Terraform

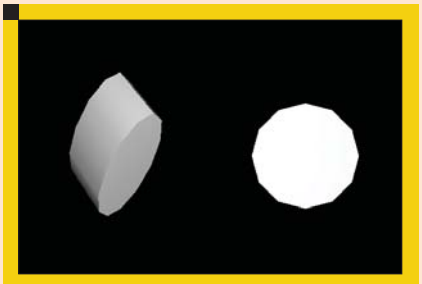
Az egyik kedvenc programom a ***Terraform***, amellyel tájat lehetett készíteni, hegyeket, folyókat és tavakat,



11. ábra KPovModeler: a cikkben szereplő nyolc gömb



12. ábra AC3D: két poligon alapú tárgy



13. ábra AC3D: a PoVRay export renderelve

ezeket magasság szerint színezni, textúrázni. Sajnos az utolsó kiadása négy éves, bináris formában nem tudtam már feltelepíteni, a lefordítását pedig pár órás munka után szintén feladtam: a program egyszerűen olyan régi, hogy a mostani disztribúciókon már nem fut (az *OpenSuSE 10.2 Alpha1* pedig különösen friss). Remélem akad valaki, aki ezt a remek programot újraéleszti.

KPovModeler

A *Terraform* fejlesztésének befejezésével egy időben jelent meg a *KPovModeler*, amely egy átlagos *3D* modelleket kezelni tudó program kinézetével bír. A fő munkaterület a *3D* világ három nézetét (balról, elölről, felülről) és a kiválasztott kamera képét mutatja. A tárgyakat meg tudjuk fogni az egérrel, illetve egérműveletekkel átméretezni, áthúzni, elforgatni is képesek vagyunk. A tárgyaink egy fa struktúrában szerepelnek, amelyben ki tudjuk választani egyenként is a szükséges elemeket. A magyarítása kissé hiányos (illetve inkább érdekes :), ennek ellenére kis töprengéssel rá lehet jönni, hogy a „lezárás” szó alatt a *PoVRay* `finish` kulcsszóját értik. A program írni és olvasni tudja a POV állományokat, viszont eléggé szűk dialektust beszél: kizárólag a 3.5-ös *PoVRay* aktuális szintaxisát érti, s nem követi nyomon az `include` állományokat sem.

AC3D

Az *AC3D* egy fizetős program, egy 14 napos próbaverziót tudunk megnézni belőle, tudása és használata kilóg a többi közül, elsősorban poligon alapú modellezésre készült, viszont képes exportálni *PoVRay* formátumba bármilyen elterjedt *3D CAD* programból (*3D Studio*, *AutoCAD*, stb), illetve játékok állományiból (*Quake3*, stb).

Zárszó

Őszintén remélem, hogy a *PoVRay* használatát bemutató kis sorozatom nem riasztott el senkit a mélyebb megismeréstől. A program jelentős megkötésekkel használható, főképp háttérként szolgál sok olyan programot segítve, amelyeknek írói nem akarnak bajlódni egy *3D* modellező megírásával, viszont a *PoVRay* leíró nyelvét elsajátítva a programjuk képes lesz szemléletes ábrák készítésére és animálására (molekula modellezők, orvosi felhasználás, ipari berendezések animációi, stb).

Auth Gábor
(auth.gabor@enaplo.hu)

Egy pécsi középiskolában informatikát és programozást oktat. Tíz éve botlott először a UNIX rendszerekbe, 7 év Linux használat után kapta el a FreeBSD lázat, amiből máig nem tudott kigyógyulni.

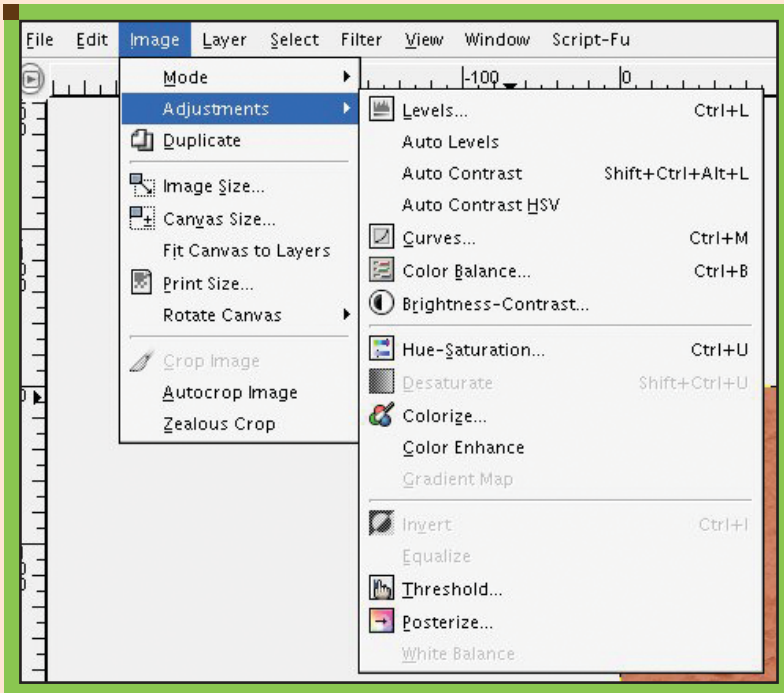
KAPCSOLÓDÓ CÍMEK

A PovRay projekt honlapja
➲ http://www.povray.org
A cikkben említett fájlok
➲ http://user.enaplo.hu/~auth.gabor/pov/
Terraform
➲ http://terraform.sourceforge.net
KPovModeler
➲ http://www.kde.org
AC3D
➲ http://www.ac3d.org

Gimpshop, a trónkövetelő

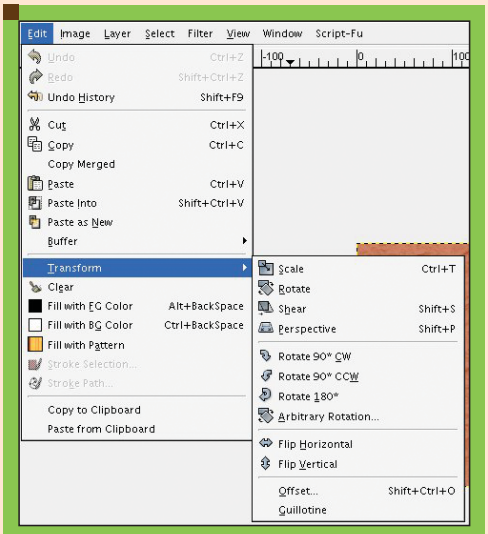
Ki ne hallotta volna még az Adobe Photoshop nevet? Kiváló program. Nem véletlenül vált a pixelgrafikus programok egyik de facto szabványává. Sokan tanultuk meg a képszerkesztés alapelveit a használata közben. Aki használta már, az tudja, hogy rendszeres használat alatt az ember fejébe vésődnek a billentyűkombinációk, és mindenféle gondolkodás nélkül, automatikusan megtalálja a menüben a szükséges parancsokat. Ennek azonban ára van.

Keserves perceket fogunk eltölteni akkor ha egy másik programmal, történetesen a *Gimppel* kell valamit gyorsan megcsinálnunk.
Sokan felváltva használják a két programot, attól függően, hogy az adott számítógépen melyik van telepítve. Aki már foglalkozott grafikai tervezéssel, az azt is tudja, hogy a legtöbb cégben a *Photoshop* ismeretét követelik meg. Ezzel a programmal talán az egyetlen gond az, hogy *Linux* alatt szinte lehetetlen életre lehelni. Pedig sokan megvásárolnák, ha lenne valami használható megoldás a futtatására. (*Crossover Office* segítségével a 7-es verzió szinte teljes funkcionalitásában életre kelthető. Az újabb kiadások azonban már nem.)
Tudásban alig van különbség a *Gimp* és a *Photoshop* között, de sajnos felépítésük, kezelésük módja elég eltérő tud lenni. Okoz is fejtörés a sok különböző billentyűkombináció és az eltérő menüstruktúra.
Az ideális megoldás az lenne, ha mind a két program alatt ugyanott lennének a menük, megegyeznének az eszközök nevei és a billentyűzet is hasonlóképpen működne. Nincsen erre valami megoldás? Hiszen ha lenne, akkor szinte fájdalom mentesen használhatnánk *Gimpet* akkor, ha nem lenne pénz egy kereskedelmi szoftverre, vagy olyan gépen, operációs rendszeren dolgoznánk, amelyiken nem tudjuk azt telepíteni és mégsem szeretnénk két különböző program minden csínját megjegyezni.
Scott Moschella éppen ezekre a problémákra kereste a választ amikor megalkotta a *Gimpshopot*, mely egy *Gimp* változat. Változtatott a kifejezéseken és az eszköz neveken, átrendezte a menüket, átírta és átalakította a az egyes dialógus ablakok tartalmát és megváltoztatta billentyűzet kombinációkat.
Hosszas munkával elérte, hogy



1. ábra Az Image menü szerkezete pontosan megegyezik a Photoshop alatt megszokottal

Scott Moschella blogjában a következőeket írta: „A *Gimp* menü struktúrája és elnevezései állandóan frusztráltak , mert *Photoshopot* használtam korábban. Így azt tettem, amit bármelyik kemény kóder tett volna, összeraktam a *Gimpshopot*. Kísérleti jellegű még és van benne jó pár hiba is, úgyhogy ne várjatok tőle sokat.”

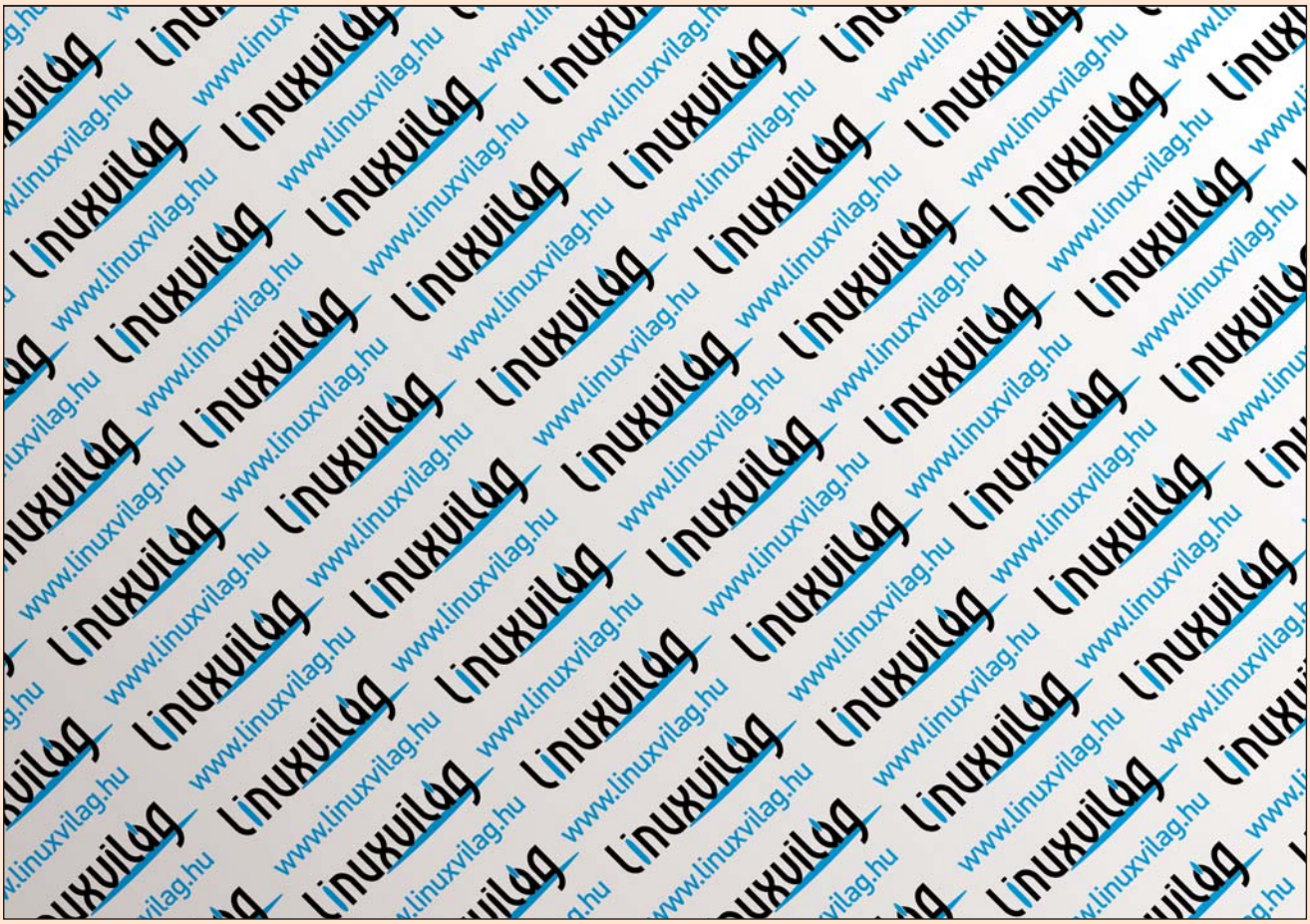


2. ábra Az Edit menüpont alatt találhatjuk a transzformációkat, pontosan ugyanúgy, ahogy Photoshop alatt is

a *Gimp* a lehetőségekhez képest a legjobban hasonlítson kereskedelmi testvérére.
A legfrissebb Gimpshop verziót a fejlesztő oldalán találhatjuk meg, a ➲ http://plasticbugs.com címen.
Natív *Windows* és *OSX* telepítő mellett a forrás és egy *.rpm* csomag is elérhető. *Debian* és *Ubuntu* alatt talán a legegyszerűbb ha az *.rpm* csomagot alakítjuk át deb csomaggá az alien program segítségével, de persze le is fordíthatjuk magunknak ha kedvünk tartja.
A billentyűkombinációkat definiáló *menurc* nevű fájlt is az oldalon találhatjuk. Ezt bármilyen *Gimp* változattal gond nélkül lehet használni, még akkor is, ha magát a *Gimpshopot* nem tudjuk vagy nem szeretnénk telepíteni. (A *menurc* fájl pontos címe: ➲ *http://epierce.freeshell.org/gimp/gimp_ps.php*)
Lássuk tehát miben más a *Gimpshop*! Vajon sikerült elérni a kitűzött célt? Mindenképpen! Ne feledjük egy pillanatig sem, hogy a cél nem a *Photoshop* egy az egyhez való lemásolása volt, hanem hogy azok számára megkönnyítse az áttérést, akik sokat dolgoztak, dolgoznak ezzel a kereskedelmi termékkel.
És ezt a célt a fejlesztő el is érte. Szinte minden menüt ott találunk ahol már megszokhattuk, a szakkifejezések is megfelelnek a *Photoshop* alatti terminológiának. Apró eltérések perszer előfordulnak, például egyes menüelemek más sorrendben követik egymást olykor, de azt hiszem ez senkinek nem okozhat már gondot.
Összességében kitűnő munkát végzett *Scott Moschella*. Ha a fejlesztés lendülete megmarad, sokan elgondolkozhatnak azon, hogy *Linuxot* használjanak grafikai munkára. Hiszen ezentúl már az átállás sem kerül túlzott agymunkába.

Juhász Attila
(rabszolga@goraffe.hu)

Az Információ Technológiai Kar hallgatója a Pázmány Péter Katolikus Egyetemen. Érdeklődik a bioinformatika és a neurális hálózatok iránt. A fotózás és a tánc mellett öt éve foglalkozik webgrafikákkal. A linux terjesztések közül a Gentoo és az Ubuntu áll legközelebb a szívéhez. Fotós oldala a http://people.goraffe.com/attila címen található.



A Földtől távol

A játékrovat e havi részében belső nézetű „űrszimulátorokról” lesz szó. Ebben a műfajban két olyan versenyzőt fogok megnevezni, melyek linuxos binárissal és mérhető múlttal egyaránt rendelkeznek, platformfüggetlen klasszikusokhoz méltóan.

Descent 3

A *Descent* nevet valószínűleg igen sokan ismerik azok közül, akik az *x86* alapú *PC*-ken már a korai időktől kezdve keresték az önfeledt kikapcsolódás lehetőségét. A most említett játék első két részével a fejlesztő csapat figyelemre méltó munkát végzett, végérvényesen beírva magát játékipar nagyjai közé. Belső nézetű játékaik a távoli jövőbe kalauzoltak el, ahol apró űrhajónkkal kellett helytállnunk programvírussal fertőzött bányarobotok ellenében, bonyolult és zárt belső térben. Ezek a programok (az akkor aktuálisan sláger *Doom* és *Quake* ellenében) teljesen vérmentesek voltak, valamint az „emberszimulátoroktól” eltérően reflexeinken túl erősen igénybe vették három dimenziós tájékozódási képességeinket is. Természetesen a harmadik rész sem hazudtolja meg a műfajt: az *arcade* jellegű *űr-FPS* stílus valaha létezett próbálkozásainak egyik vezető szerepét tölti be, mivel érdekes fizikája, kiváló grafikája és élvezetes játékmenete még évekkel a megjelenése után is kategóriájának felső szintjét képviseli. Az utolsó epizód elődeinek minden kellemes tulajdonságát örökölte, sőt a fejlesztő *Outrage* ahol csak lehetett, rá is tett egy lapáttal az említett színvonalra, igen nehéz házi feladatot adva létező és leendő konkurenciának.

Maga a Játék

A *Descent3* egy tartalmas, (bár kissé kopottas grafikájú) intróval nyit: fantasztikus zenei aláfestéssel tálalva, külső nézetből élhetjük át, ahogyan tehetetlenül száguldunk az izzó *Nap* felé, amikor egy váratlanul felbukkanó hajó vonónyalábjai megmentenek a biztos pusztulástól. Itt léphetünk be abba a virtuális világba, ahol az ellenséges robotok által nehezített, *FPS* játékokra jellemző küldetésalapú felfogásban kell pályáról pályára haladva végrehajtanunk kijelölt feladatainkat. Elődeitől eltérően, a játékmenetben előrehaladva itt egy erőteljes történeti szál is kibontakozik, melyben az ellenséges *PTMC* vállalat körüli árulás, vagy éppen orgyilkosság teszi még élvezetesebbé a képernyő előtt töltött órákat. A *Pyro* (így hívják az apró hajót) fegyverrendszere az előző részekhez képest fejlődött, ám nem hordoz kirívó meglepetéseket: szimpla lézertől kezdve, vulkán gépágyún át akár célkövető rakéták képezhetik felszerelésünket. Viszont, hogy ne csak a *Pyro* legyen terítéken, ebben

1. ábra A Descent3 linuxos telepítője

2. ábra Hatalmas külső tér

3. ábra A GuideBot „fertőtlenít"

a részben már három vadász is repülhető! Mindezek felett programoztak nekünk egy hűséges társat is, a *GuideBot*-ot. Ez egy javarészt önálló életű robot (alapértelmezetten az *F4* billentyűvel hívható elő), melyet munkára küldve instrukcióink szerint felméri a terepet, felkutatja az extra tárgyakat, megmutatja a feladatainkhoz szükséges helyes útirányt, vagy éppen a ellenségeinket bénítja meg valamely antivírus kóddal. A program oktató részét mindenképpen javallott végigcsinálni, hiszen űrhajónk irányítása (*arcade* jellegének ellenére) némi rátermettséget és rutint igényel. A mérhető rutin mindenképpen feltétel, mivel a 15 pályás kaland kezdetén berepülendő játéktér ugyan szerény méretű, azonban a szükséges tájékozódási képesség a későbbiekben mind inkább főszerephez jut a nyakatekert járatokban. A látványra sem lehet panaszunk, a leképezett fizikáért és képi világért felelős *Fusion(TM)* motor mesterien teszi dolgát: az effektekben gazdag grafika még így, „megöregedve" is tiszteletet parancsoló, nem mellékesen játszi könnyedséggel váltogatja a külső és belső tereket. A *Descent3* természetesen *multiplayer* üzemmóddal is bír.

Használatba vétel

A két *CD* -t elfoglaló játék 1999-es megjelenése csupán a *Win32* platformot érintette. Röviddel eztán került képbe a *Scott Draeker*-féle legendás *Loki Entertainment*, mely a tisztán *Windows*-specifikus programot átporolta *x86 Linux* rendszerekre: így született meg a linuxos *Descent3*. Mivel a játék meglehetősen régi (a számunkra érdekes verzió *2000 júniusában* látott napvilágot), így hardverigénye mai léptékkel igen szerénynek számít: *800 Mhz* órajelű *x86 CPU* és *32Mbyte* tárral szerelt *3D* grafikus kártya mellett (elsősorban *GLX* kapcson keresztül, lásd később) már nagyobb felbontásokban is élvezhető. Sok gondunk nem akad a telepítéssel: az első lemezt befűzve, annak gyökerében található `setup.sh` szkriptet kell indítanunk rootként. Ezáltal a bináris állományok alapértelmezett helyükre kerülnek, valamint (igényeink szerint) a lemezen található adatcsomagok is merevlemezünkre „költözhetnek". Utóbbi esetet választva a játék idejére nem fog kelleni a telepítő lemez, a teljes installálás ekkor viszont már bő *500 Mbyte* helyet igényel. Az első diszken lévő bináris sajnos nem hibátlan, így telepítés végeztével azonnal frissítést kell rajta eszközölnünk. Sajnos a *Loki* már nem létezik, ergo a ➲ *http://www.lokigames.com* honlapnak csak igen kevés funkciója működik, így a foltért el kell látogatnunk a néhai legenda FTP-szerverének egyik tükrére. Az ➲ *ftp://sunsite.dk/mirrors/lokigames/patches/descent3/* címről letöltött *Descent3-1.4.0a-x86.run* állományt elindítva, majd a program telepítési útját megadva a játékmotor fontos változáson esik át (a frissítéshez tartozó *changelog* lista szerint). A játék terminálra gépelt `Descent3` paranccsal indítható.

Végezetül néhány apróságra szeretném felhívni a figyelmet: a linuxos *Descent3 OpenGL* bázisú grafikája tapasztalatom szerint sok esetben nem állja az *ATI* grafikus hardvereit (akár gyári meghajtóval, akár külső *DRI* modullal hajtva), így *ATI* kártya mellett ez a csodaszép virtuális világ gyakran csak egy idegesítő program benyomását kelti. Fontos lehet továbbá az is, hogy a programkód *cdrom* nevű eszközt és *cdrom* nevű csatolási pontot használ lemezeinek olvasásához – így ha ezek nem léteznek a */dev*, illetve a */mnt* mappánkban, akkor a */home/$/.loki* rejtett úton lévő */Descent3* alkönyvtárban megpróbálja ezeket ideiglenesen létrehozni.

Apropó, lemezek! A második csupán a videóbetéteket tartalmazza, így természetesen nem kötelezően használandó a játékhoz: ha nem akarjuk ezt a *CD*-t behelyezni futtatás idejére, akkor a program erre kérő üzenetén nyugodtan nyomjuk meg a *Cancel* gombot. A linuxos verzióból (a *Win32* változathoz hasonlóan) demó is készült, ami által bárki által szabadon kipróbálható ez a műremek. A *40 Mbyte* méretű „próbaanyag" ma már csupán a ➲ ftp://jeuxlinux.com/lokigames/demos/descent3-demo.run címen érhető el. Időközben napvilágot látott egy hét új küldetést tartalmazó kiegészítő lemez is, *Mercenary* névvel. Ennek sajnos nem létezik demója, mindazonáltal az előbbiekben linkelt *Loki ftp* tükrén megtalálható ennek *Win32* változatához tartozó, natív linuxos binárist tartalmazó telepítője.

Freespace 2

A *Freespace* játékok a *Descent* sorozattal ellentétben sokkal inkább szimulátoros tartalmúak. A *Volition* üdvösékjének nincs könnyű dolga, hiszen a harci űrszimulátorok kategóriája „nehézsúlyú" múlttal rendelkezik, elég csak a *Wing Commander* vagy az *X-Wing* játékok elképesztő karrierére gondolnunk. A *Freespace* első része gyakorlatilag a semmiből került elő, ennek ellenére sikeresen vette az akadályt, és igényes kivitelezése miatt rövid időn belül klasszikussá vált. A szimulációs részhez tartozott egy

4. ábra A Freespace2 telepítője

5. ábra A Freespace2 világa

középszerű történet, amiben az emberek és a ***Vasudanok*** egymás ellen vívott harcát megzavarta egy idegen erő, mely ellenében csak a két (korábban ellenséges) hadtest összefogása jelenthetett megoldást. A második rész történeti magja szerint az emberek és ***vasudaniak*** szövetségéből kialakuló rendszert a ***Polaris*** térben kitört polgárháború zavarja meg, mely káoszt „megtoldva” ismét felbukkan a korábbi ***Shivan*** ellenség, mely fejlett haditechnikát birtokol.

Profi kivitel

A játék szimulációs része nagyon élvezhetőre sikerült, melyben vadászunk összetett irányítása mellett egy komoly fizikai modell is „nyomja a mérleget”. A program küldetés alapú, melyben a bevetések előtt természetesen megkapjuk a szükséges eligazítást. Mindenek előtt azonban a kiképzőprogramot végig kell csinálnunk. Enélkül ebben a játékban nem leszünk értékelhető pilóták. A ***Freespace*** második része közel 70 hajótípust vonultat fel hatalmas méretű, virtuális univerzumában, ahol a gépi ellenfeleink intelligenciája meglehetősen komoly és nehéz harcokat garantál. Azt kell mondjam, aki az ebben a játékban látott csatáknál színvonalasabbat akar mutatni, annak komolyan fel kell kötnie a gatyáját: a szédületesen nagy csatahajók egymással vívott harcai (példának okáért) monumentálisabbak, mint egy korszerű ***sci-fi*** mozi, és ebbe a csatába bizony akár be is repülhetünk saját hajónkkal. Virtuális űrhajónk manőverezési dinamikája a ***Wing Commander***-ben megismert fizikához hasonlít leginkább, és ez szintén komoly ajánlólevél az „űrharcosok” felé...

Telepítés

A játék a ***Descent3***- hoz hasonlóan, kezdetben csupán ***Win32*** felületre volt elérhető. Mivel a fejlesztő ***Volition*** nem tervezett linuxos binárist (és gyakorlatilag ekkor már sajnos a ***Loki*** sem létezett) így sokáig kellett várnunk a rendszerünkön is életképes verzióra. Annak ellenére, hogy a ***Volition*** csapata kezdetben csupán a ***Windows*** mögötti kereskedelmi potenciállal számolt, úgy később (szimpatikus módon) a játék motorjának forráskódját megnyitotta az ***Icculus*** csapata felé. Ők pedig szokásuk szerint ***CVS*** rendszerben elkezdték ezt polírozni, majd 2003 derekán egy életképes forrásverzióból felépítették a natív binárist, melyet „összedrótoztak” egy ***Loki***-alapú telepítővel. Eredményként ***2003*** óta a ***Linux*** felhasználók is élvezhetik ezt a remekművet saját gépük (és ***Szabad Rendszerük***) előtt ülve. A telepítés pofonegyszerű: a ➲ http://www.icculus.org/freespace2/ oldalról letöltött, majd rootként indított ***freespace.verzió.run*** telepítő a ***Win32 Freespace2*** telepítő korongjait bekéri a felhasználótól, melyekről (a kívánt tartalmat megjelölve) a kinyert adatállományok helyükre kerülnek, a natív port mellett. A program ezen a ponton is erősen eltér a ***Descent3***-tól, hiszen itt nincs kereskedelmi forgalomban lévő külön linuxos verzió, hanem „csak” egy ingyen elérhető / telepítőbe „drótozott” bináris, mely házasítható a játék ***Win32*** telepítőjével (és emiatt a linuxos ***Freespace2*** könnyebben beszerezhető, mint az előbb említett társa). Helyigény tekintetében, teljes telepítés esetén nagyjából ***2 Gbyte*** területtel kell számolnunk. Mai szemmel nézve már egy átlag alatti ***600 MHz*** órajelű ***x86 CPU***, és ***32 Mbyte*** tárral szerelt ***OpenGL*** grafikus hardver élvezhető játékmenetet garantál. Fontos, hogy a videóbetétek hangja sajnos nem működik, így ezzel a hiányossággal mindenképpen számolnia kell a leendő felhasználónak.
A szimulátor terminálra gépelt `freespace2` paranccsal indítható.

Zárszó

Ebben a műfajban természetesen léteznek egyéb életképes megoldások is: a ***Descent*** előző részeinek forráskódjából újjászületve e nagy elődök ***D2X-XL*** néven akár ***fork/remake*** projekttel is rendelkeznek, de a ***Vegastrike***-ot is említhetném, mint népszerű játékot. Szándékosan hagytam ki ezeket, mivel meggyőződésem, hogy a műfaj linuxos húzóerejét a most említett, két kiforrott kereskedelmi program jelenti. Így, aki komolyan gondolja a vadászatot, az a két név valamelyikébe előbb vagy utóbb belefut, nem véletlenül.

Kovács Zsolt (kovi@linuxforum.hu)

Quake fanatikus. Négy éve a debreceni linuxosok egyike. Töretlenül hisz a Slackware terjesztésben.